Mutualistic Software Services (MSS) for Large-Scale Networks

A Project

By John P. Quan

Presented to the faculty of University of Alaska Fairbanks

In Partial Fulfillment of the Requirements of

MASTERS IN SOFTWARE ENGINEERING

Fairbanks, Alaska

April 2012
Abstract

Think of a large-scale network (LSN) as a geographically separated collection of servers, or server clusters, interconnected by the Internet to achieve some common purpose, such as communication, computation, or experimentation. Several experimental LSNs have taken root across the globe, which span international boundaries through joint collaboration to develop a secure, robust global Internet. For instance, the Future Internet Research and Experimentation (FIRE) program aims to boost European innovation and its competitive role in defining future Internet concepts. It creates an open research environment, which gives researchers the opportunity to conduct large-scale experiments. FIRE will eventually incorporate users at all levels. It has identified two key challenges: security and federation (which includes governance, sustenance, and incentives). Similarly, Japan’s Generation Network 2plus (JGN2plus) is an overlay and service platform for network control and measurement, designed to encourage advanced applications and international collaboration. It aims to foster research partnerships with industry, academia, government, and regional organizations. Within the US, the Global Environment for Network Innovation (GENI) project is the largest effort in this area.\(^1\) GENI has four control frameworks: ProtoGENI, PlanetLab, ORBIT, and the Open Resource Control Architecture (ORCA).\(^2\) This project centers on creating MSS Version 1.0 on the ORCA control framework (CF) as the first step in the following successive goals.

My first goal was to create a means by which the GENI ORCA CF users, such as developers, experimenters, and researchers, can share and track software services for use within ORCA. As a proof of concept, MSS Version 1.0 focuses on the “middle-mile” and “last-mile” of service delivery. My second goal is to create an infrastructure on which all GENI CFs can share services in exchange for resources. After successfully applying MSS to GENI, my ultimate goal is to expand this concept so that other LSNs may take advantage of the mutualistic model by offering valuable services, with computer security as the flagship service, in exchange for resources.

The concept of sharing services in exchange for resources is hardly new. For example, Amazon and the Apple App Store have a centralized delivery system in which customers exchange resources (money) for services (applications and products). In addition, peer-to-peer networks such as LimeWire and Kazaa use a decentralized delivery system in which members exchange resources (videos and music) for resources (other videos and music). MSS is different from both of these systems in several ways.

First, the planned architecture for MSS uses a hierarchical tree-like distribution, which flows centrally from the root at the GENI Maintenance Operations Center (GMOC), then branches through each CF origin to its affiliates, and finally flows to the users as leaves. The second difference relates to the first in the middle-mile and last-mile of service delivery. In the middle-mile, each node in the tree is a client of its parent server, and so a server is a client of a server, is a client of a server, and so on. This means that each server must authenticate with the originating CF, and each user must authenticate with his or her parent CF in order to receive services. In the last-mile, the required number of servers at each node decreases like the diameter of a tree branch from many servers at the origin to a CF residing on a single server (e.g., laptop and desktop computers) that a single individual can donate. This single server model upon which all ORCA and MSS systems must reside did not exist before this project. Finally yet importantly, GENI experimenters “pay” for GENI by hosting a CF and sharing resources in the form of

\(^1\) Quan, J., Nance, K., & Hay, B. (May/June 2011). A Mutualistic Security Service Model: Supporting Large-Scale Virtualized Environments. IT Professional (IEEE), 18-23

virtual machines, which means GENI’s basic unit of currency is the resource. If one considers the software products of these experiments to be services, then offering services to the public in exchange for donated resources provides the payment GENI requires to grow.

This first version of MSS incorporates a lightweight Linux, Apache, MySQL, and PHP (LAMP) infrastructure that ORCA owners can use to advertise and deliver services. In doing so, it addresses the fundamental, mutual need between owners and users: owners require vast amounts of resources to meet their goal of conducting at-scale Internet experiments, and users, experimenters, and researches require software services to conduct experiments and business. MSS leverages the hierarchical GENI structure to establish a distributed service delivery system. When a business or university donates a portion of its resources to GENI as an affiliate, it receives all or part of its sponsor’s services, which organizational members can then access.
Acknowledgements
I thank Dr. Brian Hay, Department of Computer Science, University of Alaska Fairbanks, Dr. Kara Nance, Department of Computer Science, University of Alaska Fairbanks, Dr. Jon Genetti, Chair of the Department of Computer Science, University of Alaska Fairbanks, and Dr. Peter Knoke, Department of Computer Science, University of Alaska Fairbanks for their guidance, technical knowledge, and support. Most importantly, I thank my wife Megan for her love and support, without which I could not have completed any of this.
Project Conventions
This project consists of the following volumes, in which each page has a header that reflects the volume, and which are numbered and designed to be self-contained documents:

I. Software Requirements Specification for Mutualistic Software Services (MSS)
II. Software Architecture for Mutualistic Software Services (MSS)
III. Software Design for Mutualistic Software Services (MSS)
IV. Software Testing for Mutualistic Software Services (MSS)
Table of Contents

Abstract.................................................................................................................................................. iii

Acknowledgements................................................................................................................................. v

Project Conventions............................................................................................................................... vi

Table of Contents.................................................................................................................................... vii

Challenges............................................................................................................................................... xiv

Summary and Conclusions...................................................................................................................... xvi

Volume I.................................................................................................................................................... 1

Software Requirements Specification for Mutualistic Software Services (MSS)............................... 2

1. Introduction........................................................................................................................................... 3

  1.1 MSS Envisioned................................................................................................................................. 3

  1.2 MSS at Present.................................................................................................................................. 4

  1.1 Purpose .............................................................................................................................................. 5

  1.2 Document Conventions .................................................................................................................... 5

  1.3 Intended Audience and Reading Suggestions .................................................................................. 5

  1.4 Project Scope and Product Features............................................................................................... 6

2. Overall Description............................................................................................................................... 7

  2.1 Product Perspective ............................................................................................................................ 7

  2.2 User Classes and Characteristics ..................................................................................................... 8

  2.3 Operating Environment ..................................................................................................................... 9

  2.4 Design and Implementation Constraints ....................................................................................... 11

  2.5 User Documentation .......................................................................................................................... 11

  2.6 Assumptions and Dependencies...................................................................................................... 12

3. System Features................................................................................................................................... 13

4. Functional Requirements..................................................................................................................... 17

5. External Interface Requirements......................................................................................................... 18

  5.1 User Interfaces .................................................................................................................................. 18

  5.2 Hardware Interfaces ......................................................................................................................... 18

  5.3 Software Interfaces ............................................................................................................................ 20

  5.4 Communications Interfaces ............................................................................................................. 21

6. Other Nonfunctional Requirements .................................................................................................. 23

~ vii ~
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Document Introduction</td>
<td>34</td>
</tr>
<tr>
<td>1.2</td>
<td>Business Decisions</td>
<td>34</td>
</tr>
<tr>
<td>1.3</td>
<td>System Purpose and Scope</td>
<td>35</td>
</tr>
<tr>
<td>1.4</td>
<td>Definitions, Acronyms, and Abbreviations</td>
<td>35</td>
</tr>
<tr>
<td>1.5</td>
<td>System Overview</td>
<td>35</td>
</tr>
<tr>
<td>2.1</td>
<td>System Description</td>
<td>37</td>
</tr>
<tr>
<td>2.2</td>
<td>Functional Attributes</td>
<td>37</td>
</tr>
<tr>
<td>2.3</td>
<td>Non-functional Attributes</td>
<td>37</td>
</tr>
<tr>
<td>2.4</td>
<td>System Constraints</td>
<td>37</td>
</tr>
<tr>
<td>2.5</td>
<td>User Characteristics</td>
<td>40</td>
</tr>
<tr>
<td>2.5.1</td>
<td>User Classes and Characteristics</td>
<td>40</td>
</tr>
<tr>
<td>2.5.2</td>
<td>User Groups and Attributes</td>
<td>40</td>
</tr>
<tr>
<td>2.6</td>
<td>Assumptions and Dependencies</td>
<td>41</td>
</tr>
<tr>
<td>2.7</td>
<td>Stakeholders</td>
<td>41</td>
</tr>
<tr>
<td>3.1</td>
<td>Capabilities</td>
<td>44</td>
</tr>
<tr>
<td>3.2</td>
<td>Conditions</td>
<td>47</td>
</tr>
<tr>
<td>6.1</td>
<td>Performance Requirements</td>
<td>23</td>
</tr>
<tr>
<td>6.2</td>
<td>Safety Requirements</td>
<td>23</td>
</tr>
<tr>
<td>6.3</td>
<td>Security Requirements</td>
<td>24</td>
</tr>
<tr>
<td>6.4</td>
<td>Quality Attributes</td>
<td>24</td>
</tr>
<tr>
<td>7.</td>
<td>Other Requirements</td>
<td>26</td>
</tr>
<tr>
<td>8.</td>
<td>References</td>
<td>27</td>
</tr>
<tr>
<td>Appendix A</td>
<td>Data Dictionary</td>
<td>29</td>
</tr>
</tbody>
</table>

Volume II: Software Architecture for Mutualistic Software Services (MSS) - 33
3.3 Constraints ........................................................................................................................................ 52
4. System Characteristics ......................................................................................................................... 55
  4.1 Autonomy ........................................................................................................................................ 55
  4.2 Integrability .................................................................................................................................... 55
  4.3 Extensibility ..................................................................................................................................... 55
  4.4 Portability ....................................................................................................................................... 55
  4.5 Usability .......................................................................................................................................... 55
  4.6 Securability ..................................................................................................................................... 56
  4.7 Credibility ....................................................................................................................................... 56
  4.8 Interoperability ................................................................................................................................. 56
5. Architectural Plan ................................................................................................................................. 57
  5.1 Remote Server Model ........................................................................................................................ 58
  5.2 Context Diagrams and Data Model .................................................................................................... 59
    5.2.1 MSS-CENTER .......................................................................................................................... 59
    5.2.2 MSS-ORIGIN .......................................................................................................................... 59
    5.2.3 MSS-AFFILIATE ...................................................................................................................... 59
    5.2.4 MSS-RESOURCE ...................................................................................................................... 59
    5.2.5 MSS-USER ............................................................................................................................. 59
    5.2.5 MSS-DEVELOPER .................................................................................................................. 59
  5.3 Component Model ............................................................................................................................. 60
    5.3.1 Databases ............................................................................................................................... 60
    5.3.2 Components ............................................................................................................................. 60
    5.3.3 Service Repository .................................................................................................................... 61
6. Architecture Trade-off Analysis Method (ATAM) ................................................................................ 62
  6.1 Purpose .......................................................................................................................................... 62
  6.2 Main Architectural Drivers ................................................................................................................. 62
  6.3 Business Goals ................................................................................................................................. 62
  6.4 Major Stakeholders ............................................................................................................................ 62
  6.5 Architectural Approaches .................................................................................................................. 63
  6.6 Utility Tree ....................................................................................................................................... 63
7. Cost Benefit Analysis Method (CBAM) ............................................................................................... 69
  7.1 Utility Response Curves ...................................................................................................................... 69
Initialization scripts ............................................................................................................................................... 210

Volume IV ......................................................................................................................................................... 212

Software Testing for Mutualistic Software Services (MSS) ............................................................................. 213

1. Introduction .................................................................................................................................................. 214

2. Test Criteria .................................................................................................................................................. 214

3. Tested Components ...................................................................................................................................... 215
   3.1 Databases .............................................................. ................................................................................. 215
   3.2 Components .............................................................. .......................................................................... 215
   3.3 Service Repository .............................................................. .................................................................. 230

4. Evaluation ..................................................................................................................................................... 234
Challenges
Mutualistic Software Services began in my mind’s eye as a simple solution to two seemingly disparate problems. After reading about a series of computer attacks on the United States in the first decade of the 21st century, I began to wonder how one might curb such incidents when it seems that many Americans are oblivious to the risks of surfing the Internet. During this time, I worked as a GENI research assistant at the University of Alaska Fairbanks (UAF) developing federation incentives to increase GENI membership. This is when the idea of offering a security service in exchange for computer resource donations from the public occurred to me. After realizing that no use case exists for this arrangement, I set about devising such an infrastructure. My abstract portrays my goals for this project as beginning with a small scope, which progressively grows until finally LSNs offer computer security, among other services, in exchange for resources. In reality, I pictured the “grand vision” first, and then shrunk the scope in steps until I finally developed a plausible beginning for MSS.

This project presented several challenges throughout its course. Some of these hurdles included:

- creating a CF cluster that resides on a single server
- developing a hierarchical service delivery system in which the client has no access to the sponsor’s file system and the end users only have the Secure Shell (SSH) port 22 open
- pushing database updates to client sponsors
- incorporating heartbeats into Version 1.0 with no CF support
- applying a recursive attribute tree that describes the services to the file system directory.

The single greatest challenge was developing the single-server CF cluster. A canonical ORCA installation resides on at least two, but typically several computers, and condensing all of the hardware onto one computer took me several months to accomplish. Fortunately, I was able to apply this to one of the UAF GENI goals, which is to set up a GENI CF at remote locations across Alaska. Currently, Barrow, Alaska hosts the first “remote server,” on which a working ORCA installation and MSS Version 1.0 harmoniously coexist on one computer. The remote server works by virtualizing the router ORCA prescribes for Network Address Translation (NAT), and I thank Dr. Brian Hay for applying his networking expertise to modify the ORCA method of NAT in a way better suited for a single-server application.

The second difficulty was to develop a hierarchical service delivery system. In MSS, the sponsor controls service delivery by using a proxy on each client. This is a user specifically designated to push MSS services using remote synchronization (Rsync) with SSH, and the proxy serves two purposes. Primarily, sponsors will deliver new services upon a client administrator’s request if the client computer is donating resources to GENI. Secondarily, this allows sponsors to apply updates to downloaded services without the client having to initiate the action, which later may provide an avenue for a security service to keep the client computer’s virus definitions up to date. One might infer that sponsors also can delete services from unruly client sponsors, but this is not the intent of MSS. Once a client downloads a service, it belongs to the client. Fortunately, I also was able to use remote SSH commands to solve the third challenge of pushing database updates to the client sponsors. MSS uses the same user proxy to send MySQL statements remotely by allowing the user proxy to access the ORCA, Users, and Services databases.

~ xiv ~
The fourth barrier I faced was incorporating heartbeats, which are packets of information sent from participating ORCA clusters to the ORCA Remote Actor Registry, and that contain identifying information flagging the resources as available to GENI experimenters. Since I do not have access to the ORCA Actor Registry database, I had to develop an alternative means by which I use the Registry’s Hypertext Transfer Protocol (HTTP) source code to determine whether an ORCA actor is currently donating resources.

Lastly, I required an intuitive service storage system on the sponsor and client computers so that end-users easily can find their services and that would not add overhead to MSS. My first inclination was to use MySQL Binary Large Object (BLOB) storage and deliver the service to the user with the Secure File Transfer Protocol, but this would have added maintenance and upkeep in the Services database when a perfectly good file system already exists on the computer. However, the file system still requires a method to translate the attributes of the services to the file location on the computer. Coincidentally, my friend and former co-worker Donald Kline and I wrote a paper about his brainchild, the *Attribute Description Service for Large-Scale Networks* (2011) to address this problem and others.³

I did not use our Attribute Description Service in Version 1.0 because the project scope would have been too great, but I did include it in the architecture for MSS Version 2.0. The main idea I applied to MSS Version 1.0 is if one can describe a service according to its attributes, one can describe a service location according to its attributes. For example, consider a fictitious new program named the Simple Border Gateway Protocol (sBGP) to be a networking protocol service within the GENI framework that one may download as the archived file sBGP.zip. One might consider the attribute tree for sBGP like so: MSS has attributes, of which one is Software...has attributes, of which one is Services...has attributes, of which one is Network...has attributes, of which one is Protocol...has attributes, of which one is the service sBGP. Therefore, MSS stores the uniquely named file sBGP.zip in the same place on all computers: [MSS root location]/MSS/Software/Services/Network/Protocol/sBGP.zip. One need only apply the attributes one used to find the service sBGP to retrieve the full directory path to the file.

³ Kline, D., & Quan, J. (2011). *Attribute Description Service for Large-Scale Networks*. (M. Kurosu, Ed.) Lecture Notes in Computer Science, 6776 (Human Computer International Conference 2011, Human Centered Design), 519-528
Summary and Conclusions

In summary, the project successfully delivered a working system to mutually exchange services for resources to authorized users on authorized computers. MSS guarantees this mutualistic relationship because the sponsor only delivers services upon request to users when the sponsor is donating resources. This is true for every sponsor in the ORCA hierarchy. Moreover, the Software Design for Mutualistic Software Services Version 1.0 (Volume IV of this document) provides the configuration changes necessary for one to set up a single computer from which one can donate a portion of its resources to the ORCA CF in exchange for services.

MSS meets its Version 1.0 objective to create a means by which ORCA CF users, such as developers, experimenters, and researchers, can share and track software services for use within ORCA. Furthermore, it lays a solid foundation to refine this process in Version 2.0 because it provides a viable option for advertising and delivering services to current ORCA donors. As experimenters and developers create more and more services, more and more people will want to donate their resources to gain services.
Software Requirements Specification for Mutualistic Software Services (MSS) Version 1.0
1. Introduction

According to *GENI At A Glance* (June 1, 2011), the National Science Foundation sponsors GENI, which is a large-scale network based on existing infrastructure that approximately 83 academic/industrial teams and 19 corporations are developing across the United States [1]. Currently, these same entities also contribute the vast majority of experimental resources to GENI. However, the GENI Systems Overview explains, “Including real-world users and traffic in GENI is key to providing the fidelity experimenters need in the GENI suite of infrastructures to make their experimental results potentially relevant to real-world networks.” It further states that GENI goals are to:

1) Provide the flexibility for researchers to experiment on programmable components.
2) Include a wide range of technologies, to include wireless, and to incorporate new technologies as they emerge.
3) Permit experiments that act as one expects to see in the real world
4) Strongly support measurement based research.
5) Remove practical research barriers.
6) Support multiple experiments on a shared infrastructure suite.
7) Support a strong isolation of slices, to which donated resources will belong.
8) Ensure a broad array of contributors can donate resources easily.
9) Provide a secure environment safe from subversion.
10) Designed for a 15-20 year lifetime [2].

1.1 MSS Envisioned

In the future, Mutualistic Software Services (MSS) will leverage the GENI need for experimental resources with the public need for useful software services. A natural benefit of GENI experimentation is that new services arise from this effort, and so delivering these services to donors in exchange for resources forms a mutualistic relationship between the two. Unlike the current system where experimenters share their resources within GENI with little public donation, this product will provide valuable services in exchange for a small portion of the subscriber's CPU cycles, hard disk space, RAM, and Internet bandwidth. MSS is available to anyone who donates a slice of his or her networked resource to GENI.

To meet this objective, MSS will require a central office to oversee the exchange of resources for services. The MSS Center will develop and maintain MSS standards, but its final incarnation may be as part of another GENI department, like the GENI Meta-Operations Center (GMOC). The Center will collect the services from developers and advertise them on a web site, though the developers will maintain the web pages for the service. In addition, the Center will use the Attribute Description Service [3] to allow service developers to describe all aspects of their service, such as its purpose, scope, uses, versions, revisions, rescissions, additions, and other such information as deemed appropriate by the MSS Center. Donors may then query MSS to choose which services are most useful to their organization and download the service. Moreover, donors may download as many services as they desire as long as they are donating to GENI.
Currently, GENI’s Open Resource Control Architecture (ORCA) control framework (CF) already has a "heartbeat" mechanism in place to ensure donors are donating and resources are available for experimentation. These heartbeats can carry relevant data, such as how many users are using which services, how many resources and of which type are available for experimentation, and other useful data that the MSS Center deems appropriate. Alternatively, the CF developer may choose to separate usage statistics from heartbeats to decrease collection intervals. In either case, MSS Center will collect statistics for each service to determine whether a service should receive continued support.

Furthermore, the MSS Center will act as the root of a hierarchy that delivers a subset of MSS services to the four GENI CFs: PlanetLab, ProtoGENI, ORBIT, and ORCA. For example, MSS Center will maintain every service, but the ORCA administrator at Duke University will only download services geared towards ORCA users. The University of Alaska Fairbanks (UAF) ORCA administrator then will download services as an affiliate of Duke, but may choose a smaller subset of services that fit UAF users. UAF then may sponsor a single server in Barrow, Alaska, and the Barrow affiliate then may sponsor an even smaller subset of services that are unique to Barrow users. As the number of offered services grows, so do the resource donations, to the mutual benefit of GENI and the resource donors.

1.2 MSS at Present
This SRS identifies the requirements to create MSS Version 1.0, which must exhibit key features of the MSS vision above in order to demonstrate success. Some of these features include:

- A hierarchical services delivery, in which the affiliate may choose all or a subset of sponsored services.
- A heartbeat check at the CF level to ensure the affiliate is donating resources to its sponsor before one can download new services.
- A means to install all CF and MSS software on a single computer to allow individual public participants.
- A process to ensure only authorized users on authorized computers can receive services.
- A method for sponsors to deliver MSS services, service data, and user data to its affiliates.
- A way for sponsors to host virtual and physical clients.

Furthermore, certain elements of the future vision are not necessary in the initial offering of MSS, but are identified in this document as Version 2.0 requirements. These features are:

- A means to gauge the percentage of resources an affiliate or client is donating.
- The inclusion of the GMOC at the root of MSS.
- The Attribute Description Service.
- A heartbeat check at the virtual and physical client level of service delivery.
- Usage statistics delivery to the CF developer.

In sum, MSS Version 1.0 administrators must be able to host and deliver services to affiliates, physical resources, and virtual resources from a canonical CF cluster and from a single-server CF cluster as a proof of concept.
1.1 Purpose
This SRS describes the software functional and nonfunctional requirements for version 1.0 of MSS. The project team will implement and verify the correct functioning of the system with this document. Unless otherwise noted, all requirements specified here are high priorities and committed for release 1.0.

1.2 Document Conventions
For ease of use, rules tables within this document begin on new pages so that project team members may divide it among the teams. The software team must address all Version 1.0 rules within this document before stakeholders will consider this project complete. Appendix A contains a Data Dictionary of terms.

1.3 Intended Audience and Reading Suggestions
The audience for this document consists of all favored User Classes listed in the 2.2 User Classes and Characteristics, and it follows the conventions from Software Requirements (2003, p. 172) [4]:

1 Introduction
  1.1 Purpose
  1.2 Document conventions
  1.3 Intended Audience and Reading Suggestions
  1.4 Project Scope
  1.5 References
2 Overall Description
  2.1 Product Perspective
  2.2 User Classes and Characteristics
  2.3 Operating Environment
  2.4 Design and Implementation Constraints
  2.5 User Documentation
  2.6 Assumptions and Dependencies
3 System Features
4 Functional Requirements
5 External Interface Requirements
  5.1 User Interfaces
  5.2 Hardware Interfaces
  5.3 Software Interfaces
  5.4 Communications Interfaces
6 Other Nonfunctional Requirements
  6.1 Performance Requirements
  6.2 Safety Requirements
  6.3 Security Requirements
  6.4 Software Quality Attributes
7 Other Requirements
8 References
1.4 Project Scope and Product Features
This SRS identifies the requirements to create MSS Version 1.0, which must exhibit key features of the MSS vision above in order to demonstrate success. Some of these features include:

- A hierarchical services delivery, in which the affiliate may choose all or a subset of sponsored services.
- A heartbeat check at the CF level to ensure the affiliate is donating resources to its sponsor before one can download new services.
- A means to install all CF and MSS software on a single computer to allow individual public participants.
- A process to ensure only authorized users on authorized computers can receive services.
- Other processes to:
  - Authenticate users.
  - Validate services.
  - Encrypt and Decrypt services during service delivery.
- A method for sponsors to deliver MSS services, service data, and user data to its affiliates.
- A way for sponsors to host virtual and physical clients.

Furthermore, certain elements of the future vision are not necessary to demonstrate the functionality of MSS, and they are identified in this document as Version 2.0 requirements. These features are:

- A means to gauge the percentage of resources an affiliate or client is donating.
- The inclusion of the GMOC at the root of MSS.
- The Attribute Description Service.
- A heartbeat check at the virtual and physical client level of service delivery.
- Usage statistics delivery to the CF origin.
- Secure Socket Layer encryption of the website.

In sum, MSS Version 1.0 administrators must be able to host and deliver services to affiliates, physical resources, and virtual resources from a canonical CF cluster and from a single-server CF cluster as a proof of concept.

MSS will permit GENI contributors to receive GENI developed software as services for donating a portion of their computer resources. Project team members must familiarize themselves with the following in order to conceptualize MSS, and it is recommended that other stakeholders familiarize themselves with this material.

MSS leverages four prevailing technologies to accomplish its goal:

- Open Resource Control Architecture (ORCA), a GENI CF [5]
- Eucalyptus (Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems) [6]
- Xen [7] and Kernel-based Virtual Machines (KVM) [8]
• Linux, Apache, MySQL, and PHP (LAMP) Server [9]

As a use case example, one can find two proposed scenarios that explore the implementation of these features in:

• A Mutualistic Security Service Model: Supporting Large-Scale Virtualized Environments (2011) [10]
• Attribute Description Service for Large-Scale Networks (2011) [3]

2. Overall Description

2.1 Product Perspective
MSS is a new system that leverages the need for public computer services and the GENI need for contributed resources. The system will evolve over subsequent releases to offer automatic lookup queries based on the initiating resource type and other criteria listed by the resource owner.

It is important to note the difference between the resource owner and the resource user. For example, a resource owner may be the owner of an organization or a delegated authority, such as a systems administrator. The resource owner decides which services are appropriate for his or her organizational needs, acquires the desired services from his or her parent CF, and advertises those services to affiliate CFs and resource users within the organization. Resource owners may pare these services further based on mission needs and user groups, such as installing an affiliate CF that only hosts accounting services for accountants or clerical services for information specialists.

This arrangement forms a hierarchy of service delivery, in which the company provides a small amount of resources in exchange for valuable services. MSS Version 1.0 will meet all of the requirements identified as Version 1.0 in this document, with design considerations geared toward meeting Version 2.0 requirements. Research findings of exactly how many experimenters GENI supports has proved elusive; however Version 2.0 support requirements are based on figures from GENI’s largest CF, Planetlab, which Princeton University sponsors. Princeton’s Policy Report & Analysis document, Understanding and Resolving Conflicts (November 13, 2008), lists 4,700 researchers [21], and so 4,700 multiplied by four CFs and doubled for expected growth equals 37,600 MSS users.
### 2.2 User Classes and Characteristics

<table>
<thead>
<tr>
<th>User Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource Donors</td>
<td>These are members of the public who may find GENI services to be useful. Public donors view GENI Services as practically free because experimenters only use a portion of the donor’s “unused” resources. However, one might not view bandwidth, RAM, and CPU cycles as unused because they can range from 0-100% use very quickly, and so some means of regulating experimenter sharing of these resources during “slow” times will be considered by the GENI CFs.</td>
</tr>
<tr>
<td>Resource Users</td>
<td>Resource users do not necessarily own the resource, but use the services on behalf of the resource owner. For instance, if a small business owner donated 20% of her computer resources, then her administrative assistant, as a registered resource user, might have access to a customer billing service.</td>
</tr>
<tr>
<td>GMOC</td>
<td>The GENI Meta Operations Center, based out of the University of Indiana, is the central authority for CF interconnections [11]</td>
</tr>
<tr>
<td>CFs (favored)</td>
<td>Each CF will download the MSS services and updates, and it will deliver them to their donors. This greatly simplifies the process of deciding who should receive MSS because if a member of the public does not connect to a CF, he or she cannot receive the service. This way the parent CF only offers MSS to its own users and one or more child CFs instead of thousands of individual donors, which reduces overhead. Furthermore, the originating CF already tracks and connects to its donor resources through heartbeats, and so this arrangement adds minimal authentication overhead to the system, while enabling each originating CF to gauge the MSS impact on donor participation. Though individual CFs determine their own federation standards, donors must meet some minimum GENI federation standards before federating resources, and must maintain these standards while connected to the CF. Lastly, MSS must not burden the CF with additional work. Current CF administrators must be able to manage MSS, such as performing downloads and updates, and adding authorized actors (computers) and users as necessary to run MSS.</td>
</tr>
<tr>
<td>Developers (favored)</td>
<td>Service developers are also the service maintainers. Developers use MSS to share their services created during the course of research or to fill a particular need, and they advertise those services to GENI donors. Donors provide a portion of their resources in exchange for valuable services, and so the developers/researchers have more resources on which to experiment.</td>
</tr>
<tr>
<td>Experimenters</td>
<td>Experimenters require resources on which to experiment, and increasing experimental resources is the main reason for creating MSS. The more resources experimenters have, the more “at-scale” experiments are possible. Furthermore,</td>
</tr>
</tbody>
</table>
allowing the public to donate resources provides opportunities for “in the wild” experiments that the *GENI Systems Overview* (2008) identifies as crucial to GENI success [2].

| GPO (favored) | The GPO represents the NSF, which pays for MSS. Service developers must give GPO recommendations the utmost consideration when forming decisions. One key aspect of GPO concern is that donor interest and trust in MSS remain firm. Maintaining donor confidence will ensure project continuation by steadily growing CFs. |
| “Pay-for” Applications Companies (disfavored) | Initially, these companies may perceive MSS as a threat, and therefore exert political pressure on the NSF to end MSS both directly and indirectly. The project manager will notify the GPO of any correspondence with these groups. |

### 2.3 Operating Environment

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>OE-1</td>
<td>X</td>
<td>X</td>
<td>Hardware Platforms: MSS will operate on GENI clusters and single servers. MSS will operate on large server farms, small servers, personal computers, and tablet computers.</td>
</tr>
<tr>
<td>OE-2</td>
<td>X</td>
<td>X</td>
<td>Operating Systems: MSS is intended to support:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Linux</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Debian</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Slackware</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- RedHat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X</td>
<td>MSS will support other operating systems, such as:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Unix</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Macintosh OS X Leopard and above</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- FreeBSD</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- OpenSolaris</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X</td>
<td>- Windows (in subsequent releases)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Windows 7</td>
</tr>
</tbody>
</table>

~ 9 ~
| OE-3 | X   | X   | Web Browsers: MSS is intended for major web browsers, such as:
|      |     |     | • Windows
|      |     |     |   o Internet Explorer
|      |     |     | • Mozilla
|      |     |     |   o Firefox
|      |     |     | • Konquerer
|      |     |     |   o Safari
|      |     |     |   o Google Chrome
|      | X   | X   | In addition, MSS must work on a text-based web browser in order to support virtual machines and computers without desktop environments:
|      |     |     | • w3m

| OE-4 | X   | X   | X   | Geographical Locations: Release 1.0 only supports US donors due to funding. Subsequent releases may include CFs from other countries, which may have their own MSS offerings.

| OE-5 | X   | X   | Users: US public donors to GENI CFs.

| OE-6 | X   | X   | Servers: Release 1.0 support will focus on Linux server platforms.

| OE-7 | X   | X   | Databases:
|      |     |     | MSS will maintain at least three separate databases for:
|      |     |     | • Service data
|      |     |     | • CF data
|      |     |     | • User and client data
|      |     |     | MSS will use the Attribute Description Service to track service, user, and client data.
2.4 Design and Implementation Constraints

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO-1</td>
<td>X</td>
<td></td>
<td>This system’s design, code, and maintenance documentation shall conform to the W3C Quality Assurance Interest Group specifications [13].</td>
</tr>
<tr>
<td>CO-2</td>
<td>X</td>
<td>X</td>
<td>Web design will be similar to the CF style.</td>
</tr>
<tr>
<td>CO-3</td>
<td>X</td>
<td>X</td>
<td>OS Versions: Design MSS only for supported versions of operating systems.</td>
</tr>
<tr>
<td>CO-4</td>
<td>X</td>
<td>X</td>
<td>Web Browser Versions: Design MSS only for supported versions of web browsers.</td>
</tr>
</tbody>
</table>

2.5 User Documentation

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>UD-1</td>
<td>X</td>
<td></td>
<td>The services shall provide an online, cross-linked help system in HTML that describes all service functions.</td>
</tr>
<tr>
<td>UD-2</td>
<td>X</td>
<td></td>
<td>A “Quick Start” document shall provide screenshots of applicable windows and how the user shall interact with them for each major operating system. The ORCA User Manual (2011) serves as an example [12].</td>
</tr>
<tr>
<td>UD-3</td>
<td>X</td>
<td>X</td>
<td>The service shall provide the ability to download the user manual in Portable Document Format (PDF).</td>
</tr>
<tr>
<td>UD-4</td>
<td>X</td>
<td></td>
<td>The system shall provide CF operators with online documentation on how to download MSS additions.</td>
</tr>
<tr>
<td>UD-5</td>
<td>X</td>
<td></td>
<td>The system shall provide CF operators with online documentation on how to download MSS revisions.</td>
</tr>
<tr>
<td>UD-6</td>
<td>X</td>
<td></td>
<td>The system shall provide CF operators with online documentation on how to process MSS rescissions.</td>
</tr>
<tr>
<td>UD-7</td>
<td>X</td>
<td>X</td>
<td>The system shall provide CF operators with online documentation on how to maintain MSS versions.</td>
</tr>
</tbody>
</table>
2.6 Assumptions and Dependencies

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS-1</td>
<td>X</td>
<td>X</td>
<td>CFs are connected in a hierarchical tree</td>
</tr>
<tr>
<td>AS-2</td>
<td>X</td>
<td>X</td>
<td>The child CF can synchronize MSS services with its parent CF.</td>
</tr>
<tr>
<td>AS-3</td>
<td>X</td>
<td>X</td>
<td>Parent CFs have the capacity to distribute services to its child CF and donors</td>
</tr>
<tr>
<td>AS-4</td>
<td>X</td>
<td>X</td>
<td>All CFs will send “heartbeats” to its originating CF</td>
</tr>
<tr>
<td>DE-4</td>
<td></td>
<td>X</td>
<td>The system will provide CF operators with online documentation on how to download MSS revisions.</td>
</tr>
<tr>
<td>DE-5</td>
<td></td>
<td>X</td>
<td>The system will provide CF operators with online documentation on how to process MSS rescissions.</td>
</tr>
<tr>
<td>DE-6</td>
<td></td>
<td>X</td>
<td>The system will provide CF operators with online documentation on how to maintain MSS versions.</td>
</tr>
</tbody>
</table>
### 3. System Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Requirements</th>
<th>Version 1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-1</td>
<td>CF driven federation standards</td>
<td>DESCRIPTION</td>
<td>Each CF decides whether donors meet federation requirements, instead of MSS acting as a central authority for making this determination. This allows for greater autonomy among CFs and LSNs because each can have stringent standards that target certain donor types or relaxed standards that accept a broad variety of resources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRIORITY</td>
<td>HIGH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STIMULUS</td>
<td>Individual donors choose a CF to which he or she contributes resources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESPONSE</td>
<td>That CF decides whether the donor meets its federation standards, offers steps to meet the standards, or recommends another CF.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FUNCTIONAL REQUIREMENTS</td>
<td>TBD by each CF.</td>
</tr>
<tr>
<td>FE-2</td>
<td>CF controlled service</td>
<td>DESCRIPTION</td>
<td>Each CF determines whether the donor meets its standards and receives MSS. This may be as simple as, “while you are connected to this CF, you receive this service,” or the CF may limit the service, such as “if you do not maintain a connection 24 hours a day, you do not receive the service.” CFs already have a heartbeat mechanism in place to determine connection times.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRIORITY</td>
<td>HIGH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STIMULUS</td>
<td>An individual donor chooses a CF to which he or she contributes resources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESPONSE</td>
<td>That CF decides whether the donor meets its federation standards, offers steps to meet the standards, or recommends another CF.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FUNCTIONAL REQUIREMENTS</td>
<td>TBD by each CF.</td>
</tr>
<tr>
<td>FE-3</td>
<td>Software Updates</td>
<td>DESCRIPTION</td>
<td>MSS delivers MSS software updates through the MSS interface. Hierarchical delivery conserves bandwidth and streamlines service decisions (FE-1, 2)</td>
</tr>
<tr>
<td>-------</td>
<td>------------------</td>
<td>-------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRIORITY</td>
<td>HIGH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STIMULUS</td>
<td>MSS finishes final testing of its next software update.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESPONSE</td>
<td>MSS delivers the system update to participating CFs in GENI by the end of the next business day.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FUNCTIONAL REQUIREMENTS</td>
<td>MSS and the participating CF:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o Must be connected to the Internet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o Must authenticate before transmission.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o Must use data encryption.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o Must verify valid receipt.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FE-4</th>
<th>Service Updates</th>
<th>DESCRIPTION</th>
<th>MSS delivers all GENI services through the MSS interface. Hierarchical delivery conserves bandwidth and streamlines service decisions (FE-1, 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PRIORITY</td>
<td>HIGH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STIMULUS</td>
<td>MSS finishes final testing of its service or service update.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESPONSE</td>
<td>MSS delivers the service or update to participating CFs in GENI by the end of the next business day.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FUNCTIONAL REQUIREMENTS</td>
<td>MSS and the participating CF:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o Must be connected to the Internet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o Must authenticate before transmission.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o Must use data encryption.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o Must verify valid receipt.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FE-5</th>
<th>Donor Web Interface</th>
<th>DESCRIPTION</th>
<th>MSS queries will be conducted over a web interface using an appropriate authentication mechanism.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PRIORITY</td>
<td>HIGH</td>
</tr>
</tbody>
</table>

~ 14 ~
STIMULUS  Donor initiates a query for a service based on his or her parent CF service list.

RESPONSE  MSS responds with a list of services available to the donor.

FUNCTIONAL REQUIREMENTS  The web interface must support the major browsers listed in 2.3 Operating Environment.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Requirements</th>
<th>Version 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-6</td>
<td>Service Management interface</td>
<td>DESCRIPTION  ADS will provide “cradle-to-grave” service management by tracking service revisions, rescissions, dependencies, and other life-cycle data that an appropriate authority can add, edit, and delete.</td>
<td>HIGH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRIORITY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>STIMULUS</td>
<td>Manager adds a new service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESPONSE</td>
<td>The interface responds whether the new entry is complete.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FUNCTIONAL REQUIREMENTS</td>
<td>The web interface must support the major browsers listed in 2.3 Operating Environment.</td>
</tr>
<tr>
<td>FE-7</td>
<td>GENI Research Information</td>
<td>DESCRIPTION  GENI provides this venue for GENI experimenters to post service “tips and tricks,” advise donors and other experimenters, post editorials and current research information, and any other information that GENI sees fit to post. It will be in a wiki format. The GUI and Online Users Manual will have links to this wiki.</td>
<td>LOW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRIORITY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>STIMULUS</td>
<td>Donor clicks GENI Research Information Wiki link</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESPONSE</td>
<td>GENI Research Information Wiki link comes up.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FUNCTIONAL REQUIREMENTS</td>
<td>GENI provides server space for MSS wiki. The MSS wiki shall use an appropriate database.</td>
</tr>
<tr>
<td>FE-8</td>
<td>GENI Notices</td>
<td>DESCRIPTION</td>
<td>GENI provides notices of upcoming events, information on GENI experiments, how contributions help GENI develop network innovation, and any other information that GENI sees fit to post.</td>
</tr>
<tr>
<td>-----</td>
<td>--------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRIORITY</td>
<td>LOW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STIMULUS</td>
<td>Donor clicks GENI Notices link</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESPONSE</td>
<td>GENI Notices link comes up.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FUNCTIONAL</td>
<td>GENI provides server space for GENI Notices. This may link to a web site GENI already owns.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REQUIREMENTS</td>
<td></td>
</tr>
</tbody>
</table>
## 4. Functional Requirements

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR-1</td>
<td>X</td>
<td>X</td>
<td>Resource Owner User data must be safeguarded.</td>
</tr>
<tr>
<td>FR-2</td>
<td>X</td>
<td>X</td>
<td>MSS will use industry approved encryption standards, such as Digital Signature Algorithm (DSA) encryption.</td>
</tr>
<tr>
<td>FR-3</td>
<td>X</td>
<td>X</td>
<td>Only resource donors will have access to new GENI Services. MSS will rely on heartbeats sent from the resource to the originating CF to ensure this is so.</td>
</tr>
<tr>
<td>FR-4</td>
<td>X</td>
<td>X</td>
<td>ADS will track service usage</td>
</tr>
<tr>
<td>FR-5</td>
<td>X</td>
<td>X</td>
<td>Resource owners will have a means to control which authorized users can download services from its parent CF</td>
</tr>
<tr>
<td>FR-6</td>
<td>X</td>
<td>X</td>
<td>MSS-CENTER will support service mirroring</td>
</tr>
<tr>
<td>FR-7</td>
<td>X</td>
<td>X</td>
<td>MSS-PARENT will support service mirroring</td>
</tr>
</tbody>
</table>
5. External Interface Requirements

5.1 User Interfaces

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI-1</td>
<td>X</td>
<td>X</td>
<td>MSS screen displays shall conform to W3C formatting standards [13].</td>
</tr>
<tr>
<td>UI-2</td>
<td>X</td>
<td></td>
<td>MSS will comply with W3C handicap accessibility standards [13].</td>
</tr>
<tr>
<td>UI-3</td>
<td>X</td>
<td>X</td>
<td>MSS shall permit complete navigation using the keyboard alone, in addition to using mouse and keyboard combinations.</td>
</tr>
<tr>
<td>UI-4</td>
<td>X</td>
<td>X</td>
<td>MSS shall comply with GENI CF style designs</td>
</tr>
<tr>
<td>UI-5</td>
<td>X</td>
<td></td>
<td>The system shall display a help link on each page that links to a help page. At a minimum, the help page will contain a link to:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- The user manual</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- The online manual</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- The wiki</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- The GENI Notification web page</td>
</tr>
<tr>
<td>UI-6</td>
<td>X</td>
<td>X</td>
<td>The web pages shall permit complete navigation using the keyboard alone, in addition to using mouse and keyboard combinations.</td>
</tr>
</tbody>
</table>

5.2 Hardware Interfaces

At a minimum, MSS-CENTER will maintain these components:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>HI-1</td>
<td>X</td>
<td></td>
<td>MSS-CENTER will support web site mirroring.</td>
</tr>
<tr>
<td>HI-2</td>
<td>X</td>
<td></td>
<td>A web server will host all of the web pages necessary for finding service information. The web pages will include the Home page, About, News, Wiki, Events, Member, Links, and Help with appropriate information included in each. The Links page will contain hyperlinks to helpful service information, with a GENI link prominently displayed.</td>
</tr>
<tr>
<td>HI-3</td>
<td>X</td>
<td></td>
<td>A DNS server will act as the authoritative name server for MSS. Authoritative name servers are assigned to be responsible for their particular domains, and in turn can assign other authoritative name servers for their sub-domains. It will also implement the recursive algorithm necessary to resolve a given name starting with the DNS root through to the authoritative name servers of the queried domain. With this function...</td>
</tr>
</tbody>
</table>
implemented in the name server, user applications gain efficiency in design and operation [14].

**HI-4**

X

A communications server shall be an enterprise real-time communications server, providing the infrastructure for enterprise instant messaging, presence, file transfer, peer-to-peer and multiparty Voice and Video calling, ad hoc and structured conferences (audio, video and web) and PSTN connectivity. These features are available within an organization, between organizations, and with external users on the public internet, or standard phones, on the PSTN as well as SIP trunking [15].

**HI-5**

X X

A database server will contain the information about MSS services and Resource Owners/Users data.

**HI-6**

X

A systems administration server will utilize Simple Network Management Protocol (SNMP) and Cacti, among other tools. SNMP is a UDP-based network protocol. It is used mostly in network management systems to monitor network-attached devices for conditions that warrant administrative attention [16]. Cacti is a complete network graphing solution [under the GNU General Public License] designed to harness the power of RRDTool's data storage and graphing functionality. Cacti provides a fast poller, advanced graph templating, multiple data acquisition methods, and user management features out of the box. All of this is wrapped in an intuitive, easy to use interface that makes sense for LAN-sized installations up to complex networks with hundreds of devices [17].

**HI-7**

X

A load balancer will distribute workload evenly across two or more computers, network links, CPUs, hard drives, or other resources, in order to get optimal resource utilization, maximize throughput, minimize response time, and avoid overload. Using multiple components with load balancing, instead of a single component, may increase reliability through redundancy. The load balancing service is usually provided by a dedicated program or hardware device (such as a multilayer switch or a DNS server) [18].

At a minimum, MSS-ORIGIN/AFFILIATES will maintain these components:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Interface Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HI-8</td>
<td>X</td>
<td>Single-server components will connect behind a firewall</td>
</tr>
<tr>
<td>HI-9</td>
<td>X</td>
<td>Constant connection to the Internet through a Cable Modem, Direct Service Line, or other network connection</td>
</tr>
<tr>
<td>HI-10</td>
<td>X</td>
<td>A computer resource, provisioned according to a GENI CF standard.</td>
</tr>
</tbody>
</table>
## 5.3 Software Interfaces

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>SI-1</td>
<td></td>
<td><strong>MSS-ENTITY (CENTER, ORIGIN, AFFILIATE)</strong></td>
</tr>
<tr>
<td>SI-1.1</td>
<td>X</td>
<td>MSS-ENTITY shall maintain MSS in accordance with MSS-CENTER standards.</td>
</tr>
<tr>
<td>SI-1.2</td>
<td>X X</td>
<td>MSS-ENTITY shall connect through an encrypted means</td>
</tr>
<tr>
<td>SI-1.3</td>
<td>X X</td>
<td>MSS-ENTITY shall deliver services to its children.</td>
</tr>
<tr>
<td>SI-1.4</td>
<td>X X</td>
<td>MSS-ENTITY shall deliver MSS information to its children.</td>
</tr>
<tr>
<td>SI-1.5</td>
<td>X X</td>
<td>MSS-ENTITY shall maintain a hash, such as Secure Hash Algorithm (SHA), for all services it maintains as proof of the correct download.</td>
</tr>
<tr>
<td>SI-1.6</td>
<td>X X</td>
<td>MSS-ENTITY shall maintain a subset of service distributions for itself and its children according to MSS-CENTER standards.</td>
</tr>
<tr>
<td>SI-1.7</td>
<td>X</td>
<td>MSS-ENTITY shall maintain current ADS management information, such as additions, revisions, and rescissions for all of its services.</td>
</tr>
<tr>
<td>SI-1.8</td>
<td>X</td>
<td>MSS-ENTITY shall send heartbeats to its MSS-ORIGIN.</td>
</tr>
<tr>
<td>SI-1.9</td>
<td>X</td>
<td>MSS-ENTITY shall send registered user information to its parent</td>
</tr>
<tr>
<td>SI-1.9.1</td>
<td>X</td>
<td>MSS user information will only contain what MSS-CENTER requires to determine whether software should receive continued support, be revised, rescinded, and other similar data.</td>
</tr>
<tr>
<td>SI-1.10</td>
<td>X</td>
<td>MSS-ENTITY shall be chained through its parent to MSS-CENTER in normal operation, excepting short, unplanned periods.</td>
</tr>
<tr>
<td>SI-1.11</td>
<td>X</td>
<td>MSS-ENTITY shall maintain its connection to its children in normal operation, excepting short, unplanned periods.</td>
</tr>
<tr>
<td>SI-1.12</td>
<td>X</td>
<td>MSS-ENTITY shall transmit service download statistics for itself and its children to its MSS-ORIGIN.</td>
</tr>
<tr>
<td>SI-2</td>
<td></td>
<td><strong>MSS-CENTER</strong></td>
</tr>
<tr>
<td>SI-2.1</td>
<td>X</td>
<td>MSS-CENTER shall approve ADS definitions.</td>
</tr>
<tr>
<td>SI-2.2</td>
<td>X</td>
<td>MSS-CENTER shall approve new and existing services</td>
</tr>
<tr>
<td>SI-2.3</td>
<td>X</td>
<td>MSS-CENTER shall maintain all MSS services.</td>
</tr>
</tbody>
</table>
SI-2.4 | X | MSS-CENTER shall provide a web site for service developers to deliver service information to users (HI-2)

SI-2.5 | X | The developer’s web site shall have a means to log in securely and prevent tampering with its web pages.

SI-3 | MSS-ORIGIN

SI-3.1 | X X | MSS-ORIGIN shall maintain a current GENI CF.

SI-4 | MSS-AFFILIATE

SI-4.1 | X X | MSS-AFFILIATE shall maintain a current GENI CF.

SI-5 | MSS-RESOURCE

SI-5.1 | X X | MSS-RESOURCE shall be authorized by its parent.

SI-6 | MSS-USER

SI-6.1 | X X | MSS-USER shall register under his or her parent CF.

SI-7 | MSS-DEVELOPER

SI-7.1 | X | MSS-DEVELOPER shall register as a MSS developer under his or her parent CF.

SI-7.2 | X | MSS-DEVELOPER shall submit service information to the developer’s website according to MSS-CENTER standards.

SI-7.3 | X | MSS-DEVELOPER shall submit a service to MSS according to MSS-CENTER standards.

5.4 Communications Interfaces

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>CI-1</td>
<td>X</td>
<td></td>
<td>MSS employees shall use an email application that uses these protocols at a minimum:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• IMAP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• POP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMTP</td>
</tr>
<tr>
<td>CI-2</td>
<td>X</td>
<td></td>
<td>MSS-CENTER shall have an HTTP secure method for customers to contact a central mailbox.</td>
</tr>
<tr>
<td>CI-3</td>
<td>X X</td>
<td></td>
<td>MSS-ENTITY shall transfer files using at least DSA encryption</td>
</tr>
<tr>
<td>CI-4</td>
<td>X</td>
<td>MSS-CENTER shall set up a secure FTP box for any other customer transfers.</td>
<td></td>
</tr>
</tbody>
</table>
6. Other Nonfunctional Requirements

6.1 Performance Requirements

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>PE-1</td>
<td>X</td>
<td></td>
<td>MSS-CENTER will support site mirroring and return to service within 1 minute of a hard disk failure. (HI-1)</td>
</tr>
<tr>
<td>PE-2</td>
<td>X</td>
<td></td>
<td>A web server will act as the primary interface between the public and the service, and it will host all of the web pages necessary for MSS-CENTER interaction. Release 2.0 will support 37,600 visitors. (HI-2)</td>
</tr>
<tr>
<td>PE-3</td>
<td>X</td>
<td></td>
<td>A DNS server will act as the authoritative name server for MSS. Release 2.0 will support 37,600 visitors. (HI-3)</td>
</tr>
<tr>
<td>PE-4</td>
<td>X</td>
<td></td>
<td>A communications server shall be an enterprise real-time communications server. Release 2.0 will support 37,600 visitors. (HI-4)</td>
</tr>
<tr>
<td>PE-5</td>
<td>X</td>
<td></td>
<td>A database server will contain the ADS service information loaded into MSS-CENTER. Release 2.0 will support 37,600 visitors. (HI-5)</td>
</tr>
<tr>
<td>PE-6</td>
<td>X</td>
<td></td>
<td>A systems administration server is used mostly in network management systems to monitor network-attached devices for conditions that warrant administrative attention. Release 2.0 will support 37,600 visitors. (HI-6)</td>
</tr>
<tr>
<td>PE-7</td>
<td>X</td>
<td></td>
<td>A load balancer will distribute workload evenly across MSS-CENTER. Release 2.0 will support 37,600 visitors. (HI-7)</td>
</tr>
<tr>
<td>PE-8</td>
<td>X</td>
<td></td>
<td>Routers will interconnect all of these components and provide a connection to the internet. Release 2.0 will support 37,600 visitors. (HI-8)</td>
</tr>
</tbody>
</table>

6.2 Safety Requirements
None listed
6.3 Security Requirements
List the security requirements not covered elsewhere within this document.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSS-USER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE-1</td>
<td>X</td>
<td>Must register with a password</td>
</tr>
<tr>
<td>SE-2</td>
<td></td>
<td>Read the End User License Agreement and electronically sign his or her</td>
</tr>
<tr>
<td></td>
<td></td>
<td>understanding with an “I agree” button before using the service.</td>
</tr>
<tr>
<td>SE-3</td>
<td></td>
<td>Use CAPTCHA when emailing MSS-CENTER over HTTP</td>
</tr>
<tr>
<td>MSS-CENTER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE-4</td>
<td>X</td>
<td>Will anonymize subscriber information to the greatest extent possible to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>guard against personal information loss.</td>
</tr>
<tr>
<td>SE-5</td>
<td>X</td>
<td>Will use strong password protection, which will be enforced by the system</td>
</tr>
<tr>
<td>SE-6</td>
<td></td>
<td>Must use a physical security token to access workstations and servers.</td>
</tr>
<tr>
<td>SE-7</td>
<td></td>
<td>Passwords must be changed every 90 days</td>
</tr>
<tr>
<td>MSS-ORIGIN/AFFILIATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE-8</td>
<td>X</td>
<td>Will anonymize subscriber information to the greatest extent possible to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>guard against personal information loss.</td>
</tr>
<tr>
<td>SE-9</td>
<td></td>
<td>Will use strong password protection, which will be enforced by the system</td>
</tr>
</tbody>
</table>

6.4 Quality Attributes

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexibility</td>
<td></td>
<td>MSS should be flexible enough to work on several manufacturer-supported</td>
</tr>
<tr>
<td></td>
<td>X</td>
<td>operating systems such as Windows, Linux, and Macintosh.</td>
</tr>
<tr>
<td>FL-1</td>
<td>X</td>
<td>A maintenance programmer who has at least six months experience</td>
</tr>
<tr>
<td></td>
<td></td>
<td>programming a Windows supported operating system shall be able to update</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSS data and services within one hour.</td>
</tr>
<tr>
<td>FL-2</td>
<td></td>
<td>A maintenance programmer who has at least six months experience</td>
</tr>
<tr>
<td></td>
<td>X</td>
<td>programming a Linux supported operating system shall be able to update MSS</td>
</tr>
<tr>
<td>Requirement</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>FL-3</td>
<td>A maintenance programmer who has at least six months experience programming a Macintosh supported operating system shall be able to update MSS data and services within one hour.</td>
<td></td>
</tr>
<tr>
<td>Integrity</td>
<td>The MSS Chief Security Officer has the final say in what security authentication measures are implemented to ensure appropriate measures are taken. He or she should also keep a history of changes to the security authentication measures.</td>
<td></td>
</tr>
<tr>
<td>IN-1</td>
<td>The MSS CSO must approve any change to MSS security authentication protocols in writing.</td>
<td></td>
</tr>
<tr>
<td>IN-2</td>
<td>The MSS CSO must maintain records of changes to security authentication protocols for 10 years.</td>
<td></td>
</tr>
<tr>
<td>Reliability</td>
<td>MSS shall only be unavailable for 24 hours per month during peak hours, and 48 hours per month during non-peak hours.</td>
<td></td>
</tr>
<tr>
<td>RE-1</td>
<td>MSS shall be at least 96.77% available from 6:00 a.m. EST to 10:00 p.m. EST.</td>
<td></td>
</tr>
<tr>
<td>RE-2</td>
<td>MSS shall be at least 93.75% available from 10:00 p.m. EST to 6:00 a.m. EST.</td>
<td></td>
</tr>
</tbody>
</table>
7. Other Requirements
At least one GENI CF must work on a single server in order to gain public resource donations. A key requirement for MSS Version 1.0 is developing a systems design so that all software required by one CF and all MSS software works on a single server.
8. References


Appendix A

Data Dictionary

ADS – Attribute Description Service [3]

Aggregate – a collection of components that usually comprise a system

Bandwidth - a bit rate measure of available or consumed data communication resources expressed in Gigabits/second.

CF – Control Framework

Component – a physical computer resource, such as a router, switch, computer, phone, or copy machine

Community – MSS users identified by a GENI control framework

Contributor – an entity that donates a portion of its resources to GENI

Control framework – one of four GENI architectures used to federate computer resources

Donation – Donations include nearly all types of network resources, such as computers, network routers and switches, cell phones, computer tablets, copy machines, and so on. However, MSS is only available for computers as of this writing.

Experimenter – one who conducts network research on GENI

Federate – incorporate one’s computer resource into a GENI control framework

File list – hierarchical locations of files on a resource

GENI – Global Environment for Network Innovation

GUI – Graphical User Interface

Hard Disk – Non-volatile storage device for digital media

HTTP (Hypertext Transfer Protocol) – is not a protocol dedicated for email communications, but it can be used for accessing your mailbox. In addition, called web based email, this protocol can be used to compose or retrieve emails from your account. Hotmail is a good example of using HTTP as an email protocol [19].

IMAP (Internet Message Access Protocol) – Is a standard protocol for accessing e-mail from your local server. IMAP is a client/server protocol in which e-mail is received and held for you by your Internet server. As this requires only a small data transfer this works well even over a slow connection such as a modem. Only if you request to read a specific email message will it be downloaded from the server. You can also create and manipulate folders or mailboxes on the server, delete messages etc... [19]
Insertable media – a storage device that is not native to the resource, such as a USB drive, Compact Disc, and external hard drive

ISP – Internet Service Provider

Kbps – Kilobytes per second

Kernel - In computing, the kernel is the central component of most computer operating systems; it is a bridge between applications and the actual data processing done at the hardware level. The kernel's responsibilities include managing the system's resources (the communication between hardware and software components). Usually as a basic component of an operating system, a kernel can provide the lowest-level abstraction layer for the resources (especially processors and I/O devices) that application software must control to perform its function. It typically makes these facilities available to application processes through inter-process communication mechanisms and system calls [20].

LSN – Large-Scale Network. GENI is an example of a large-scale network.

LTS – refers to a Long-Term Service agreement provided by many operating system developers, such as Ubuntu.

Malware – includes computer viruses, computer worms, Trojan horses, most rootkits, spyware, dishonest adware and other malicious and unwanted software, including true viruses.

Mbps – Megabytes per second

MSS – Mutualistic Software Services

MSS-CENTER – refers to the main MSS office, where MSS is developed and distributed to participating control frameworks.

MSS-AFFILIATE – refers to all child nodes of the MSS-ORIGIN control framework. At a minimum, MSS-AFFILIATE will contain its MSS-ORIGIN control framework and a subset of its services.

MSS-DEVELOPER – refers to a member of academia or business who submits a service to MSS.

MSS-ENTITY – refers to MSS-CENTER, MSS-ORIGIN, and MSS-AFFILIATE. These are all of the MSS entities that have a control framework installed.

MSS-ORIGIN – refers the control framework developer and maintainer. MSS-ORIGIN falls directly beneath MSS-CENTER in the hierarchy.

MSS-RESOURCE – a resource without a control framework, such as a physical computer or virtual machine.

MSS-USER – refers to an authorized user of a MSS-RESOURCE due to his or her working relationship with a resource owner.
Node – a computer

NSF – National Science Foundation

OS – Operating System

POP (Post Office Protocol 3) – provides a simple, standardized way for users to access mailboxes and download messages to their computers. When using the POP protocol all your email messages will be downloaded from the mail server to your local computer. You can choose to leave copies of your emails on the server as well. The advantage is that once your messages are downloaded you can cut the internet connection and read your email at your leisure without incurring further communication costs. On the other hand you might have transferred a lot of message (including spam or viruses) in which you are not at all interested at this point [19].

Programmable component – a network resource with a modifiable operational instruction set

RAM – Random Access Memory. For MSS, this refers to the computer’s volatile memory, such as Dynamic RAM (DRAM).

Real-world user – a private citizen

Resource – a portion or an entire physical computer hardware component. MSS is only concerned with resources such as laptops, desktops, and computer tablets (such as iPad).

Resource Owner – one who is authorized to donate all or a portion of a resource to GENI

Slice – a collection of one or more aggregates and components

SMTP (Simple Mail Transfer Protocol) – is used by the Mail Transfer Agent (MTA) to deliver your email to the recipient's mail server. The SMTP protocol can only be used to send emails, not to receive them. Depending on your network / ISP settings, you may only be able to use the SMTP protocol under certain conditions (see incoming and outgoing mail servers [19].

SRS – Software Requirements Specification

Standard – the minimum hardware and software configuration required to federate into GENI

Traffic – the movement of information across the Internet

UI – User Interface

USB – universal serial bus

Wireless – radio-based Internet connection
Volume II
Software Architecture for Mutualistic Software Services (MSS) Version 1.0
1. Introduction

1.1 Document Introduction

The *Software Requirements Specification for Mutualistic Software Services Version 1.0* identifies Mutualistic Software Services (MSS) as a means for GENI to advertise and deliver services that experimenters develop in exchange for experimental resources. As Len Bass et al., recommend in *Software Architecture in Practice* (2003), this document intends to develop an architecture for the “…structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them” [1]. Therefore, this document will develop an architectural skeleton based on the functional requirements, non-functional requirements, and constraints identified in the SRS.

1.2 Business Decisions

Mutualistic Software Services (MSS) addresses several experimenter, researcher, developer, entrepreneur, and resource owner needs. Its most basic meaning derives from:

- Mutualistic – a win-win relationship in which both parties benefit.
- Software – the tools that those parties use to perform experiments, do research, sell products, run resources, and create services, to include the created services.
- Services – the virtual machine images, libraries, scripts, code, directions, and other useful products, to include the advertising of these products to stakeholders.

MSS has four different business models: trade, monetized, GENI-only, and a hybrid of the first three. All but the GENI-only model hinges on the idea that GENI requires many resources to create at-scale network experiments, and all models recognize that experimenters need a way to advertise and share experiments. In the trade model, GENI experimenters create services in the course of research that are unique and valuable, and public organizations and individuals can donate a portion of their resources in exchange for these worthy services. The monetized model is in keeping with the US Ignite program explained below, which relies on the notion that GENI experimenters will develop services that are so useful that people are willing to pay for them. The GENI-only model considers that GENI experimenters create software to create services, and that both of these products are useful to other experimenters. The hybrid model combines one or more of the first three models. However, no one will know about these products within any model unless one creates some advertising and sharing mechanisms.

The MSS architecture relies on distributed service delivery because this fits all of the models, if one considers a monetized system to have one parent with every other entity as its child. Therefore, the architectural perspective is that MSS will use the trade model because it is the most highly distributed of the three.
1.3 System Purpose and Scope

MSS leverages the GENI need for experimental resources with the public’s need for useful services, and serves to meet the goals of two National Science Foundation (NSF) sponsored programs. The Computer & Information Science & Engineering (CISE) branch of the NSF sponsors U.S. Ignite:

US Ignite is an initiative to spark the development of killer apps in areas of national priority: health, education, energy, economic development (including advanced manufacturing), transportation, and public safety on an ultra high speed (>100 Mbps up- and download), deeply programmable (not requiring internet protocol) and sliceable network. US Ignite is doing this by: 1) funding researchers and developers to create applications and services, and 2) stitching together an at-scale testbed with real users that researchers, developers, and entrepreneurs can use as a platform to develop applications and services [2].

The CISE also sponsors GENI, which is the “at scale testbed” on which U.S. Ignite will develop the “killer apps” of the future. The GENI Project Office describes GENI in this way:

GENI, a virtual laboratory for exploring future internets at scale, creates major opportunities to understand, innovate and transform global networks and their interactions with society. Dynamic and adaptive, GENI opens up new areas of research at the frontiers of network science and engineering, and increases the opportunity for significant socio-economic impact. GENI will:

- support at-scale experimentation on shared, heterogeneous, highly instrumented infrastructure;
- enable deep programmability throughout the network, promoting innovations in network science, security, technologies, services and applications; and
- provide collaborative and exploratory environments for academia, industry and the public to catalyze groundbreaking discoveries and innovation [3].

MSS serves as the conduit through which GENI experimenters gain resources for at-scale experiments and U.S. Ignite delivers services. Some examples of new services in development by GENI experimenters are uCap, FlowScale, OpenFlow, and MobilityFirst. All of these were highlighted in a recent GENI “News and Events” article [4], which is reproduced in its entirety in Appendix C.

1.4 Definitions, Acronyms, and Abbreviations

Please find definitions, acronyms, and abbreviations in Software Requirements Specification for Mutualistic Software Services Version 1.0.

- Appendix A: Data Dictionary

1.5 System Overview

MSS solves the problem of providing a vast number of real-world experimental resources to GENI experimenters by providing a means to “pay” for the resources with the services they create. This arrangement is mutually beneficial to resource donors, who must only provide a small portion of their resources in order to receive the services. MSS resides on one computer in a GENI cluster and distributes services rooted at MSS-CENTER to its subordinate CFs. MSS-ORIGINS deliver a subset of services to their MSS-AFFILIATES, and this distribution continues until services are distributed to all MSS-RESOURCES. This forms a hierarchical tree, with each parent CF acting as a node and its children as its
branches. MSS branches are capable of autonomy in that a MSS-ENTITY can continue using downloaded services even when its upstream connection to the MSS-CENTER is severed. However, the MSS-ENTITY must be sending heartbeats to its MSS-ORIGIN in order to receive new services.

MSS favors this distributed method over a centralized service warehouse because it:

- reduces donors’ operational reliance on MSS-CENTER
- distributes MSS management to the lowest levels required
- allows each MSS-ENTITY to choose only those services it finds useful or necessary
- enables donors to choose a MSS-ENTITY that delivers services most in line with their objectives
- gives donors a means to manage their remaining resources with the required CF software
- gives donors a means to virtualize their remaining resources with the required CF software

In addition, sending “heartbeats” from an MSS-ENTITY up to its MSS-ORIGIN serves a two-fold purpose. In Version 1.0, sending heartbeats from the resource ensures that the CF is operational, that donated resources are available for experimentation, and that the resource is authorized to receive services.

In Version 2.0, adding intelligence-filled heartbeats from an MSS-ENTITY to its MSS-ORIGIN provides a means to deliver information, such as which services are in use, how many users are using the service, and other data required by the MSS-CENTER. The MSS-ORIGIN may separate MSS heartbeats from its CF heartbeats, but it will deliver the MSS data to MSS-CENTER. This data is ultimately serves as usage statistics to determine which services deserve continued support.
2. Decomposing the SRS

2.1 System Description
MSS has four main components: MSS-CENTER, MSS-ORIGIN, MSS-AFFILIATE, and MSS-RESOURCE. The actual hardware and software used may vary from component to component, but each component must support service delivery to its children. As of this writing, MSS will only support the remote server as a single-server MSS/CF installation, as described in section 5.1 Remote Server Model.

Currently, the MSS-AFFILIATE architecture only supports computers with two network interface cards and a static IP address or a Fully Qualified Domain Name. ORCA clusters meet these requirements, to include the remote server. In addition, later versions of MSS will support the Windows OS; therefore, one must develop MSS with common workstations and laptop computers in mind.

2.2 Functional Attributes
The System must:

- Deliver services over an Internet connection
- Provide a management interface for MSS administrators
- Allow the administrator to choose a subset of parent services
- Allow the user to choose a service from the parent CF
- Check for heartbeats on MSS-ORIGIN

2.3 Non-functional Attributes
The System shall have:

- Portability
- Autonomy
- Securability
- Credibility
- Integrability
- Extensibility
- Interoperability
- Usability

2.4 System Constraints

2.4.1 Hardware Constraints
Mutualistic Software Services Software Requirements Specification, Version 1.0 lists:

- 5.2 Hardware Interfaces
- 6.1 Performance Requirements

The following tables list examples to help describe the minimum hardware necessary to host MSS-CENTER and a MSS-enabled CF for each component.
Table 1. MSS-CENTER System Description

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiple Servers</td>
<td>These servers may be used as a platform for other MSS Center functions.</td>
</tr>
<tr>
<td>Note:</td>
<td>Servers will be similar to MSS-ORIGIN/AFFILIATE below.</td>
</tr>
</tbody>
</table>

Table 2. MSS-ORIGIN/AFFILIATE Head Node System Description

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make</td>
<td>Dell</td>
</tr>
<tr>
<td>Model</td>
<td>PowerEdge C1100</td>
</tr>
<tr>
<td>Processor</td>
<td>Quad-Core Intel® Xeon® Processor E5620 (12M Cache, VT enabled)</td>
</tr>
<tr>
<td>Processor Clock Speed</td>
<td>2.40 GHz, 5.86 GT/s Intel® QPI</td>
</tr>
<tr>
<td>System RAM</td>
<td>12 GB</td>
</tr>
<tr>
<td>System Storage</td>
<td>1 TB</td>
</tr>
<tr>
<td>Operating System</td>
<td>Unix, Linux with hypervisor support</td>
</tr>
</tbody>
</table>

Table 3. MSS-ORIGIN/AFFILIATE Worker Node System Description

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make</td>
<td>Dell</td>
</tr>
<tr>
<td>Model</td>
<td>PowerEdge C1100</td>
</tr>
<tr>
<td>Processor</td>
<td>2 Quad-Core Intel® Xeon® Processors E5620 (12M Cache, VT enabled)</td>
</tr>
<tr>
<td>Processor Clock Speed</td>
<td>2.40 GHz, 5.86 GT/s Intel® QPI</td>
</tr>
<tr>
<td>System RAM</td>
<td>48 GB</td>
</tr>
<tr>
<td>System Storage</td>
<td>250 GB</td>
</tr>
<tr>
<td>Operating System</td>
<td>Unix, Linux with hypervisor support</td>
</tr>
</tbody>
</table>
Table 4. MSS-AFFILIATE (and Remote Server) System Description

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make</td>
<td>Dell</td>
</tr>
<tr>
<td>Model</td>
<td>PowerEdge 2850</td>
</tr>
<tr>
<td>Processor</td>
<td>2 Dual-core 64-bit Intel Xeon processors</td>
</tr>
<tr>
<td>Processor Clock Speed</td>
<td>2.8GHz</td>
</tr>
<tr>
<td>System RAM</td>
<td>12GB DDR-2 400 SDRAM</td>
</tr>
<tr>
<td>System Storage</td>
<td>&gt; 1 TB</td>
</tr>
<tr>
<td>Operating System</td>
<td>Unix, Linux with hypervisor support</td>
</tr>
</tbody>
</table>

Table 5. MSS-RESOURCE Physical System Description

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make</td>
<td>Apple</td>
</tr>
<tr>
<td>Model</td>
<td>MacBook Pro 15”</td>
</tr>
<tr>
<td>Processor</td>
<td>2.8GHz Intel Core 2 Duo processor with 6MB shared L2 cache</td>
</tr>
<tr>
<td>Processor Clock Speed</td>
<td>2.8GHz</td>
</tr>
<tr>
<td>System RAM</td>
<td>4GB (two 2GB SO-DIMMs) of 1066MHz DDR3 SDRAM</td>
</tr>
<tr>
<td>System Storage</td>
<td>&gt; 300 GB</td>
</tr>
<tr>
<td>Operating System</td>
<td>Unix, Linux, (and Windows in subsequent releases)</td>
</tr>
</tbody>
</table>

2.4.2 Software Constraints
In addition to the Constraints listed in Software Requirements Specification for Mutualistic Software Services Version 1.0

- 2.3 Operating Environment
- 2.4 Design and Implementation Constraints

The System must:

- Support a CF, such as ORCA [8]
- Support a CF-approved virtualized or paravirtualized hypervisor, such as Xen [9]
- Support a CF-approved virtualization technology, such as Eucalyptus [10]
- Support a CF-approved image-passing system, such as Image Proxy [11]
- Only modify the OS in accordance with CF instructions

2.5 User Characteristics

2.5.1 User Classes and Characteristics
Please find this section in *Software Requirements Specification for Mutualistic Software Services Version 1.0*

- 2.2 User Classes and Characteristics

2.5.2 User Groups and Attributes
A typical CF has several actor roles that a member of a MSS-ENTITY fills. The System must support:

<table>
<thead>
<tr>
<th>Actor</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource Donor</td>
<td>Has the authority to donate resources to GENI in exchange for services</td>
</tr>
<tr>
<td>Administrator</td>
<td>Manages all aspects of the CF within the organization in accordance with MSS Center policies and organizational policies, to include:</td>
</tr>
<tr>
<td></td>
<td>• Availability</td>
</tr>
<tr>
<td></td>
<td>• Access</td>
</tr>
<tr>
<td></td>
<td>• Control</td>
</tr>
<tr>
<td></td>
<td>• Child CF connections</td>
</tr>
<tr>
<td></td>
<td>• Parent CF connection</td>
</tr>
<tr>
<td></td>
<td>• Services subset</td>
</tr>
<tr>
<td>Site</td>
<td>Each resource has one or more site actors. For instance, each of the following resources would have one site actor:</td>
</tr>
<tr>
<td></td>
<td>• Eucalyptus Virtual Machine Cluster</td>
</tr>
<tr>
<td></td>
<td>• External Network Switch Cluster</td>
</tr>
<tr>
<td></td>
<td>• Internal Network Switch Cluster</td>
</tr>
<tr>
<td>Broker</td>
<td>Each Site Actor has one or more brokers, who control a portion of the Site’s resources</td>
</tr>
<tr>
<td>Service Manager</td>
<td>Manages all aspects of the CF within his or her organizational department, to include:</td>
</tr>
<tr>
<td></td>
<td>• Availability</td>
</tr>
</tbody>
</table>
User

Experimenter uses the resource delegated by the Service manager to conduct experiments. For business donors, the user conducts the daily business of his or her organizational department, such as management, information technology, marketing, manufacturing, shipping, and customer service.

### 2.6 Assumptions and Dependencies

Please find this section in *Software Requirements Specification for Mutualistic Software Services, Version 1.0*.

#### 2.7 Stakeholders

<table>
<thead>
<tr>
<th>Stakeholder</th>
<th>Major Value</th>
<th>Attitudes</th>
<th>Major Interests</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>NITRD</td>
<td>MSS helps meet its objective to increase GENI experimental resources and bolster the U.S. Ignite program</td>
<td>Concerns about sustained funding for MSS, no matter how useful; cost vs. gain in resources</td>
<td>MSS must be less than the requisite “seed” money; MSS must help meet NITRD objectives</td>
<td>MSS must show a proven increase in GENI donations</td>
</tr>
<tr>
<td>NSF</td>
<td>Increasing LSN size meets NITRD “at-scale” experimental requirement</td>
<td>Concern about cost vs. gain in resources; contingent on GENI leadership attitudes</td>
<td>MSS cost must be less than requisite NSF “seed” money</td>
<td>GENI must maintain a philanthropic public image</td>
</tr>
<tr>
<td>GENI PI</td>
<td>MSS enables “at-scale” experimentation</td>
<td>Relatively small risk for the reward; phased implementation lowers risk</td>
<td>Gain in resources must greatly exceed business/academic contributions</td>
<td>MSS development should not hamper experimentation</td>
</tr>
<tr>
<td>GENI GPO</td>
<td>Control framework growth ensures project continuation</td>
<td>MSS management may overshadow GENI development of other fields</td>
<td>Donor’s interest and perceived value in MSS remains high</td>
<td>MSS management must not outweigh GENI management</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------------</td>
<td>------------------------------------------------------------</td>
<td>------------------------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>GENI Experimenters</td>
<td>“In the wild” resource availability; allows for realistic scenarios</td>
<td>Concern about opt-out policy allowing donors to pull resources in mid-experiment; experimental isolation</td>
<td>At-scale experimentation and resource variety</td>
<td>None</td>
</tr>
<tr>
<td>MSS Developers</td>
<td>Job opportunities; service sharing and potential for monetization; GPL enables code modification, open-source community support</td>
<td>Concerned whether version 1 features will provide enough value to gain/retain donors</td>
<td>Creating new opportunities for service distribution based on emerging research</td>
<td>CFs must deliver MSS to customers to keep a low development overhead.</td>
</tr>
<tr>
<td>Control Frameworks</td>
<td>Increases CF size; increases experimental value; possibly increases NSF funding</td>
<td>Concern about upkeep of MSS</td>
<td>MSS must need minimal CF upkeep; must not make CF vulnerable to cyber attack</td>
<td>MSS must not lose donors due to poor service</td>
</tr>
<tr>
<td>GENI Meta-Operations Center</td>
<td>Increases job security</td>
<td>Concerns about manning for additional workload</td>
<td>Each service must have its own maintain its own support structure</td>
<td>GMOC requires additional funding for additional workload</td>
</tr>
<tr>
<td>GENI Donors</td>
<td>Relatively free services</td>
<td>Concerns about slowed computer/Internet; experiment isolation</td>
<td>MSS should provide reasonable value for the donation</td>
<td>MSS should not hamper organizational operations</td>
</tr>
<tr>
<td>General Public</td>
<td>Provides a way to get involved in the future of the Internet; puts emerging technologies in their hands now</td>
<td>Concerns about increased taxes; whether increased funding is better than current “pay-for” services.</td>
<td>Whether MSS will charge money for services.</td>
<td>MSS should remain “free” to the public</td>
</tr>
</tbody>
</table>
3. System Capabilities, Conditions, and Constraints

Software Requirements Specification for Mutualistic Software Services Version 1.0 lists:

- 2. Overall Description
- 3. System Features
- 4. Functional Requirements
- 5. External Interface Requirements
- 6. Other Non-functional Requirements

This section addresses the system capabilities, conditions, and constraints necessary to implement the above with the functional attributes, non-functional attributes, and constraints identified in this document. The following tables contain the architectural rule, the SRS reference, the solution, the location where the software resides, the solution maintainer, and the owner who controls the solution.

### 3.1 Capabilities

#### Table 8. Software Capabilities

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Ref.</th>
<th>Solution</th>
<th>Location</th>
<th>Creator</th>
<th>Maintainer</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP1</td>
<td>X</td>
<td></td>
<td>FE1</td>
<td>CF compares MSS-AFFILIATE to federation standards</td>
<td>MSS-ORIGIN</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CP2</td>
<td>X</td>
<td></td>
<td>FE4</td>
<td>CF compares MSS-AFFILIATE to connection standards</td>
<td>MSS-ORIGIN</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CP3</td>
<td>X</td>
<td>X</td>
<td>FE3</td>
<td>Service interface describes system updates and services</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FE4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SI2.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CP4</td>
<td>X</td>
<td>X</td>
<td>FE1</td>
<td>Authentication Process authenticates/de-authenticates MSS-AFFILIATES</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FE2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FE3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FE4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CP</td>
<td>X</td>
<td>X</td>
<td>FE3, FE4, SI1.2, CI3</td>
<td>Encryption/Decryption process encrypts/decrypts data</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>-----</td>
<td>---</td>
<td>---</td>
<td>----------------------</td>
<td>------------------------------------------------------</td>
<td>------------</td>
<td>-----------</td>
<td>-----------</td>
<td>------------</td>
</tr>
<tr>
<td>CP6</td>
<td>X</td>
<td>X</td>
<td>FE3, FE4, SI1.5</td>
<td>Validation Process validates transmissions</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CP7</td>
<td>X</td>
<td>X</td>
<td>FR6, FR7, CP1, CP2, CP3, CP4, CP5, CP6, SI1.3, CI3</td>
<td>Service Interface delivers and receives service updates and services</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CP8</td>
<td>X</td>
<td>X</td>
<td>FE5, FE6, CP3, CP4, CS3, CS4, SI1.4</td>
<td>Web interface describes services subset</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CP9</td>
<td>X</td>
<td>X</td>
<td>FE6</td>
<td>SI6.1</td>
<td>SI7.1</td>
<td>SE1</td>
<td>SE2</td>
<td>SE5</td>
</tr>
<tr>
<td>CP10</td>
<td>X</td>
<td>X</td>
<td>FE6</td>
<td>CP3</td>
<td>CP5</td>
<td>CP6</td>
<td>CP9</td>
<td>SI2.2</td>
</tr>
<tr>
<td>CP11</td>
<td>X</td>
<td>X</td>
<td>FR3</td>
<td>FR5</td>
<td>CD12</td>
<td>MSS interface describes users and donors</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
</tr>
</tbody>
</table>
### Table 9. Software Conditions

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Ref.</th>
<th>Solution</th>
<th>Location</th>
<th>Creator</th>
<th>Maintainer</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD1</td>
<td>X</td>
<td>X</td>
<td>SI3.1 SI4.1 SI5.1 SE1 SE2</td>
<td>MSS-ENTITY has a CF installed</td>
<td>MSS-ENTITY</td>
<td>CF</td>
<td>MSS-ENTITY</td>
<td>CF</td>
</tr>
<tr>
<td>CD2</td>
<td>X</td>
<td>X</td>
<td>FR6 FR7 SI1.6 SI2.3 RE1 RE2</td>
<td>File system holds services</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CD3</td>
<td>X</td>
<td>X</td>
<td>OE7 FE3 FE4 CD1 SI1.7</td>
<td>Services database holds system updates and services data</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CD4</td>
<td>X</td>
<td>X</td>
<td>OE7</td>
<td>CF database holds MSS-ENTITY data</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CD5</td>
<td>X</td>
<td>X</td>
<td>OE7</td>
<td>User database holds user data</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>CD6</td>
<td>X</td>
<td>CI1</td>
<td>CI1 CENTER receives emails</td>
<td>MSS-CENTER</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>---</td>
<td>------</td>
<td>----------------------------</td>
<td>------------</td>
<td>-----------</td>
<td>-----------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CI2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RE1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RE2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CD7</th>
<th>X</th>
<th>CI4</th>
<th>CI4 CENTER receives files</th>
<th>MSS-CENTER</th>
<th>Developer</th>
<th>Developer</th>
<th>MSS-CENTER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SE1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SE5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RE1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RE2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD8</td>
<td>X</td>
<td>UD1</td>
<td>UD2</td>
<td>UD3</td>
<td>OE7</td>
<td>CO1</td>
<td>CO2</td>
</tr>
<tr>
<td>-----</td>
<td>---</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td></td>
<td></td>
<td>User web interface accesses GENI and service information, including:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD9</td>
<td>X</td>
<td>UD4</td>
<td>UD5</td>
<td>UD6</td>
<td>UD7</td>
<td>DE4</td>
<td>DE5</td>
</tr>
<tr>
<td>------</td>
<td>---</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Administrator web interface accesses management information, including: HTML File Downloads</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD10</td>
<td>X</td>
<td>X</td>
<td>AS1</td>
<td>AS2</td>
<td>AS3</td>
<td>AS4</td>
<td>SI1.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSS interface connects many MSS-USERS to many MSS-ENTITIES and MSS-RESOURCES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD11</td>
<td>X</td>
<td>X</td>
<td>AS4</td>
<td>SI1.8</td>
<td>SI1.9</td>
<td>SI1.9.1</td>
<td>SI1.10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CF interface sends heartbeats to MSS-ORIGIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD12</td>
<td>X</td>
<td>X</td>
<td>AS4</td>
<td>SI1.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CF interface receives heartbeats from MSS-AFFILIATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSS-ENTITIES Developer CF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSS-DEVELOPER CF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSS-CENTER CF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSS-CENTER CF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

~ 50 ~
| CD13 | X | X | AS4 | FR4 | CD8 | SE4 | SE8 | CF interface encodes heartbeats to MSS-ORIGIN | MSS-AFFILIATE | Developer CF | CF | CF |
| CD14 | X | AS4 | FR4 | CD8 | SI1.12 | CI3 | MSS interface decodes heartbeats from MSS-AFFILIATE | MSS-ORIGIN | Developer CF | CF | CF |
| CD15 | X | X | FR1 | FR2 | CP5 | IN1 | IN2 | Database holds user and donor data | MSS-ENTITY | Developer | Developer | MSS-CENTER |
### 3.3 Constraints

**Table 10. Software Constraints**

<table>
<thead>
<tr>
<th>Rule</th>
<th>Version 1.0</th>
<th>Version 2.0</th>
<th>Ref.</th>
<th>Solution</th>
<th>Location</th>
<th>Creator</th>
<th>Maintainer</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS1</td>
<td>X</td>
<td>X</td>
<td>OE1</td>
<td>MSS must operate on Linux platforms initially and port to several types of hardware platforms later</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OE5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OE6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FL1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FL2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FL3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS2</td>
<td>X</td>
<td>X</td>
<td>OE2</td>
<td>MSS libraries must operate on Unix and Linux platforms initially and Windows platforms later</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OE6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS3</td>
<td>X</td>
<td>X</td>
<td>OE3</td>
<td>MSS web interfaces must display correctly on major web browsers</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td>-----</td>
<td>---</td>
<td>---</td>
<td>------</td>
<td>---------------------------------------------------------------</td>
<td>------------</td>
<td>-----------</td>
<td>-----------</td>
<td>------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS4</td>
<td>X</td>
<td>X</td>
<td>OE4</td>
<td>MSS web interfaces must display in English, with the ability to add other languages later</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UI6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS5</td>
<td>X</td>
<td></td>
<td>OE5</td>
<td>MSS must verify donor location data to ensure US location</td>
<td>MSS-ENTITY</td>
<td>Developer</td>
<td>Developer</td>
<td>MSS-CENTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SE2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS6</td>
<td>X</td>
<td>S11.1</td>
<td>MSS-CENTER sets MSS standards</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. System Characteristics

4.1 Autonomy
The System shall:

- Not require MSS-CENTER interaction to use existing parent services.
- Be separable from its parent node without incurring interruption of current services

4.2 Integrability
The System shall:

- Install on a CF with no modification to the CF, excepting the remote server. For instance:
  - ORCA only requires a MSS scripts and configuration file adjustments using commonly installed software, such as Apache, MySQL, and PHP.

4.3 Extensibility
The System shall:

- Easily add child nodes and resources, for instance:
  - An administrator adds a child CF by inserting the child’s IP, port, encryption key, or other identity information into a file.

4.4 Portability
The System shall:

- (Version 1.0 and 2.0) Operate on Linux
- (Version 2.0) Operate on Unix, and Windows OS
- Operate on servers

4.5 Usability
The System shall:

- Be geared towards users in a non-technical field, such as clerical.
- Be geared towards administrators in a non-technical field, for instance:
  - MSS will offer helpful instructions without technical jargon.
4.6 Securability
The System shall:

- (Version 1.0 and 2.0) At a minimum, require a Unique user name and password for login
- (Version 2.0) Provide an option for elevating secure access for different user groups, for instance:
  - User – Unique user name and password for login
  - Service Manager – add Captcha login
  - Owner/Administrator – add Common Access Card login
  - This option may itself be a service

4.7 Credibility
The System shall:

- Provide a means to verify the service as a MSS service, for instance:
  - list a MD5 hash value
  - Provide a MSS signature
- Easily connect to a parent sponsor, or change to a new parent sponsor

4.8 Interoperability
The System shall:

- Easily exchange information between parent and child sponsors, for instance:
  - MSS parent services are queriable and downloadable through a web browser
5. Architectural Plan

MSS relies on the notion that experimenters use the GENI architecture to develop at least some unique services that other organizations cannot offer. In fact, other experimenters, developers, and users find these services valuable enough to donate their resources to the experimenters in exchange for these services. With more resources available, more experimenters can create more services, and the mutualistic cycle continues.

Version 1.0 provides a skeleton architecture in preparation for Version 2.0 by creating a service advertisement and delivery system between two MSS-AFFILIATES that administrators can use to download entire subsets of services, and that resource users can use to query and download individual services. MSS checks for the heartbeats sent to the MSS-ORIGIN before allowing administrators and users to download services onto the actor. The following figure depicts the hierarchical relationships between the entities.

In Version 2.0, this architecture combines MSS Version 1 & 2 software requirements. It provides a means for the MSS-ORIGINS to plug into the MSS-CENTER and receive the services that MSS-DEVELOPERS upload and maintain. An MSS-ORIGIN (node 1) then advertises the applicable subset of services, and MSS-AFFILIATES (nodes 2 & 3) may choose all, or a subset of its parent’s services. At each node, users can download individual services on to authorized MSS-RESOURCES.

In order to receive the services in Version 1.0 and 2.0, donors simply host a CF and dedicate a portion of their resources for GENI experimentation. After donors advertise their resources as available by sending heartbeats to its MSS-ORIGIN, it may download services from its parent MSS-AFFILIATE.

The hardware footprint of the system depends entirely on how many child CFs the parent owns. MSS Version 2.0 may look like this:

- MSS-CENTER will require a separate server or servers for each MSS component
- MSS-ORIGIN/AFFILIATES will require several servers: one to hold MSS, one to hold services, and several for its GENI CF
MSS-AFFILIATES, at the lowest level, will only require a portion of one server, workstation, or laptop and serve as its own resource.

MSS-RESOURCE, at the lowest level, will only require a portion of one server, workstation, or laptop.

“MSS-AFFILIATES, at the lowest level...” (above) forms a special case in which all software components reside on one server. To meet this goal, the following section describes an architecture that resides on one server, and that conforms to MSS-CHILD requirements.

5.1 Remote Server Model

According to Software Requirements Specification for Mutualistic Software Services Version 1.0, a MSS-AFFILIATE must:

- [Maintain a] Constant connection to the Internet, through a Cable Modem, Direct Service Line, or other network connection
- [Be a] A computer resource, provisioned according to CF standards.

This means that a MSS-AFFILIATE may only have a single internet connection and must reside on a single computer, which is not how one sets up a typical CF.

Normally, a CF resides on several computers, with one head node and several worker nodes. The CF may have several IP addresses, or it might have only one IP address and use a separate Network Address Translation (NAT) server to shuffle traffic between the public address space and a private address space where the CF resides. In a canonical ORCA installation, a CF must have at least two computers or several IP addresses to run all of the required software components [13], which violates the SRS minimum hardware interface requirements.

This is not an easy problem to solve because the virtualization software (e.g., Xen) requires direct access to the computer kernel to run a hypervisor, and the virtualization management software (e.g., Eucalyptus) requires direct access to the networking tables (e.g., iptables) to allocate public or pseudo-public IP addresses using Dynamic Host Control Protocol (DHCP). Therefore, one cannot use NAT on the hypervisor, and one cannot run Eucalyptus on a virtual machine because the Eucalyptus DHCP server would hand out addresses to other computers on its network that are not Eucalyptus Instances.

The remote server model creates an unexploited solution (at least in the author’s research) that uses a virtual NAT server to control the network connections, shields the outside network from the Eucalyptus DHCP server, and leaves the Eucalyptus software to control virtualization on the hypervisor. Appendix B contains a component diagram that illustrates how one might implement a remote server to act as a MSS-AFFILIATE.
5.2 Context Diagrams and Data Model

Appendix B contains two context diagrams and one data model. The context diagrams CD-1 and CD-2 represent the external connections, and the data flow model DM-1 represents the generation and flow of data for each MSS entity in version 1.0.

5.2.1 MSS-CENTER
- (Version 2.0) Hosts all GENI CFs to utilize their heartbeat functionalities
- (Version 2.0) Receives heartbeats, service information requests, and service requests from its child CFs
- (Version 2.0) Delivers service information and service requests to its child CFs
- (Version 2.0) Delivers service content to resource users and administrators

5.2.2 MSS-ORIGIN
- Develops and maintains a GENI CF
- Has one or more MSS-AFFILIATES
- Receives heartbeats from its MSS-AFFILIATES
- (Version 2.0) Receives service information requests and service requests from its MSS-AFFILIATES
- (Version 2.0) Delivers service information and service requests to its MSS-AFFILIATES

5.2.3 MSS-AFFILIATE
- Hosts a GENI CF and maintains resources
- Delivers heartbeats to its MSS-ORIGIN
- Receives service information requests, and service requests from its child MSS-AFFILIATES
- Delivers service information and service requests to its child MSS-AFFILIATES

5.2.4 MSS-RESOURCE
- Authorized by an MSS-ENTITY to receive services, make service information requests, and receive service information
- Has authorized MSS-USERS

5.2.5 MSS-USER
- Authorized to use at least one MSS-RESOURCE
- Requests service information and services from his or her MSS-RESOURCE to his or her MSS-ENTITY
- Receives service information and services from his or her MSS-RESOURCE to his or her MSS-ENTITY
- (Version 2.0) Requests service content from MSS-CENTER
- (Version 2.0) Receives service content from MSS-CENTER

5.2.5 MSS-DEVELOPER
- Usually a GENI experimenter
- Creates MSS services
5.3 Component Model

Appendix B contains one component diagram. The component diagram CM-1 represents the external connections, and the data flow model DM-1 represents the generation and flow of data for each MSS-ENTITY.

5.3.1 Databases

5.3.1.1 Service Data
• Holds the subset of service attribute data of its MSS-ENTITY

5.3.1.2 CF Data
• Holds the MSS-ENTITY data for itself. For instance, ORCA uses database orca

5.3.1.3 User Data
• Holds the authorized user’s data for itself and its child MSS-AFFILIATES

5.3.2 Components

5.3.2.1 CF
• MSS-ORIGIN presents an interface to its children to receive heartbeats
• MSS-AFFILIATES deliver heartbeats to its MSS-ORIGIN
• Provides a CF interface with a list of current donors. For instance, ORCA provides the ORCA Actor Registry [14]
• Provides a list of available resources to GENI experimenters and users
• Enables GENI experimenters and users to request resources

5.3.2.2 User Interface
• Uses the Authentication process to verify actor and user requests
• Receives a list of MSS-ENTITY resources from the CF
• Receives user service information and service requests
• Provides service delivery information to the Service Interface
• Triggers the Service Interface to deliver services
• Checks for heartbeats on the CF interface
• Triggers the Service Interface to download services

5.3.2.3 Service Interface
• Receives a service information request from the User Interface
• Receives a service download request from the User Interface
• Uses the Encryption process to encrypt services
• Uses the Decryption process to decrypt services
• Uses the Validation process to validate services
• Receives services from parent MSS-ENTITY
• Delivers services to child MSS-AFFILIATES

5.3.2.4 User Web Site
• (Version 2.0) Delivers service content to service users
• (Version 2.0) Enables MSS-CENTER/MSS-DEVELOPERS to add service content
• (Version 2.0) Delivers GENI information as required by MSS-CENTER
• (Version 2.0) Enables users to email MSS-CENTER/MSS-DEVELOPERS

5.3.2.5 Administrator Web Site
• (Version 2.0) Delivers administrative content to MSS administrators
• (Version 2.0) Allows MSS-CENTER/MSS-DEVELOPERS to add administrative content
• (Version 2.0) Delivers GENI information as required by MSS-CENTER
• (Version 2.0) Enables users to email MSS-CENTER/MSS-DEVELOPERS

5.3.2.6 Developer Uploads
• (Version 2.0) Allows enrolled MSS-DEVELOPERS to add services
• (Version 2.0) Allows enrolled MSS-DEVELOPERS to add service descriptions
• (Version 2.0) Enables MSS-CENTER to approve services
• (Version 2.0) Enables MSS-CENTER to approve service descriptions

5.3.3 Service Repository

5.3.3.1 File System
• Holds MSS services
• (Version 2.0) Holds all MSS services for MSS-CENTER
• Holds a subset of parent services, to include the entire subset, for its MSS-AFFILIATES
6. Architecture Trade-off Analysis Method (ATAM)

6.1 Purpose

*Software Architecture in Practice* (2003) explains that the ATAM “… reveals how well an architecture satisfies particular quality goals, and (because it recognizes that architectural decisions tend to affect more than one quality attribute) it provides insight into how quality goals interact—that is, how they trade off” [1]. To this end, this section evaluates MSS in order to verify the architectural decisions that make up the MSS architecture.

6.2 Main Architectural Drivers

From 2.3 Non-functional Attributes

The System shall have:

- Portability
- Autonomy
- Securability
- Credibility
- Integrability
- Extensibility
- Interoperability
- Usability

6.3 Business Goals

- MSS-DEVELOPERS can advertise services
- GENI CFs can exchange services for a portion of a resource owner’s resources
- An individual member of the public can donate his or her resources in exchange for services
- GENI CFs grow as a result of exchanging services for resources
- A break in heartbeats will not affect the operation of current services
- Decentralized management to reduce MSS-CENTER operational costs

6.4 Major Stakeholders

*Mutualistic Software Services Software Requirements Specification, Version 1.0* lists:

- 2.2 User Classes and Characteristics

Favored User Classes:

- Resource donors
- CFs
- Developers
- GENI Project Office (GPO)
6.5 Architectural Approaches

MSS:

- separates databases according to the data held
- requires that a MSS-AFFILIATE send MSS-ORIGIN heartbeats to download new services
- requires that only authorized users on authorized MSS-AFFILIATES and MSS-RESOURCES can download new services
- CF Interface:
  - provides the CF Interface with a list of donors [14]
  - provides the User Interface with a list of currently connected MSS-ENTITIES
- User Interface:
  - authenticates MSS-ENTITIES and users, and validates service requests
  - requests service information and services from its parent
  - validates donation status from its MSS-AFFILIATE and its children
  - delivers service information to its children
  - triggers the Service Interface service transmission and reception
- Service Interface:
  - transmits and receives services
  - encrypts and decrypts services for transmission
  - validates whether services are successfully delivered or received

6.6 Utility Tree

Table 11. Utility Tree

<table>
<thead>
<tr>
<th>Quality Attribute</th>
<th>Attribute Refinement</th>
<th>Case</th>
<th>Scenario</th>
<th>Rating (Importance, Difficulty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portability</td>
<td>OS Change</td>
<td>1.</td>
<td>A donor installs MSS on a new Linux OS distribution with the ORCA control framework. This OS has a newer kernel than the old OS.</td>
<td>(H, M)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Best: MSS works with the new OS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Worst: MSS requires older libraries than the OS supports</td>
<td></td>
</tr>
<tr>
<td>CF Change</td>
<td></td>
<td>2.</td>
<td>A donor installs MSS on a Unix OS with the protoGENI control framework. The CF is compatible with Unix, but the CF does not use PHP, on which the MSS interface relies.</td>
<td>(H, M)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Best: PHP interpreter and libraries are available from the Unix package manager, such as aptitude.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Worst: The administrator must install the PHP interpreter and appropriate libraries from the PHP repository.</td>
<td></td>
</tr>
</tbody>
</table>
3. A donor installs MSS resources on a virtual machine. (H, L)

Best: MSS services work on the virtual machine.

Worst: MSS services will not work on the virtual machine.

4. A user requests a new service from its parent MSS-ENTITY, but the MSS-ENTITY is not sending heartbeats to MSS-ORIGIN. (H, H)

Best: The MSS-ENTITY does not deliver the new service.

Worst: The MSS-ENTITY does deliver the new service.

5. A user requests a currently owned service from its parent MSS-ENTITY that is sending heartbeats. (H, H)

Best: The parent CF does deliver the service.

Worst: The parent CF does not allow the service download.

6. A MSS-RESOURCE requests a service from its parent MSS-ENTITY, but the MSS-ENTITY is not sending heartbeats. (H, H)

Best: The CF does not deliver the service to the resource.

Worst: The CF does deliver the service to the resource.

7. A MSS-ENTITY has more stringent donor rules than its parent. Its MSS-AFFILIATES are unhappy with the strict rules. (H, M)

Best: Another MSS-ENTITY advertises donor rules that are in keeping with this MSS-AFFILIATE. The child switches parents by contacting another MSS-ENTITY, providing MSS-ENTITY, MSS-RESOURCE, and MSS-USER data, and sending heartbeats to MSS-ORIGIN.

Worst: The children of the MSS-ENTITY with strict donor rules remove their resources permanently from GENI.

8. A MSS-ENTITY has less stringent donor rules than its parent. (M, L)
Best: The resource donors enjoy the less stringent rules and the MSS-ENTITY maintains the hardware necessary to support the increase in donors.

Worst: This attracts more donors than the MSS-ENTITY can adequately serve. The donors permanently remove their resources from GENI.

9. A MSS-ENTITY stops using service encryption to deliver services. This allows hackers to “pluck” the services from the Internet during transmission.

Best: The parent of the MSS-ENTITY starts using encryption keys, and once again starts encrypting service transmissions.

Worst: Several MSS-ENTITIES stop encrypting services, and the services are available to a large number of non-donating resource owners, which devalues resource donation in exchange for services because most of these services are now free. Donations dwindle to a point where MSS loses usefulness.

10. A donor connects to a CF and obeys all rules, except he or she gives GENI services away to others who are not donating.

Best: The parent notices a large number of downloads by the donor, and enforces MSS rules that require service donations for services.

Worst: The parent does not notice the increased downloads from the offending donor, and the services are available to a large number of non-donating resource owners, which devalues resource donation in exchange for services because most of these services are now free. Donations dwindle to a point where MSS loses usefulness.

11. A donor connects to a CF only long enough to download services and then intentionally stops sharing resources with the CF. Once a week, this same donor connects again, downloads new services, and disconnects from the parent CF.

Best: The MSS-PARENT keeps a history of donor connection times. The MSS-PARENT rules are strict.
enough that the donor cannot download more new services until the donor meets the resource donation threshold.

Worst: The parent does not notice the connection history of the offending donor, and too few resources are available to experimenters than MSS maintenance is worth.

MSS-ENTITY Encryption 12. A branch of MSS-ENTITIES uses the same encryption keys for its MSS-AFFILIATES. (H, L)

Best: The parent at the highest point in the branch starts changing encryption keys after a specified time, which requires its children down the chain to follow suit out of necessity.

Worst: One MSS-ENTITY is hacked, which leaves all of the CFs in the branch vulnerable to attack.

Credibility CF Fraud 13. A donor connects to a disreputable parent CF, and user identities, encryption keys, and other information are stolen. (H, M)

Best: Encryption keys are replaced.

Worst: User identity information containing sensitive personally identifiable information is in criminal hands.

Poor Service Operation 14. The service does not perform as the developer described. (M, M)

Best: Users voice their displeasure on the User Web Site, and the developer responds by enhancing the service operation or describing it more accurately.

Worst: Poor service operation is common throughout MSS, which drives users away from donating resources.

Poor Service Descriptions 15. The service does not have the functionality that the service description provides. (M, M)

Best: Users voice their displeasure on the User Web Site, and the developer responds by enhancing the service operation or describing it more accurately.

Worst: Poor service descriptions are common throughout
<table>
<thead>
<tr>
<th>Poor Service Instructions 16.</th>
<th>MSS, which drives users away from donating resources.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best:</td>
<td>The service does not have clear, accurate directions for service use, or is missing directions.</td>
</tr>
<tr>
<td>Worst:</td>
<td>Users voice their displeasure on the User Web Site, and the developer responds by enhancing the service operation or describing it more accurately.</td>
</tr>
<tr>
<td></td>
<td>Poor service instructions are common throughout MSS, which drives users away from donating resources.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Integrability CF Versions 17.</th>
<th>A new CF version comes out, and MSS must work with both versions.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best:</td>
<td>MSS requires no change to work with the new version, or the changes are minimal because MSS is developed with integrability in mind.</td>
</tr>
<tr>
<td>Worst:</td>
<td>MSS requires significant changes to work with each version of every CF, and the lack of “plug-n-play” functionality overburdens MSS developers.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Database type 18.</th>
<th>CFs choose to use a different database than the design prescribes. Some database queries do not work.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best:</td>
<td>The database uses similar statements to the intended database, and so requires little work to integrate.</td>
</tr>
<tr>
<td>Worst:</td>
<td>Each CF decides to use different databases with dissimilar statements, and so developers must tailor software commands to each database type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CF Interface Compatibility 19.</th>
<th>A donor decides to change CFs without changing MSS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best:</td>
<td>The CF Interface requires no change to work with the new CF.</td>
</tr>
<tr>
<td>Worst:</td>
<td>The CF interface has significant compatibility problems.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Interface 20.</th>
<th>Database version changes format of its query structure.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(L, L)</td>
</tr>
<tr>
<td>Compatibility</td>
<td>Best:</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
</tr>
<tr>
<td></td>
<td>Worst:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Extensibility Parent Nodes</th>
<th>21.</th>
<th>MSS-ENTITY has poor service because it does not have the hardware to support all of its children.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best:</td>
<td>Resource donors easily switch to a new MSS-ENTITY that perhaps the current MSS-ENTITY recommends.</td>
</tr>
<tr>
<td></td>
<td>Worst:</td>
<td>MSS-ENTITY does not acquire the hardware, nor does the donor switch. Service is so poor that donors leave MSS.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tree Balancing</th>
<th>22.</th>
<th>Too many child nodes create a long tree branch from MSS-CENTER, which slows down cascading service delivery.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best:</td>
<td>This MSS branch re-links itself to balance the sub-tree, and services flow quickly.</td>
</tr>
<tr>
<td></td>
<td>Worst:</td>
<td>MSS-ENTITIES continue to ignore sub-tree lengths, and service delivery slows to a point that donors stop donating resources.</td>
</tr>
</tbody>
</table>
7. Cost Benefit Analysis Method (CBAM)
Len Bass et al., summarize the purpose of the CBAM in *Software Architecture in Practice* (2003):

The CBAM is an iterative elicitation process combined with a decision analysis framework. It incorporates scenarios to represent the various quality attributes. The stakeholders explore the decision space by eliciting utility-response curves to understand how the system’s utility varies with changing attributes. The consensus basis of the method allows for active discussion and clarification amongst the stakeholders. The traceability of the design decision permits updating and continuous improvement of the design process over time [1].

The MSS CBAM meets this goal by creating a utility response curve for each case in the Utility Tree rated (H, H), (H, M), or (M, H). These cases are ordered according to their:

- Utility response curves and ratings
- The architectural strategy to combat the negative effects of the worst-case scenarios
- The cost of implementing the strategy
- List potential risks that the strategy may not mitigate

7.1 Utility Response Curves

7.1.1 Autonomy vs. Service Delivery

Table 12. Autonomy vs. Service Delivery

<table>
<thead>
<tr>
<th>Service Delivery</th>
<th>Case Description</th>
<th>Best Scenario</th>
<th>Worst Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>(H, H)</td>
<td>A user requests a new service from its parent MSS-ENTITY, but the parent CF is not sending heartbeats to MSS-ORIGIN.</td>
<td>The MSS-ENTITY does not deliver the new service.</td>
<td>The user can download the new service, even though the MSS-ENTITY is not sending heartbeats.</td>
</tr>
<tr>
<td>(H, M)</td>
<td>A user requests a currently owned service from its parent MSS-ENTITY, but its parent MSS-ENTITY does not have a parent sponsor.</td>
<td>The parent MSS-ENTITY delivers the service because it is sending heartbeats.</td>
<td>The parent CF does not allow the service download, even though it is sending heartbeats.</td>
</tr>
<tr>
<td>(M, H)</td>
<td>A user requests a currently owned service from its parent MSS-ENTITY, but the MSS-ENTITY is not sending heartbeats to MSS-ORIGIN.</td>
<td>The MSS-ENTITY does not deliver the service.</td>
<td>The MSS-ENTITY does not deliver the service to the</td>
</tr>
</tbody>
</table>
resource.

Worst: The CF does deliver the service to the resource.

<table>
<thead>
<tr>
<th>Strategy / Cost</th>
<th>Description</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Strictly adhere to the MSS architecture, in which heartbeats must go to MSS-ORIGIN.</td>
<td>LOW – Ensure developers adhere to the MSS Architecture</td>
</tr>
<tr>
<td>2.</td>
<td>Strictly adhere to the MSS architecture, in which the MSS-ENTITY must be enrolled in the ENTITY database</td>
<td>LOW – Ensure developers adhere to the MSS Architecture</td>
</tr>
<tr>
<td>3.</td>
<td>Strictly adhere to the MSS architecture, in which the user must be enrolled in User database</td>
<td>LOW – Ensure developers adhere to the MSS Architecture</td>
</tr>
</tbody>
</table>

7.1.2 Integrability vs. User Interface Compatibility

**Table 13. Integrability vs. User Interface Compatibility**

<table>
<thead>
<tr>
<th>User Interface Compatibility</th>
<th>Description</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>19.</td>
<td>A donor decides to change CFs without changing MSS.</td>
<td>LOW – This approach fits with the architectural model</td>
</tr>
<tr>
<td>Best:</td>
<td>The MSS Interface requires no change to work with the new CF.</td>
<td>LOW – This approach fits with the architectural model</td>
</tr>
<tr>
<td>Worst:</td>
<td>The MSS interface has significant compatibility problems.</td>
<td>LOW – This approach fits with the architectural model</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strategy / Cost</th>
<th>Description</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Use the same software as the CF, which is generally MySQL, PHP, and BASH scripts.</td>
<td>LOW – This approach fits with the architectural model</td>
</tr>
<tr>
<td>2.</td>
<td>Use the same software libraries as the CF. The CFs already have authentication, decryption, and validation processes in place, and so using these causes as few compatibility problems as possible.</td>
<td>LOW – This approach fits with the architectural model</td>
</tr>
</tbody>
</table>

7.1.3 Credibility vs. CF Fraud

**Table 14. Credibility vs. CF Fraud**

<table>
<thead>
<tr>
<th>CF Fraud</th>
<th>Description</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.</td>
<td>A donor connects to a disreputable parent CF, and user identities, encryption keys, and other information are stolen.</td>
<td>LOW – This approach fits with the architectural model</td>
</tr>
<tr>
<td>(H, M)</td>
<td>Best: Encryption keys are replaced.</td>
<td>LOW – This approach fits with the architectural model</td>
</tr>
</tbody>
</table>
Worst: User identity information containing sensitive personally identifiable information is in criminal hands.

Strategy / Cost

1. Trust Campaign – begin a real-world campaign, such as on the NSF, GENI, CF, and MSS-CENTER websites, explaining that a donor should only attach to a reputable organization. Avoid unknown parents.

2. MSS Instructions have notices and warnings prominently displayed that warn the donor to only join a trustworthy MSS parent, such as a university or respected business.

MEDIUM – MSS-CENTER can add this content to which other agencies link, but it adds additional development and maintenance costs.

7.1.4 Extensibility vs. Tree Balancing

Table 15. Extensibility vs. Tree Balancing

<table>
<thead>
<tr>
<th>Tree Balancing</th>
<th>22.</th>
<th>Too many child nodes create a long tree branch from MSS-CENTER, which slows down cascading service delivery.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(H, M)</td>
<td></td>
<td>Best: This MSS branch re-links itself to balance the sub-tree, and services flow quickly.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Worst: MSS-ENTITIES continues to ignore sub-tree lengths, and service delivery slows to a point that donors stop donating resources.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strategy / Cost</th>
<th>1.</th>
<th>Self-balancing binary search tree – In computer science, a self-balancing (or height-balanced) binary search tree is any node based binary search tree that automatically keeps its height (number of levels below the root) small in the face of arbitrary item insertions and deletions [12].</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.</td>
<td>Recommend to MSS-ENTITIES on the Administrator Web Site to keep balanced branches to the greatest extent possible to speed service delivery.</td>
</tr>
</tbody>
</table>

HIGH – implementing this requires MSS-CENTER to know all of its nodes and MSS-ENTITY rules at each level.

MEDIUM – MSS-CENTER can add this content, to which other agencies link, but it adds additional development and maintenance costs.
7.1.5 Integrability vs. CF Versions

Table 16. Integrability vs. CF Versions

<table>
<thead>
<tr>
<th>CF Versions</th>
<th>17. A new CF version comes out, and MSS must work with both versions.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best: MSS requires no change to work with the new version, or the changes are minimal because MSS is developed with integrability in mind.</td>
</tr>
<tr>
<td></td>
<td>Worst: MSS requires significant changes to work with each version of every CF, and the lack of “plug-n-play” functionality overburdens MSS developers</td>
</tr>
</tbody>
</table>

Strategy / Cost

1. Use the same software as is loaded on the CF, which is generally MySQL, PHP, and BASH scripts. LOW – This approach fits with the architectural model

2. Use the same software libraries as the CF. The CFs already have authentication, decryption, and validation processes in place, and so using these causes as few compatibility problems as possible. LOW – This approach fits with the architectural model

7.1.6 Securability vs. Service Fraud

Table 17. Securability vs. Service Fraud

<table>
<thead>
<tr>
<th>Service Fraud</th>
<th>10. A donor connects to a CF and obeys all rules, except he or she gives GENI services away to others who are not donating.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best: The parent notices a large number of downloads by the donor, and enforces MSS rules that require service donations for services.</td>
</tr>
<tr>
<td></td>
<td>Worst: The parent does not notice the increased downloads from the offending donor, and the services are available to a large number of non-donating resource owners, which devalues resource donation in exchange for services because most of these services are now free. Donations dwindle to a point where MSS loses usefulness.</td>
</tr>
</tbody>
</table>

Strategy / Cost

1. Management – include color coding and other means to highlight children that download more than a given amount. RISK – This strategy may not mitigate the worst-case scenario

MEDIUM – The parent CF needs a way to know how many downloads a child has, to recognize each child (by its
MAC address, for instance) and to permanently exclude the offending child.

2. Add MAC address to MSS-ENTITY database

MEDIUM – Adding a MAC address is not difficult, but GENI CFs do not currently require donors to provide a MAC address. GENI must approve this unilaterally.

7.1.7 Portability vs. OS Change

Table 18. Portability vs. OS Change

<table>
<thead>
<tr>
<th>OS Change</th>
<th>1. A donor installs MSS on a new Linux OS distribution with the ORCA control framework. This OS has a newer kernel than the old OS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(H, M)</td>
<td>Best: MSS works with the new OS</td>
</tr>
<tr>
<td></td>
<td>Worst: MSS requires older libraries than the OS supports</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strategy / Cost</th>
<th>1. MSS Instructions stress using LTS OS distributions to avoid this problem altogether.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy / Cost</td>
<td></td>
</tr>
</tbody>
</table>

MEDIUM – MSS-CENTER can add this content to which other agencies link, but it adds additional development and maintenance costs. PHP, BASH, and MySQL are widely supported by all Linux distributions. The best strategy to combat this is to avoid “bleeding edge” distributions, such as beta and daily build distributions, which are generally used for testing and not for enterprise level deployments.
### 7.1.8 Portability vs. CF Change

#### Table 19. Portability vs. CF Change

<table>
<thead>
<tr>
<th>CF Change</th>
<th>Strategy / Cost</th>
<th>Reasoning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(H, M)</td>
<td>1. MSS Instructions stress using LTS OS distributions to avoid this problem altogether.</td>
<td>MEDIUM – MSS-CENTER can add this content to which other agencies link, but it adds additional development and maintenance costs. PHP, BASH, and MySQL are widely supported by all Unix distributions. The best strategy to combat this is to avoid “bleeding edge” distributions, such as beta and daily build distributions, which are generally used for testing and not for enterprise level deployments.</td>
</tr>
<tr>
<td>A donor installs MSS on a Unix OS with the protoGENI control framework. The CF is compatible with Unix, but the CF does not use PHP, on which the MSS interface relies.</td>
<td>2. The administrator must install the PHP interpreter and appropriate libraries from the PHP repository.</td>
<td></td>
</tr>
</tbody>
</table>

Best: PHP interpreter and libraries are available from the Unix package manager, such as *ports*.
7.1.9 Autonomy vs. MSS-ENTITY Rules

Table 20. Portability vs. CF Change

<table>
<thead>
<tr>
<th>MSS-ENTITY Rules</th>
<th>7. A MSS-ENTITY has more stringent donor rules than its parent. The children of the MSS-ENTITY are unhappy with the strict rules.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Best:</strong> Another MSS-ENTITY advertises donor rules that are in keeping with this MSS-AFFILIATE. The child switches parents by contacting another MSS-ENTITY, providing MSS-ENTITY data and User data, and sending heartbeats to the MSS-ORIGIN.</td>
</tr>
<tr>
<td></td>
<td><strong>Worst:</strong> The children of the MSS-ENTITY with strict donor rules remove their resources permanently from GENI.</td>
</tr>
<tr>
<td>Strategy / Cost</td>
<td>1. MSS-AFFILIATES can choose whichever MSS-ENTITY meets their needs or can switch to a different parent. An MSS-AFFILIATE can also become a parent MSS-ENTITY. However, it still must obey the minimum connection, service distribution, and service delivery requirements of the ultimate parent, MSS-CENTER.</td>
</tr>
</tbody>
</table>

7.1.10 Securability vs. Donation Fraud

Table 21. Securability vs. Donation Fraud

<table>
<thead>
<tr>
<th>Donation Fraud</th>
<th>11. A donor connects to a CF only long enough to download services and then intentionally stops sharing resources with the CF. Once a week, this same donor connects again, downloads new services, and disconnects from the parent CF.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Best:</strong> The MSS-ORIGIN keeps a history of donor connection times. The MSS-ENTITY rules are strict enough that the donor cannot download more new services until the donor meets the resource donation threshold.</td>
</tr>
<tr>
<td></td>
<td><strong>Worst:</strong> The parent does not notice the connection history of the offending donor, and too few resources are available to experimenters than MSS maintenance is worth.</td>
</tr>
</tbody>
</table>
Strategy / Cost

1. Management – Each parent MSS-ENTITY decides the required connection criteria for each child, such as how many hours of the day the child is actively donating resources.

NONE – In the MSS Architecture, the parent MSS-ENTITY can enforce its own connection rules. Each parent in the chain has this ability, and so an entire branch may be cut off from new services until it meets the higher parent's connection criteria.

7.2 Architectural Strategies

The following strategies are ranked according to its Utility rating

- Adhere to architecture requirement for heartbeats, and MSS-ENTITY and user data
- Use the same software and libraries that are on the CFs
- (Version 2.0) Add to the MSS web sites:
  - Trust Campaign – Reputable parent and child information
  - The importance of balancing child nodes
  - Define service fraud and how to prevent it.
  - Using LTS OS distributions
- (Version 2.0) Provide MSS-ENTITIES with MSS-AFFILIATE information highlighting high downloads or low donation times.
- (Version 2.0) Add MAC address to the ENTITY database.
8. References


~ 77 ~
Appendix A

Data Dictionary

Aggregate – a collection of components that usually comprise a system

Bandwidth - a bit rate measure of available or consumed data communication resources expressed in Gigabits/second.

CF – Control Framework

Component – a physical computer resource, such as a router, switch, computer, phone, or copy machine

Community – MSS users identified by a GENI control framework

Contributor – an entity that donates a portion of its resources to GENI

Control framework – one of four GENI architectures used to federate computer resources

Donation – Donations include nearly all types of network resources, such as computers, network routers and switches, cell phones, computer tablets, copy machines, and so on. However, MSS is only available for computers as of this writing.

Experimenter – one who conducts network research on GENI

Federate – incorporate one’s computer resource into a GENI control framework

File list – hierarchical locations of files on a resource

GENI – Global Environment for Network Innovation

GUI – Graphical User Interface

Hard Disk – Non-volatile storage device for digital media

ISP – Internet Service Provider

Kbps – Kilobytes per second

LAMP – (Linux, Apache, MySQL, and PHP) – this is a common configuration for a database server

LSN – Large-Scale Network. GENI is an example of a large-scale network.

LTS – refers to a Long-Term Service agreement provided by many operating system developers, such as Ubuntu.

Mbps – Megabytes per second

MSS – Mutualistic Software Services
MSS-CENTER – refers to the main MSS office, where MSS is developed and distributed to participating control frameworks.

MSS-AFFILIATE – refers to a control frameworks child relationship with its MSS-ORIGIN control framework. At a minimum, MSS-AFFILIATE will contain its MSS-ORIGIN control framework and a subset of its services.

MSS-DEVELOPER – refers to a member of academia or business who submits a service to MSS.

MSS-ORIGIN – refers the control framework developer and maintainer. MSS-ORIGIN falls directly beneath MSS-CENTER in the hierarchy.

MSS-RESOURCE – a resource without a control framework, such as a physical computer or virtual machine.

MSS-USER – refers to an authorized user of a MSS-RESOURCE due to his or her working relationship with a resource owner.

Node – a computer

NSF – National Science Foundation

OS – Operating System

Programmable component – a network resource with a modifiable operational instruction set

RAM – Random Access Memory. For MSS, this refers to the computer’s volatile memory, such as Dynamic RAM (DRAM).

Real-world user – a private citizen

RENCI – Rennaissance Computing Institute

Resource – a portion or an entire physical computer hardware component. MSS is only concerned with resources such as laptops, desktops, and computer tablets (such as iPad).

Resource Owner – one who is authorized to donate all or a portion of a resource to GENI

Slice – a collection of one or more aggregates and components

Standard – the minimum hardware and software configuration required to federate into GENI

Traffic – the movement of information across the Internet

UI – User Interface

USB – universal serial bus
Appendix B

Context Diagrams

CD-1

NOTE: Usage statistics may be reported directly to MSS-ORIGIN (node 1) from MSS-AFFILIATES (node N, N+1)
Legend

S - Service
U - Usage Stats
1 - NODE 1
N - NODE N

NOTE: Usage statistics may be reported directly to MSS-ORIGIN (node 1) from MSS-AFFILIATES (node N, N+1)
System Overview

SO-1

MSS Overview

MSS-RESOURCE
- MSS-USER
- Parent Services Subset

MSS-ENTITY Hierarchy
- GENI CF
- MSS-RESOURCE
- MSS-USER data
- MSS Software
- Services Subset

MSS-CENTER
- MSS Software Full Service Repository

MSS-DEVELOPER (GENI Experimenter)

MSS-CENTER Web Site Service Content

MSS-ENTITY
- GENI CF
- MSS-RESOURCE
- MSS-USER data
- MSS Software
- Services Subset

MSS-DEVELOPER (GENI Experimenter)

Service Information

Donor

Maintains

Installs

Creates

Operates

Informs

Connects

{ Connection | Authentication | Encryption }

Fail

Heartbeats

Delivers

Owns

Retrieves

Delivers

Connects

Heartbeats

1

1

1

3

3

1

MSS Overview
Component Models

CM-1

MSS-CENTER

Full Service Repository

Encrypt
Decrypt
Validate

User Interface

CF Data

User Data

MSS-ORIGIN

Services Subset

Encrypt
Decrypt
Validate

User Interface

CF Data

User Data

MSS-AFFILIATE

Services Subset

Encrypt
Decrypt
Validate

User Interface

CF Data

User Data

MSS-RESOURCE

Laptops

Workstations

Servers

~ 83 ~
Remote Server Model
Appendix C

GENI News and Events (2011)
With the GENI project advancing smoothly into Spiral 4, many key projects were highlighted at GEC 12 during the Experiments Plenary at the twelfth GENI Engineering Conference (GEC12) in Kansas City on November 3, 2011.

Leading researchers presented live demonstrations of their experiments to two hundred ninety five attendees. Experiments built on the unique capabilities of the GENI mesoscale deployment, a prototype distributed virtual laboratory for exploring future internets, currently spanning over a dozen university campuses and backbone points of presence across the US. Using GENI's capabilities of slicing and deep programmability, experimenters were able to deploy and validate novel services and applications, many of which are not realizable in today's internet.

One group of experiments focused on taking advantage of server and cloud resources to provide new and more efficient capabilities to end users of home networks and mobile devices. Researchers from Georgia Institute of Technology demonstrated uCap, a tool that works with a specialized home router to permit home users to manage network usage allocations across family members, applications (browsing, e-mail, video streaming), and devices. The University of Wisconsin used a suite of chess-playing smart phones to show how computationally intensive tasks can be offloaded to heterogeneous cloud resources while meeting users’ goals for security and power efficiency. The Infinity project at the University of Michigan integrates energy-efficient wireless communication techniques and predictive caching to optimize performance of smart phone applications such as Facebook photo sharing.

In a collaborative effort with the SC11 SCInet Research Sandbox (SRS), another group of experiment teams highlighted novel in-network capabilities that build upon GENI’s deeply programmable network resources. Researchers from Northwestern University showed how advanced programmable networks can shortcut years of custom engineering to bring ad hoc specialized networks to individuals and organizations. Indiana University gave attendees a live view of their FlowScale system, which balances multi-gigabit network loads across the campus’ multiple intrusion detection system (IDS) servers, an integral part of the University’s network operations. A combined team from Indiana University and the University of Delaware used their eXtensible Session Protocol (XSP) to boost performance by seamlessly connecting GENI-enabled resources at the network edge to core routers running a high-performance transfer protocol. Clemson University researchers showed off their Steroid OpenFlow Service, transparently tuning network performance to increase end-to-end TCP transfer rates by two orders of magnitude.

Finally, a team led by Rutgers WINLAB showed how they are using the GENI mesoscale deployment to deploy, test, and validate their MobilityFirst architecture. MobilityFirst is sponsored by the National Science Foundation under its Future Internet Architecture program. Using new protocols and design paradigms, MobilityFirst is developing a novel architecture for a future internet where mobility is the norm, with dynamic host and network mobility at scale. Two key MobilityFirst capabilities, Storage Aware Routing and the Global Name Resolution Service, were demonstrated with wired and two different wireless connectivity modalities across a nationwide GENI slice covering eight campuses, two national network backbones, and nine backbone points of presence.

Watch GEC12 Demonstration Videos

Calendar of upcoming GECs
Volume III
Software Design for Mutualistic Software Services (MSS) Version 1.0
1. Introduction
The software design for Mutualistic Software Services incorporates key features identified in the Software Requirements Specification for Mutualistic Software Services (MSS) Version 1.0 and the Software Architecture for Mutualistic Software Services (MSS) Version 1.0. The primary goal addressed by these documents is to create a means by which the GENI Open Resource Control Architecture (ORCA) control framework (CF) users, such as developers, experimenters, and researchers, can share and track software services for use within ORCA.

The concept of sharing services in exchange for resources is hardly new. For example, Amazon and the Apple App Store have a centralized delivery system in which customers exchange resources (money) for services (applications and products). In addition, peer-to-peer networks such as LimeWire and Kazaa use a decentralized delivery system in which members exchange resources (videos and music) for resources (other videos and music). MSS is different from both of these systems in several ways.

First, MSS uses a hierarchical tree-like distribution, which flows centrally from the root at the GENI Maintenance Operations Center (GMOC), then branches through each CF origin to its affiliates, and finally flows to the resources as leaves. The second difference relates to the first in the “middle-mile” and “last-mile” of service delivery. In the middle-mile, each node in the tree is a client of its parent server, and so a server is a client of a server, is a client of a server, and so on. This means that each server must authenticate with the originating CF, and each resource and user must authenticate with his or her parent CF in order to receive services. In the last-mile, the required number of servers at each node decreases like the diameter of a tree branch from many servers at the origin to a CF residing on a single server (e.g., laptop and desktop computers) that a single individual can donate. This single server model on which all ORCA and MSS systems must reside did not exist before this project. Finally yet importantly, GENI experimenters “pay” by hosting a CF and sharing resources in the form of virtual machines, which means GENI’s basic unit of currency is the resource. If one considers the software products of these experiments to be services, then offering services to the public in exchange for donated resources provides the payment GENI requires to grow.

This first version of MSS incorporates a lightweight Linux, Apache, MySQL, and PHP (LAMP) infrastructure that ORCA owners can use to advertise and deliver services. In doing so, it addresses the fundamental, mutual need between owners and users: owners require vast amounts of resources to meet their goal of conducting at-scale Internet experiments, and users, experimenters, and researchers require software services to conduct experiments and business. MSS leverages the hierarchical GENI structure to establish a distributed service delivery system. When a business or university donates a portion of its resources to GENI as an affiliate, it receives all or part of its sponsor’s services, which organizational members can then access.
2. Document Outline

This document incorporates the design features in the *Software Requirements Specification for Mutualistic Software Services (MSS) Version 1.0* and the *Software Architecture for Mutualistic Software Services (MSS) Version 1.0*. The reader should be thoroughly familiar with these because this document refers to them often.

This document follows this outline:

1. Introduction
2. Document Outline
3. System Overview
   3.1 Functional Attributes
   3.2 Non-functional Attributes
   3.3 Components
4. System Architecture
   4.1 Databases
      4.1.1 Services Data
      4.1.2 CF Data
      4.1.3 Users Data
   4.2 Components
      4.2.1 CF Interface
      4.2.2 User Interface
      4.2.3 Services Interface
   4.3 Service Repository
      4.3.1 File System
   4.4 Remote Server
5. Detailed System Design
6. References

Appendix A: SQL Code
Appendix B: PHP and BASH code
Appendix C: Remote Server Configuration
3. System Overview
MSS Version 1.0 has three main components: MSS-ORIGIN, MSS-AFFILIATE, and MSS-RESOURCE. The actual hardware and OS used on the origin and affiliates vary, and each must support service delivery to its child resources. As of this writing, MSS will only support the remote server as a single-server MSS/CF installation, as described in the *Software Architecture for Mutualistic Software Services (MSS) Version 1.0* section 5.1 Remote Server Model.

MSS-AFFILIATE does not fully support a typical computer in the personal computer domain that a member of the public might own, or that an employee of a resource owner might use because of IP address and network interface card requirements. However, later versions of MSS will support these systems. MSS-AFFILIATE does fully support the current state of CF server clusters and its virtual machines, to include the remote server. In addition, later versions of MSS will support the Windows OS; therefore, one must design MSS with common workstation and laptop computers in mind.

3.1 Functional Attributes
The System must:

- Deliver services over an Internet connection
- Provide a management interface for MSS administrators
- Allow the administrator to choose a subset of parent services
- Allow the user to choose a service from the parent sponsor
- Check for heartbeats on MSS-ORIGIN

3.2 Non-functional Attributes
The System shall have:

- Portability
- Autonomy
- Securability
- Credibility
- Integrability
- Extensibility
- Interoperability
- Usability

3.3 Components
Refer to the *Software Architecture for Mutualistic Software Services (MSS) Version 1.0* section 5.3 Component Model for a complete listing of MSS components.
4. System Architecture
The System Architecture is based on the *Software Architecture for Mutualistic Software Services (MSS)* Version 1.0 section 5.3 Component Model Version 1.0 components. The following sections link the architectural requirement with the function or interface required to demonstrate adherence to the architectural strategy. Components are listed in font *Courier New*, with functions or classes listed in lower case and with parentheses like so, *courier_new()*. Components, functions, or classes in green already exist, such as *orca.sql*.

4.1 Databases

4.1.1 Services Data
- Holds the subset of service attribute data of its MSS-ENTITY
  - connect_Services
  - Services.sql [*Diagram DB-1 in Appendix A depicts Services.sql*]
    - Attributes
      - attribute_ID
      - attribute
      - description
      - parent_ID
    - Attributes_have_Services
      - Attributes_attribute_ID
      - Services_service_ID
    - Services
      - service_ID
      - name
      - filename
      - description
      - shasum
      - developer
      - publisher

4.1.2 CF Data
- Holds the MSS-ENTITY data for itself.
  - connect_orca
  - orca.sql
    - Actors

4.1.3 Users Data
- Holds the authorized user’s data for itself and its child MSS-AFFILIATES (actors)
  - connect_Users
  - Users.sql [*Diagram DB-2 in Appendix A depicts Users.sql*]
    - Actor

~ 91 ~
• guid
• name
• ip
• port
• private_key_loc
• mss_user
• sponsor
  ▪ Actor_has_User
    • actor_guid
    • user_user_ID
  ▪ Services
    • service_ID
    • totals
  ▪ User
    • user_ID
    • name
    • password
    • guid
    • administrator
  ▪ User_has_Services
    • User_user_ID
    • Services_service_ID

4.2 Components

4.2.1 CF Interface
• MSS-ORIGIN presents an interface to its children to receive heartbeats
  o The ORCA Actor Registry
• MSS-AFFILIATES deliver heartbeats to its MSS-ORIGIN
  o The ORCA Actor Registry
• Provides a CF interface with a list of current donors.
  o The ORCA Actor Registry
  o authorized_sponsor()
• Provides a list of available resources to GENI experimenters and users
  o The ORCA Actor Registry
• Enables GENI experimenters and users to request resources
  o The ORCA Web Portal on each CF

4.2.2 User Interface
• Uses the Authentication process to verify actor and user requests
  o System Specific Constants [constants.php]
  o authorized_user() [Index.php]
• Receives a list of MSS-ENTITY resources from the CF
  o List Sponsor
• Receives user service information and service requests
  o List Users
  o List Actors
  o Delete Users
  o Delete Actors
• Provides service delivery information to the Service Interface
  o Add Client
  o Add Sponsor
  o Add User
• Triggers the Service Interface to deliver services
  o Choose Category [for administrators]
  o Choose Service
• Checks for heartbeats on the CF interface
  o check_heartbeats()
• Triggers the Service Interface to download services
  o Assign Users [... to authorized clients]
  o authorized_actor()

4.2.3 Service Interface
• Receives a service information request from the User Interface
  o List Services
  o Delete Services
• Receives a service download request from the User Interface
  o Download Services
• Uses the Encryption process to encrypt services
  o Secure Shell (SSH)
    ▪ ssh_command()
    ▪ rsync_command()
• Uses the Decryption process to decrypt services
  o Secure Shell (SSH)
    ▪ ssh_command()
    ▪ rsync_command()
• Uses the Validation process to validate services
  o compare_shasum()
• Receives services from parent MSS-ENTITY
  o Secure Shell (SSH)
    ▪ ssh_command()
    ▪ rsync_command()
• Delivers services to child MSS-AFFILIATES
  o Secure Shell (SSH)
    ▪ ssh_command()
4.3 Service Repository

4.3.1 File System
- Holds MSS services
  - service_tree()
  - match_attributes() [directory structure must match attributes’ recursive structure]
- Holds a subset of parent services, to include the entire subset, for its MSS-ENTITY
  - Clear the Services DB for a potentially totally different subset of services
    - DELETE FROM Attributes
    - DELETE FROM Attributes_have_Services
    - DELETE FROM Services
  - Make sure there is an administrator or no one can log in
    - INSERT INTO User [...the client’s administrator]
  - Add the new Services data
    - INSERT INTO Attributes
    - INSERT INTO Attributes_have_Services
    - INSERT INTO Services

Section 5. Detailed System Design below expands the requirements above.

4.4 Remote Server

The document Software Architecture for Mutualistic Software Services (MSS) Version 1.0, 5.1 Remote Server Model, identifies the remote server as an essential element to MSS success. This is because it enables individual members of the public to donate their resources to GENI in exchange for services. Furthermore, some key components for the remote server are:

- It must reside on a single server
- It must contain all CF programs
- It must incorporate a firewall

In contrast, canonical ORCA [1] installation has at several physical computers. It has one Eucalyptus [2] head node with the program ImageProxy [3] that hosts virtual machine (VM) images, several Eucalyptus worker nodes, and an additional computer that acts as a router to control iptables by using Shorewall [4]. The ORCA cluster administrator uses Shorewall to map the public IP and a static port number to the ORCA cluster’s web portal, to the Eucalyptus web portal, and to ImageProxy, and Shorewall dynamically maps the public IP and one of an assigned block of port numbers to a Eucalyptus “Instance.” Eucalyptus typically uses a Xen or KVM image-based VM as an instance. The ORCA “cloud” centers at the main Duke University ORCA cluster and includes all other geographically separated ORCA clusters, such as the one at the University of Fairbanks, Alaska. The following describes a typical ORCA experimenter request for one instance.
An experimenter opens the ORCA web portal and requests one or more VMs, which are Eucalyptus instances in this case, from the ORCA “cloud.” ORCA main finds an ORCA cluster with available resources in the ORCA Actor Registry, and Shorewall performs Name Address Translation (NAT) between that ORCA web portal’s public IP address and port number. Then ORCA requests one instance from the Eucalyptus head node, which commands a worker node to instantiate an instance. Upon success, the worker node assigns a private IP address to the instance’s virtual interface (VIF), 10.10.10.130 for example, and requests an IP address from DHCP installed on the Eucalyptus head node. The head node assigns an address, 192.168.1.11 for example, from a pre-defined block of IP addresses, and performs NAT between 10.10.10.1 and 192.168.1.11. Eucalyptus reports the address 192.168.1.11 to ORCA, which requests Shorewall perform NAT between 192.168.1.11, the public IP address, and a port number Shorewall dynamically assigns. ORCA delivers the public IP address and port number to the experimenter’s ORCA web portal, and he or she now can access the instance using Secure Shell (SSH) from his or her remote location. In addition, ORCA uses a programmable switch to interconnect any second and subsequent VIFs that the experimenter requests so that the instances may communicate with each other across virtual local area networks (VLAN). The figure below depicts the canonical ORCA cluster.
The remote server combines all of the canonical ORCA elements into one computer, by allocating a small portion of the hard drive using the Logical Volume Manager. For instance, hypervisor uses two logical volumes, hyper-root and hyper-swap, and Domain 1 uses two logical volumes, router-root and router-swap, while the Eucalyptus instances use image-based VMs. Furthermore, Xen and KVM allow one to create VIFs 0 & 1 as an extension of the physical Network Interface Cards (NIC) 0 & 1, and Linux includes a “dummy” module to make an internal dummy interface. The picture below exhibits these components.

![Diagram](image.png)

**Figure 2. ORCA Remote Server**

One important note is that this setup requires two NICs, but some computers owned by individuals only have one. The Dell Inspiron 570 mini-tower computer serves as an example, which one can purchase for $300 as of this writing [5]. Dell includes an option to add a wireless LAN PCIe card, and this computer has expansion slots to add additional cards [6]. Moreover, many computers have two NICs installed from the factory. For instance, laptops often have one Ethernet card and one wireless LAN card. Furthermore, suitable NICs are relatively inexpensive, with prices ranging from less than $10 for a 10/100 Mbps to less than $30 for a 10/100/1000 Mbps Ethernet card at NewEgg.com [7]. However, I believe one could change the configuration to use only one Ethernet card with a little more experimentation with the configuration files. Appendix C contains all the configuration file changes necessary to create the remote server set up.
5. Detailed System Design
The detailed system design lists the sub-interface identified in 4. System Architecture above, with pseudo-code written for the requirements. In addition, the MSS System only maintains essential files in the web-facing MSS directory, and so the MSS System is partitioned like so:

```
[web server root directory]
├── MSS
│   ├── actors.jsp
│   ├── add_client.php
│   ├── add_sponsor.php
│   ├── add_user.php
│   ├── assign_users.php
│   ├── choose_category.php
│   ├── choose_service.php
│   ├── del_actors.php
│   ├── del_services.php
│   ├── del_users.php
│   ├── download_service.php
│   ├── functions_php.php
│   ├── functions_Services.php
│   ├── functions_Shell.php
│   ├── functions_User.php
│   ├── index.php
│   ├── list_actors.php
│   ├── list_services.php
│   ├── list_sponsor.php
│   ├── list_users.php
│   ├── menu.php
│   ├── mssLogo.png
│   ├── style.css
│   └── user_logout.php
└── scripts
    ├── connect_orca.php
    ├── connect_Services.php
    ├── connect_Users.php
    ├── constants.php
    ├── cookie.php
    ├── footer.php
    └── header.php
└── logs
    ├── rsync
    │   └── [YYYYMMDD]_mss_rsync.log
    ├── sh_check_heartbeats
    ├── sh_rsync_command
    └── sh_ssh_command
```
Appendix A contains the SQL Entity-Relationship model for the Services and Users databases and the full SQL code. Diagram CM-3 in Appendix B depicts the high-level system design for one MSS-ENTITY, which is a stretched and expanded version of diagram CM-1 in the Software Architecture for Mutualistic Software Services (MSS) Version 1.0. In addition, Appendix B includes the full PHP code for MSS.

<table>
<thead>
<tr>
<th>Sub-Interface</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL INTERFACES</td>
<td>All interfaces must ensure that the user is valid and that a session has begun.</td>
</tr>
</tbody>
</table>

- Start a new session
- The session expires after 20 minutes of inactivity
- Destroy the session upon exit or expiry
- Ensure the user is valid for every page or display an error
- Display a menu relevant to the user’s permissions

<table>
<thead>
<tr>
<th>add_client</th>
<th>Add a new client actor, onto which its LOCAL Administrator may download services from its sponsor.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If the user is valid and an administrator, then</td>
</tr>
<tr>
<td></td>
<td>If Submit is pressed, then</td>
</tr>
<tr>
<td></td>
<td>If the variables are not empty, then</td>
</tr>
<tr>
<td></td>
<td>Check the IP format</td>
</tr>
<tr>
<td></td>
<td>Check the Port format</td>
</tr>
<tr>
<td></td>
<td>Check Actor table to see if the Name already exists</td>
</tr>
<tr>
<td></td>
<td>Else print an error message</td>
</tr>
<tr>
<td></td>
<td>Set Sponsor to “client”</td>
</tr>
<tr>
<td></td>
<td>Give the Client a new GUID</td>
</tr>
<tr>
<td></td>
<td>INSERT name, ip, port, guid, private_key_loc, mss_user, sponsor into Actor</td>
</tr>
<tr>
<td></td>
<td>Else</td>
</tr>
<tr>
<td></td>
<td>Display Name IP PORT Private_Key_Location MSS_User_Name from Actor as textboxes</td>
</tr>
<tr>
<td></td>
<td>Else print an error message and go to user_logout</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>add_sponsor</th>
<th>Add a new sponsor actor, if the user is valid and an administrator. Sponsors host a CF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If the user is valid and an administrator, then</td>
</tr>
<tr>
<td></td>
<td>If Submit is pressed, then</td>
</tr>
<tr>
<td></td>
<td>If the variables are not empty, then</td>
</tr>
<tr>
<td></td>
<td>Check the Actor table for the GUID</td>
</tr>
<tr>
<td></td>
<td>Check the Actor table for the Name</td>
</tr>
<tr>
<td></td>
<td>Check the IP format</td>
</tr>
<tr>
<td></td>
<td>Check the Port format</td>
</tr>
<tr>
<td></td>
<td>Else print an error message</td>
</tr>
<tr>
<td></td>
<td>INSERT guid, name, ip, port, private_key_loc, mss_user, sponsor into Actor</td>
</tr>
<tr>
<td></td>
<td>Else</td>
</tr>
<tr>
<td></td>
<td>Display guid, name, ip, port, private_key_loc, mss_user from Actor as textboxes</td>
</tr>
<tr>
<td></td>
<td>Else print an error message and go to user_logout</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>add_user</th>
<th>Add a new user if the adding user is valid and an administrator. Users can be a Local Admin, Client Admin, or user</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If the user is valid and an administrator, then</td>
</tr>
<tr>
<td></td>
<td>If Submit is pressed, then</td>
</tr>
<tr>
<td></td>
<td>If variables are not empty, then</td>
</tr>
<tr>
<td></td>
<td>Check the Actor table to see if the name already exists</td>
</tr>
<tr>
<td></td>
<td>Compare password1 to password2 to check for typos</td>
</tr>
<tr>
<td></td>
<td>Set the new user to Admin, client, or user</td>
</tr>
<tr>
<td></td>
<td>Else print an error message</td>
</tr>
</tbody>
</table>

~ 98 ~
assign_users

Administrators can assign a user to an actor. This allows Local or Client Administrators to add new services to their sponsor actor, and users to download services from their sponsor onto the approved actor

If the user is valid and an administrator, then
If Submit is pressed, then
For each Actor
    INSERT the checked user into Actor_has_User
    Display a confirmation message
Else
    For each Actor
        Display GUID Name IP Port Sponsor from Actor as textboxes
        For each User
            Display Name Administrator from User as checkboxes
Else print an error message and go to user_logout

choose_category

This is the only way that a Client Administrator can add new services to his or her Client Sponsor. The user must be a Client Administrator on the parent and a Local Administrator on the child

If the user is valid and on an authorized sponsor, then
    If the user is a local or client administrator and sending heartbeats to MSS-ORIGIN, then ON THE CLIENT
        If Submit is pressed, then
            If the user does not exist [special case: first install]
                INSERT user_ID, name, password, guid, administrator into User
            Clear the Attributes Attributes_have_Services and Services tables
            INSERT all Attributes into Attributes
            For each checked Attribute
                Make directories on the client that match the attribute tree in the System Constants location
                Rsync the entire directory named [Attribute]
                INSERT the matching Service
                INSERT the matching Attributes_have_Services
            Else
                For each Attribute
                    Display the Attribute indented by its level
                For each Attributes_have_Services
                    Display a checkbox
            Elself print an error message
Else print an error message and go to user_logout

choose_service

All users can choose services if they are valid users on an authorized computer

If the user is valid and on an authorized client, then
    If the Service_ID button is pressed
        If sending MSS-ORIGIN heartbeats
            Display alert for download size
            Go to download_service
    Else
        Display the attribute tree by its level
        For each Attribute
            Display Service_ID as Submit button
            Display the list of matching Services

~ 99 ~
Else print an error message and go to user_logout

<table>
<thead>
<tr>
<th>constants</th>
<th>Allow the administrator to tailor the installation by providing definable Constants. Separate database names, passwords, and file locations from the PHP web pages.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSS_SERVICES directory</td>
<td>GENI_HEARTBEATS webpage</td>
</tr>
<tr>
<td>GENI_DB for GENI database name</td>
<td>GENI_DB_USER for GENI database access</td>
</tr>
<tr>
<td>GENI_DB_PASS for GENI database password</td>
<td>USERS_DB for Users database name</td>
</tr>
<tr>
<td>USERS_DB_USER for Users database access</td>
<td>USERS_DB_PASS for Users database password</td>
</tr>
<tr>
<td>SERVICES_DB for Services database name</td>
<td>SERVICES_DB_USER for Services database access</td>
</tr>
<tr>
<td>SERVICES_DB_PASS for Services database password</td>
<td>MSS_USER that conducts MSS activities on the host only</td>
</tr>
<tr>
<td>MSS_HOME the directory that contains MSS webpages</td>
<td>MSS_SCRIPTS the directory that contains MSS scripts</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>del_actors</th>
<th>Administrators can delete a client from the sponsor</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the user is valid and a local administrator, then</td>
<td></td>
</tr>
<tr>
<td>If the Submit button is pressed, then</td>
<td></td>
</tr>
<tr>
<td>DELETE the Actor from Actor_has_User where the checkbox is checked</td>
<td></td>
</tr>
<tr>
<td>DELETE the Actor from Actor where the checkbox is checked</td>
<td></td>
</tr>
<tr>
<td>Else</td>
<td></td>
</tr>
<tr>
<td>Display “delete actor” checkbox, name, guid, ip, port, private_key_loc, mss_user, sponsor from Actor</td>
<td></td>
</tr>
<tr>
<td>Else print an error message and go to user_logout</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>del_services</th>
<th>Administrators can choose to delete services from the sponsor immediately. For everyone, only delete the service after every user deletes the service to save bandwidth and download time.</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the user is valid, then</td>
<td></td>
</tr>
<tr>
<td>If the Submit button is pressed, then</td>
<td></td>
</tr>
<tr>
<td>DELETE the service from User_has_Services where the “delete my service” checkbox is checked</td>
<td></td>
</tr>
<tr>
<td>Decrement Services Totals</td>
<td></td>
</tr>
<tr>
<td>If this is the last user to own the service, then</td>
<td></td>
</tr>
<tr>
<td>DELETE the service from Services where the “delete my service” checkbox is checked</td>
<td></td>
</tr>
<tr>
<td>Delete the service from the client’s file system</td>
<td></td>
</tr>
<tr>
<td>If the user is a local administrator, then</td>
<td></td>
</tr>
<tr>
<td>DELETE the service from User_has_Services where the “delete all service” checkbox is checked</td>
<td></td>
</tr>
<tr>
<td>DELETE the service from Services where the “delete all service” checkbox is checked</td>
<td></td>
</tr>
<tr>
<td>Delete the service from the client’s file system</td>
<td></td>
</tr>
<tr>
<td>Else</td>
<td></td>
</tr>
<tr>
<td>Display “delete my service” checkbox</td>
<td></td>
</tr>
<tr>
<td>If the user is a local administrator, then</td>
<td></td>
</tr>
<tr>
<td>Display “delete all service” checkbox</td>
<td></td>
</tr>
<tr>
<td>Display Name, Attribute, Filename, description from view_Attributes_Services</td>
<td></td>
</tr>
<tr>
<td>Else print an error message and go to user_logout</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>del_users</th>
<th>Administrators can delete a user from the sponsor</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the user is valid and a local administrator, then</td>
<td></td>
</tr>
<tr>
<td>If the Submit button is pressed, then</td>
<td></td>
</tr>
<tr>
<td>DELETE the user from User_has_Services where the checkbox is checked</td>
<td></td>
</tr>
<tr>
<td>DELETE the user from Actor_has_User where the checkbox is checked</td>
<td></td>
</tr>
<tr>
<td>DELETE the user from User where the checkbox is checked</td>
<td></td>
</tr>
</tbody>
</table>
Else
Display "delete user" checkbox, user_ID, name, guid, "local or client" administrator from User
Else print an error message and go to user_logout

download_service
Download the service if all of the choose_service requirements are met, and if the client and host IP addresses match an authorized actor.

If the user is valid and on an authorized client, then
  If the host IP exists and the client IP exists in Actors
    Send the service to the client
    Compare the SHA sums of the original to the downloaded service
    If the shasums do not match
      Delete the downloaded service from the client
      Print an error message
  Else print an error message
Else print an error message and go to user_logout

index
This is the home page for MSS, where authorized users on authorized clients log in. Administrators can log in from any computer, which also solves the “initial installation” problem where no users are assigned to any actors.

If username and password are blank, then
  If Submit button is pressed, then
    If username and password match a user in Users
      valid_user == true
    If the user is a local administrator, then
      authorized_user == true
      Load a list of services into the session
      Display administrator instructions
    If Actor_has_User where Actor.ip == client ip
      authorized_user == true
      Load a list of services into the session
      If user is a client administrator
        Display client administrator instructions
      If user is a ordinary user
        Display user instructions
    Else
      Else print an error message and deny access
Else
  Display username and password as textboxes
Else print an error message and go to user_logout

list_actors
Valid and authorized users may view a list of actors authorized to download from the sponsor.

If the user is valid and authorized, then
  Display name, guid, ip, port, private_key_loc, mss_user from User
  Display "local or client" sponsor based on host IP
Else print an error message and go to user_logout

list_services
Valid and authorized users may view a list of services available on the sponsor.

If the user is valid and authorized, then
  Display service_ID, name, filename, description, shasum, developer, publisher from Services
Else print an error message and go to user_logout

list_sponsor
Valid and authorized users may view a list of sponsors, such as the ORCA actors Service Manager, Broker, and Site, that are registered in the GENI database

If the user is valid and is authorized, then
  Display act_id, act_name, act_guid from GENI_DB.Actors
Else print an error message and go to user_logout

list_users
Valid users and administrators may view a list of users that are registered on the sponsor

If the user is valid and an administrator, then
  Display user_ID, name, guid, administrator from User
<table>
<thead>
<tr>
<th>menu</th>
<th>Display a menu ribbon that changes based on the user type: administrator, client administrator, user</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If the user is valid and authorized, then</td>
</tr>
<tr>
<td></td>
<td>Display index, list_sponsor, list_services, choose_services</td>
</tr>
<tr>
<td></td>
<td>If the user is a local or client administrator, then</td>
</tr>
<tr>
<td></td>
<td>Display choose_category</td>
</tr>
<tr>
<td></td>
<td>If the actor is an authorized sponsor, then</td>
</tr>
<tr>
<td></td>
<td>Display choose_category</td>
</tr>
<tr>
<td></td>
<td>If the actor is an administrator, then</td>
</tr>
<tr>
<td></td>
<td>Display list_users, list_actors, add_user, add_client, add_sponsor</td>
</tr>
<tr>
<td></td>
<td>If the web browser is not text-based</td>
</tr>
<tr>
<td></td>
<td>Display assign_users</td>
</tr>
<tr>
<td></td>
<td>Display del_actors, del_users</td>
</tr>
<tr>
<td></td>
<td>Display del_services, user_logout</td>
</tr>
<tr>
<td></td>
<td>Else print an error message and go to user_logout</td>
</tr>
<tr>
<td>user_logout</td>
<td>Log the user out of the session when the link “Log Out” is chosen, time expires, or the user is not valid.</td>
</tr>
<tr>
<td></td>
<td>Close the session write</td>
</tr>
<tr>
<td></td>
<td>Unset the session</td>
</tr>
<tr>
<td></td>
<td>Destroy the cookie</td>
</tr>
</tbody>
</table>
6. References


Appendix A

Services Database

Overview

DB-1

SQL Code

1. SET @OLD_UNIQUE_CHECKS = @@UNIQUE_CHECKS, UNIQUE_CHECKS = 0;
2. SET @OLD_FOREIGN_KEY_CHECKS = @@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS = 0;
3. SET @OLD_SQL_MODE = @@SQL_MODE, SQL_MODE = 'TRADITIONAL';
4. DROP SCHEMA IF EXISTS `Services` ;
5. CREATE SCHEMA IF NOT EXISTS `Services` DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci ;
6. USE `Services` ;
7. -- Table `Services`.'Services'
8. -- ---------------------------------------------------------------
9. DROP TABLE IF EXISTS `Services`.'Services' ;
10. CREATE TABLE IF NOT EXISTS `Services`.'Services' ( 
11.   `service_ID` INT(11) NOT NULL ,
12.   `name` VARCHAR(100) NULL DEFAULT NULL ,
13.   `filename` VARCHAR(100) NULL DEFAULT NULL ,
14.   `description` VARCHAR(1000) NULL DEFAULT NULL ,
15.   `shasum` VARCHAR(40) NULL DEFAULT NULL ,
16.   `developer` VARCHAR(100) NULL DEFAULT NULL ,
17.   `publisher` VARCHAR(100) NULL DEFAULT NULL ,
18.   `UNIQUE INDEX `Service_ID_UNIQUE` (`service_ID` ASC) }
19. ENGINE = InnoDB
20. AUTO_INCREMENT = 2
21. DEFAULT CHARACTER SET = utf8
22. COLLATE = utf8_general_ci
23. COMMENT = 'This table holds the MSS Service Repository.';
24. -- Table `Services`.'Attributes'
25. -- ---------------------------------------------------------------
26. DROP TABLE IF EXISTS `Services`.'Attributes' ;
27. CREATE TABLE IF NOT EXISTS `Services`.'Attributes' ( 
28.   `attribute_ID` INT(11) NOT NULL ,
29.   `attribute` VARCHAR(100) NULL DEFAULT NULL ,
30.   `description` VARCHAR(1000) NULL DEFAULT NULL ,
31.   `parent_ID` INT(11) ,
32.   `UNIQUE INDEX `attribute_ID_UNIQUE` (`attribute_ID` ASC) }
33. ENGINE = InnoDB
34. AUTO_INCREMENT = 2
35. DEFAULT CHARACTER SET = utf8
36. COLLATE = utf8_general_ci
37. COMMENT = 'This table holds the MSS Service Repository.';
`attribute_ID` INT(11) NOT NULL,
`attribute` VARCHAR(100) NOT NULL,
`description` VARCHAR(1000) NOT NULL,
`parent_ID` INT(11) NULL,
UNIQUE INDEX `attribute_ID_UNIQUE` (`attribute_ID` ASC),
INDEX `parent_ID` (`parent_ID` ASC),
CONSTRAINT `parent_ID`
FOREIGN KEY (`parent_ID`)
REFERENCES `Services`(`attribute_ID`)
ON DELETE CASCADE
ON UPDATE CASCADE)
ENGINE = InnoDB
AUTO_INCREMENT = 10
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci
COMMENT = 'This table holds the MSS Attribute Repository.';

-- Table `Services`.'Attributes_have_Services`
 CREATE TABLE IF NOT EXISTS `Services`.'Attributes_have_Services` (  `attribute_ID` INT(11) NOT NULL ,  `Services_service_ID` INT(11) NOT NULL ,  PRIMARY KEY (`attribute_ID`, `Services_service_ID`) ,  INDEX `fk_Attributes_have_Services_Services1` (`Services_service_ID` ASC) ,  CONSTRAINT `fk_Attributes_have_Services_Services1`
FOREIGN KEY (`attribute_ID`)
REFERENCES `Services`(`attribute_ID`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
 CONSTRAINT `fk_Attributes_have_Services_Services1`
FOREIGN KEY (`Services_service_ID`)
REFERENCES `Services`(`service_ID`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

-- Placeholder table for view `Services`.'view_Attributes_Services`
 CREATE TABLE IF NOT EXISTS `Services`.'view_Attributes_Services` (`service_ID` INT, `name` INT, `filename` INT, `description` INT, `shasum` INT, `developer` INT, `publisher` INT, `attribute_ID` INT, `attribute` INT);  DROP VIEW IF EXISTS `Services`.'view_Attributes_Services`;
 DROP TABLE IF EXISTS `Services`.'view_Attributes_Services`;
 USE `Services`;
 CREATE OR REPLACE VIEW `Services`.'view_Attributes_Services` AS
SELECT `Services`.'service_ID', `Services`.'name', `Services`.'filename', `Services`.'description', `Services`.'shasum', `Services`.'developer', `Services`.'publisher', `Attributes`.'attribute_ID', `Attributes`.'attribute'
FROM `Services`
INNER JOIN (`Attributes` INNER JOIN `Attributes_have_Services` ON `Attributes`.`attribute_ID`=`Attributes_have_Services`.`Attributes_attribute_ID`) ON `Services`.`service_ID`=`Attributes_have_Services`.`Services_service_ID`;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
Users Database

Overview

DB-2

SQL Code

1  SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
2  SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
3  SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';
4
5  DROP SCHEMA IF EXISTS `Users`;
6  CREATE SCHEMA IF NOT EXISTS `Users` DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci ;
7  USE `Users` ;
8
9  -- Table `Users`.`User`
10  -- ---------------------------------
11  DROP TABLE IF EXISTS `Users`.`User` ;
12
13  CREATE TABLE IF NOT EXISTS `Users`.`User` ( 
14    `user_ID` INT NOT NULL AUTO_INCREMENT ,
15    `name` VARCHAR(50) NOT NULL ,
16    `password` VARCHAR(10) NOT NULL ,
17    `guid` VARCHAR(45) NOT NULL ,
18    `administrator` TINYINT(1) NOT NULL DEFAULT 0 ,
19    PRIMARY KEY (`user_ID`) ,
20    UNIQUE INDEX `user_ID_UNIQUE` (`user_ID` ASC ) ,
21    UNIQUE INDEX `guid_UNIQUE` (`guid` ASC ) )
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci
COMMENT = 'Contains the user GUID, password, name, and ssh_key_location';

-- Table 'Users`.`Actor`
-------------------------------------
DROP TABLE IF EXISTS `Users`.`Actor` ;
CREATE TABLE IF NOT EXISTS `Users`.`Actor` ( 
  `guid` VARCHAR(45) NOT NULL,
  `name` VARCHAR(45) NOT NULL,
  `ip` VARCHAR(15) NOT NULL,
  `port` INT NOT NULL DEFAULT 22,
  `private_key_loc` VARCHAR(500) NOT NULL,
  `mss_user` VARCHAR(50) NOT NULL,
  `sponsor` TINYINT(1) NOT NULL DEFAULT 0,
  PRIMARY KEY (`guid`),
  UNIQUE INDEX `guid_UNIQUE` (`guid` ASC)
) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8 COLLATE = utf8_general_ci;
COMMENT = 'The GENI actor that owns the resource the user is using.';

-- Table 'Users`.`Services`
-------------------------------------
DROP TABLE IF EXISTS `Users`.`Services` ;
CREATE TABLE IF NOT EXISTS `Users`.`Services` ( 
  `service_ID` INT NOT NULL,
  `totals` INT NULL DEFAULT 0,
  PRIMARY KEY (`service_ID`) )
ENGINE = InnoDB;

-- Table 'Users`.`User_has_Services`
-------------------------------------
DROP TABLE IF EXISTS `Users`.`User_has_Services` ;
CREATE TABLE IF NOT EXISTS `Users`.`User_has_Services` ( 
  `User_user_ID` INT NOT NULL,
  `Services_service_ID` INT NOT NULL,
  PRIMARY KEY (`User_user_ID`, `Services_service_ID`),
  INDEX `fk_User_has_Services_Services1` (`Services_service_ID` ASC),
  CONSTRAINT `fk_User_has_Services_User1` FOREIGN KEY (`User_user_ID`)
    REFERENCES `Users`.`User` (`user_ID` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_User_has_Services_Services1` FOREIGN KEY (`Services_service_ID`)
    REFERENCES `Users`.`Services` (`service_ID` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB DEFAULT CHARACTER SET = utf8 COLLATE = utf8_general_ci;
DROP TABLE IF EXISTS `Users`.`Actor_has_User` ;

CREATE TABLE IF NOT EXISTS `Users`.`Actor_has_User` (  
  `Actor_guid` VARCHAR(45) NOT NULL ,  
  `User_user_ID` INT NOT NULL ,  
  PRIMARY KEY (`Actor_guid`, `User_user_ID`) ,  
  INDEX `fk_Actor_has_User_User1` (`User_user_ID` ASC) ,  
  CONSTRAINT `fk_Actor_has_User_Entity1`  
    FOREIGN KEY (`Actor_guid`)  
    REFERENCES `Users`.`Actor` (`guid`)  
    ON DELETE NO ACTION ,  
  CONSTRAINT `fk_Actor_has_User_User1`  
    FOREIGN KEY (`User_user_ID`)  
    REFERENCES `Users`.`User` (`user_ID`)  
    ON DELETE NO ACTION  
) ENGINE = InnoDB  
DEFAULT CHARACTER SET = utf8  
COLLATE = utf8_general_ci ;

SET SQL_MODE = @OLD_SQL_MODE ;
SET FOREIGN_KEY_CHECKS = @OLD_FOREIGN_KEY_CHECKS ;
SET UNIQUE_CHECKS = @OLD_UNIQUE_CHECKS ;
Appendix B

High-Level System Design

CM-3

The ORCA Actor Registry

System Specific Constants

Service Data

CF Data

CF

User Interface Service Interface

Authenticate

User Data

Connect Users

Actor

guid
name
ip
port
private_key_loc
mss_user
sponsor
Actor_has_User
actor_guid
user_user_ID
Services
service_ID
totals
User
user_user_ID
name
password
guid
administrator
User_has_Services
User_user_ID
Services_service_ID

authorized_actor()

authorized_sponsor()

authorized_user()

check_heartbeats()

ssh_command()

rsync_command()

match_attributes()

inserts & deletes

connect_orca

connect_orca

match_attributes()

authorized_actor()
PHP Code

add_client.php

```php
<?php
//Purpose: Add a new client actor
//Means: Connect to the Users database as user "workers"
//Conventions: na stands for "new actor"
//Author: John P. Quan
//Version: 1.0
//Date: 20120105
?>

<?
session_cache_expire(20);
sesssion_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");

        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $_SESSION;
        session_destroy();
    }
}

//Set the session start time
$_SESSION['start'] = time();

//////// START OF PAGE /////////////

if ($_SESSION['valid_user'] == true AND $_SESSION['authorized_user'] == true AND $_SESSION['administrator'] == "1") {
    ?>
    <?php
    include '../scripts/connect_Users.php';
    include '../scripts/header.php';
    include 'functions_User.php';
    include 'functions_php.php';
    connect_Users();
    echo "<h3><center>Add Client</center></h3>";
    //Display the menu ribbon
    include 'menu.php';
    //adjusts the header
    echo "<BR>";
    ?>
</div id="wrap">

<? // Check the posted variables after "Submit"
if (isset($_POST['na_submit'])) {
    //if a new user name is entered...
```
if(!empty($_POST['na_name']) and
!empty($_POST['na_ip']) and
!empty($_POST['na_port']) and
!empty($_POST['na_private_key_loc']) and
!empty($_POST['na_mss_user']))
{
    // add a GUID
    $na_guid = gen_uuid();
    // see if the user already exists
    $query=sprintf("SELECT * FROM Actor");
    $result = mysql_query($query);
    while($row = mysql_fetch_array($result))
    {
        //Check the name
        if ($_POST['na_name'] == $row['name'])
        {
            echo "<BR>Actor name ",$_POST['na_name']," already exists.<BR>";
            unset_na-vars();
            die(mysql_error());
        }
        else
        {
            echo "<BR>Please enter your Client's Name.<BR>";
            unset_na-vars();
            die(mysql_error());
        }
    }
    //Check the IP address format and port
    // and load the private key and mss_user
    if(filter_var($_POST['na_ip'], FILTER_VALIDATE_IP))
    {
        $na_ip = $_POST['na_ip'];
        if(isset( $_POST['na_port']) and
            $_POST['na_port'] >= 0 and
            $_POST['na_port'] < 65536) {
            $na_port = $_POST['na_port'];
        }
        else {
            //default port
            $na_port = 22;
        }
        //Individual checks for na_private_key_loc,
        // mss_user, and name
        if(!empty($_POST['na_private_key_loc'])) {
            $na_private_key_loc = $_POST['na_private_key_loc'];
        }
        else {
            echo "<BR>Please enter your Client's Private Key.<BR>";
unset_na_vars();

die(mysql_error());

if(!empty($_POST['na_mss_user'])) {
    $na_mss_user = $_POST['na_mss_user'];
}
else {
    echo "<BR>Please enter your Client's MSS User.<BR>"
    unset_na_vars();
    die(mysql_error());
}

else {
    echo "<BR>The IP address for ".$_POST['na_name']." appears to be invalid.<BR>
    unset_na_vars();
    die(mysql_error());
}

//INSERT all of the values into User db
$query = mysql_query("INSERT INTO Actor (guid, name, ip, port, private_key_loc, mss_user) VALUES('".$na_guid', '".$na_name', '".$na_ip', '".$na_port', '".$na_private_key_loc', '".$na_mss_user') ");

or die(mysql_error() . "\nQuery1: ". $query);

echo "New CLIENT ".$_na_name." inserted.<HR><pre>
<b>GUID:</b> 
".$na_guid."<BR>
<b>IP and Port:</b> 
".$na_ip":".$na_port."<BR>
<b>Private Key:</b> 
".$na_private_key_loc."<BR>
<b>MSS User:</b> ".$na_mss_user."<HR>

unset_na_vars();
mysql_close(connect_Users());

else {
    echo "<BR> Please fill out the form completely.";
    unset_na_vars();
    mysql_close(connect_Users());
}

// Set the variables to null and close the connection
unset_na_vars();
mysql_close(connect_Users());

else {
    //Display the texboxes in table form
    ?>
Insert the new CLIENT data:

```html
<form method="post" action="add_client.php">
  <tr>
    <th align="right">Name</th>
    <td align="left">
      <input type="text" name="na_name" size="40"></td>
  </tr>
  <tr>
    <th align="right">IP Address</th>
    <td align="left">
      <input type="text" name="na_ip" size="17"></td>
  </tr>
  <tr>
    <th align="right">Port</th>
    <td align="left">
      <input type="text" name="na_port" size="5">(default: 22)</td>
  </tr>
  <tr>
    <th align="right">Private Key Location</th>
    <td align="left">
      <input type="text" name="na_private_key_loc" size="50"></td>
  </tr>
  <tr>
    <th align="right">MSS User Name</th>
    <td align="left">
      <input type="text" name="na_mss_user" size="20"></td>
  </tr>
  <tr>
    <th><? //Placeholder ?></th>
    <td align="left">
      <input name="na_submit" type="Submit" value="Submit"></td>
  </tr>
</form>
```

```html
<div id="foot">
  <? include('../scripts/footer.php'); ?>
</div>
```
```php
<?php

///////// END OF PAGE ///////////

else {
    // time expired or access denied; log in again
    include ('../scripts/header.php');
    ?>

    Either you are not allowed to access this page, or your session has expired.
    Please <A href="index.php">log in</a> again.

<?php

```
add_sponsor.php

```php
<?php

//Purpose: Add a new sponsor actor
//Means: Connect to the Users database as user "workers"
//Conventions: ne stands for "new actor"
//Author: John P. Quan
//Version: 1.0
//Date: 20120105
?>

session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        $_SESSION = session_destroy();
    }
}

//Set the session start time
$_SESSION['start'] = time();

////////// START OF PAGE //////////

if ($_SESSION['valid_user'] == true AND $_SESSION['administrator'] == 1) {

    include '../scripts/connect_Users.php';
    include '../scripts/header.php';
    include 'functions_User.php';

    connect_Users();
    echo "<h3><center>Add Sponsor</center></h3>";
    //Display the menu ribbon
    include 'menu.php';
    //adjusts the header
    echo "<BR>";
    
    echo "</center><center>Add Sponsor</center></h3>";
    //Display the menu ribbon
    include 'menu.php';
    //adjusts the header
    echo "<BR>";
}

<?php

// Check the posted variables after "Submit"
if (isset($_POST['na_submit'])) {
    //Set the actor as a sponsor
    $na_sponsor = 1;
    //if a new user name is entered...
    if (!empty($_POST['na_guid']) and
        !empty($_POST['na_name']) and
```
```php
!empty($_POST['na_ip']) and
!empty($_POST['na_port']) and
!empty($_POST['na_private_key_loc']) and
!empty($_POST['na_mss_user'])
{
    // see if the user already exists
    $query = sprintf("SELECT *
                      FROM Actor");
    $result = mysql_query($query);
    while($row = mysql_fetch_array($result))
    {
        // Check the GUID
        if($_POST['na_guid'] == $row['guid'])
        {
            echo "<BR>Actor GUID ".$_POST['na_guid'].
                 " already exists.<BR>"
            unset_na_vars();
            die(mysql_error());
        }
        else
        {
            if(!empty($_POST['na_guid'])) {
                $na_guid = $_SESSION['na_guid'];
            } else
            {
                echo "<BR>Please enter your Sponsor's GUID.<BR>
                unset_na_vars();
                die(mysql_error());
            }
        }
        // Check the name
        if($_POST['na_name'] == $row['name'])
        {
            echo "<BR>Actor name ".$_POST['na_name'].
                   " already exists.<BR>"
            unset_na_vars();
            die(mysql_error());
        }
        else
        {
            if(!empty($_POST['na_name'])) {
                $na_name = $_POST['na_name'];
            } else
            {
                echo "<BR>Please enter your Sponsor's
                   Name.<BR>"
                unset_na_vars();
                die(mysql_error());
            }
        }
    }
    // Check the IP address format and port
    // and load the private key and mss_user
    if(filter_var($_POST['na_ip'], FILTER_VALIDATE_IP))
    {
        $na_ip = $_POST['na_ip'];
    }
```
if(isset( $_POST['na_port']) and
    $_POST['na_port'] >= 0 and
    $_POST['na_port'] < 65536) {
    $na_port = $_POST['na_port'];
} else {
    $na_port = 22;
} //Individual checks for na_private_key_loc, mss_user, and name
if(!empty($_POST['na_private_key_loc'])) {
    $na_private_key_loc = $_POST['na_private_key_loc'];
} else {
    echo "<BR>Please enter your Sponsor's Private Key.<BR>";
    unset_na_vars();
    die(mysql_error());
}
if(!empty($_POST['na_mss_user'])) {
    $na_mss_user = $_POST['na_mss_user'];
} else {
    echo "<BR>Please enter your Sponsor's MSS User.<BR>";
    unset_na_vars();
    die(mysql_error());
}
//INSERT all of the values into User db
$query = mysql_query("INSERT INTO Actor (guid, name, ip, port, private_key_loc, mss_user, sponsor)
    VALUES('$_na_guid', '$_na_name', '$_na_ip', '$_na_port', '$_na_private_key_loc', '$_na_mss_user', '$_na_sponsor') ")
or die(mysql_error());  "\nQuery2: "$ . $query);

echo "New SPONSOR ",$_na_name." inserted."<HR><pre>
<b>GUID: </b>"._na_guid."<BR>
<b>IP and Port: </b>"._na_ip:"._na_port."<BR>
<b>Private Key: </b>"._na_private_key_loc."<BR>
<b>MSS User:</b> ".$na_mss_user."</pre>

```
unset_na_vars()
mysql_close(connect_Users());

else
{
    echo "<br> Please fill out the form completely.";
    unset_na_vars();
    mysql_close(connect_Users());
}

// Set the variables to null and close the connection
unset_na_vars();
mysql_close(connect_Users());

} else {

    // Display the textboxes in table form

    <table style="margin-left:25px; table-layout: auto;"
        border="0"
        cellspacing="10"
        cellpadding="1">

        Insert the new SPONSOR data.  The Sponsor's GUID is
        <font color="red">REQUIRED</font>:

        <form method="post" action="add_sponsor.php">

            <tr>
                <th align="right">GUID</th>
                <td align="left">
                    <input type="text"
                        name="na_guid"
                        size="50"></td>
            </tr>

            <tr>
                <th align="right">Name</th>
                <td align="left">
                    <input type="text"
                        name="na_name"
                        size="40"></td>
            </tr>

            <tr>
                <th align="right">IP Address</th>
                <td align="left">
                    <input type="text"
                        name="na_ip"
                        size="17"></td>
            </tr>

            <tr>
                <th align="right">Port</th>
                <td align="left">
                    <input type="text"
                        name="na_port"
                        size="5"> (default: 22)</td>
            </tr>

            <tr>
                <th align="right">Private Key Location</th>
                <td align="left">
                    <input type="text"
                        name="na_private_key_loc"
                        size="50"></td>
            </tr>

        </form>
    </table>
```
<tr>
  <th align="right">MSS User Name</th>
  <td align="left">
    <input type="text" name="na_mss_user" size="20"></td>
</tr>
<tr>
  <th><i>/* Placeholder */</i></th>
  <td align="left">
    <input name="na_submit" type="Submit" value="Submit"></td>
</tr>
</form>
<div id="foot">
  <?php
  echo "</table>";
  echo "</div>
  include('../scripts/footer.php');
  echo "</div>";
  ?>
</div>

<?php
  //////// END OF PAGE ///////////////
  }
else {
  //time expired or access denied; log in again
  include ('../scripts/header.php');
  ?>
  Either you are not allowed to access this page, or your session has expired.
  Please <A href="index.php">log in</A> again.
  <?php
  ?>
  } ??>
add_user.php

```php
<?php
//Purpose: Add a new user or administrator
//Means: Connect to the Users database as user "workers"
//Conventions: nu stands for "new user"
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105

session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $_SESSION;
        session_destroy();
    }
}
//Set the session start time
$_SESSION['start'] = time();

//////// START OF PAGE //////////
if ($_SESSION['valid_user'] == true
    AND $_SESSION['administrator'] == 1) {
    ?><
include '../scripts/connect_Users.php';
include '../scripts/header.php';
include 'functions_User.php';
include 'functions_php.php';
connect_Users();

echo "<h3><center>Add User</center></h3>";

//Display the menu ribbon
include 'menu.php';
//adjusts the header
echo "<BR>";

<?
</div id="wrap">
```

```php
//Check the user name and password after "Submit"
if (!empty($_POST['nu_submit']))
{
    //if a new user name is entered...
    if (!empty($_POST['nu_name']))
    {
        // see if the user already exists
        $query=sprintf("SELECT name
```

```
```
FROM User
WHERE name = %s'',
    mysql_real_escape_string($_SESSION['nu_name']));

$result = mysql_query($query);
$row = mysql_fetch_array($result);

if($row["name"])
{
    echo "<BR>User ".$_POST['nu_name']." already exists.<BR>";
    null_nu_vars();
    die(mysql_error());
}
else
{
    $nu_name = $_SESSION['nu_name'];
}

if
{
    echo "<BR>You must include a user name.<BR>";
    null_nu_vars();
    die(mysql_error());
}

// make sure the admin puts in the same password,
// or or do not add the user
if ($_POST['nu_password1'] != $_POST['nu_password2'])
{
    echo "<BR>Password entries do not match!";
    null_nu_vars();
    die(mysql_error());
}
else
{
    $nu_password = $_SESSION['nu_password1'];
}

// add a GUID
$nu_guid = gen_uuid();

// set the administrator/user field
if ($_SESSION['nu_client_administrator'] == 'on')
{
    $nu_administrator = 2;
}
elseif ($_SESSION['nu_local_administrator'] == 'on')
{
    $nu_administrator = 1;
}
else
{
    $nu_administrator = 0;
}

// INSERT all of the values into User db
$query = mysql_query("INSERT INTO User (name, password, guid, administrator) VALUES($nu_name', '$nu_password','

~ 122 ~
126  'nu_guid',
127  'nu_administrator')
128  or die(mysql_error() . "\n Query3: " . $query);
129  // Display upon success
130  if ($nu_administrator == 1)
131  {
132      echo "<BR>New ADMINISTRATOR ".$nu_name." Inserted."
133  }
134  else
135  {
136      echo "<BR>New USER ".$nu_name." Inserted."
137  }
138  // Set the variables to null and close the connection
139  null_nu_vars();
140  mysql_close(connect_Users());
141  }
142  else
143  {
144      // Display the textboxes in table form
145  ?><table style="margin-left:25px; table-layout: auto;"
146  border="0"
147  cellspacing="10"
148  cellspacing="1" >
149  <tr>
150  <td align="right">Name</td>
151  <td align="left">
152      <input type="text"
153      name="nu_name"
154      size="20"/></td>
155  </tr>
156  <tr>
157  <td align="right">Password</td>
158  <td align="left">
159      <input type="password"
160      name="nu_password1"
161      size="20"/></td>
162  </tr>
163  <tr>
164  <td align="right">Enter Password Again</td>
165  <td align="left">
166      <input type="password"
167      name="nu_password2"
168      size="20"/></td>
169  </tr>
170  <tr>
171  <td align="right">Local Administrator</td>
172  <td align="left">
173      <input type="checkbox"
174      name="nu_local_administrator"></td>
175  </tr>
176  <tr>
177  <td align="right">Client Administrator</td>
178  <td align="left">
179      <input type="checkbox"
180      name="nu_client_administrator"></td>
181  </tr>
182  <tr>
183  <td><? // Placeholder ?></td>
<td align="left">
<input name="nu_submit" type="Submit" value="Submit"></td>
</tr>
</form>

</div>

<?
echo "</table>

echo "</div>

include('../.scripts/footer.php');
echo "</div>

<?php

///////// END OF PAGE ///////////////

<?php
else
{
//time expired or access denied; log in again
include ('../.scripts/header.php');
?
Either you are not allowed to access this page, or your session has expired.
Please <A href="index.php">log in</a> again.

<?php

~ 124 ~
assign_users.php

```php
<?php

//Purpose: Grant/Deny user download ability from an Actor.
//Means: Connect to the Actor, Actor_has_User, and User dbs.
// One must assign the user to a client and a sponsor because
// MSS checks for both by comparing session data with the databases.
//Conventions: au stands for "assign user"
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105
?>

<?

//////////////////////////////////////////////////////
//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $_SESSION;
        session_destroy();
    }
}

//Set the session start time
$_SESSION['start'] = time();

///// START OF PAGE /////////////

if ($_SESSION['valid_user'] == true
    AND $_SESSION['administrator'] == 1) {
    ?><

include '../scripts/connect_Users.php';
include '../scripts/header.php';
include 'functions_User.php';
connect_Users();

echo "<h3><center>Assign Users</center></h3>";
//Display the menu ribbon
include 'menu.php';
//adjusts the header
echo "<BR>";
?>

<?

<div id="wrap">

<?

// Insert user assignments after "Submit"
if (isset($_POST['au_submit'])) {
    //First, delete the current Actor/User relationships
    $del= mysql_query("DELETE FROM Actor_has_User")
    or die(mysql_error() . "\n Query4: ": $del);
```
 //Relate each checked user to an actor
foreach ($_POST['au_assign'] as $value) {
    //Parse the "$Actor_guid." "$User_user_ID" string
    // into array $var
    $var = explode(" ", $value);
    //Insert the Actor_guid, user_ID into Actor_has_User
    $ins = mysql_query("INSERT INTO Actor_has_User (Actor_guid, User_user_ID) VALUES('".$var[0]."', '".$var[1]."') ");
    //null the au variables
    unset_au_vars();
} //null the au variables
else
{
Assign a user to his or her respective actor.
<font color="green">Checked</font> users are authorized.
<font color="red">Unchecked</font> users are not.<BR>
<HR>
//Display the textboxes in table form
// of each actor with all users below
// with checkboxes to relate the actor
// to a user
$query=sprintf("SELECT guid, name, ip, port, sponsor FROM Actor");
$result = mysql_query($query);
while($row = mysql_fetch_array($result)) {
    <table style="margin-left:2px"
        border="0"
        cellspacing="10"
        cellpadding="1"
        <h2>
            <tr align="left"
                <th width="198">? echo $row['name']; ?></th>
                <th align="left"><div style="border: solid 0 #660;
                    border-color: rgb(248, 180, 66);
                    border-left-width:2px;
                    padding-left:0.5ex">
                    <li>? echo $row['ip']." ".$row['port'];
                        $ip_explode = explode(" ", $_SERVER['HTTP_HOST']);
                        if($row['sponsor'] == 1 and
126 ($ip_explode[0] == $row['ip'])
127     echo "<td><b>LOCAL SPONSOR</b></td>";
128 elseif ($row['sponsor'] == 1)
129     echo "<td><b>Client Sponsor</b></td>";</li>
130     </div></th>
131     <tr align="right">
132     <td><form id="form_assign" method="POST" action="assign_users.php">
133     <input name="au_submit" type="Submit" value="Submit All">
134     </form></td>
135     </tr>
136     <? get_user_assignments($row['guid']); ?>
137     </td>
138     </tr>
139     </h2>
140     <div id="foot">
141     </form>
142     <?
143     echo "</table>";
144     echo "</div>";
145     include('../scripts/footer.php');
146     ?></div>
147 }
148 else
149 {
150     //time expired or access denied; log in again
151     include('../scripts/header.php');
152     ?>
153     Either you are not allowed to access this page, or your session has expired.
154     Please <A href="index.php">log in</a> again.
155 }
156 </div>
157 ?></div>
158
159 /* end of page */
choose_category

```php
<?php
//Purpose: Get an entire category of services as an authorized sponsor.
//Update the Attributes, Attributes_have_Services, and Services tables. Insert the administrator's info in the User table if not already inserted, such as during a fresh install.
//Means: Connect to the Services and Users db as "workers", rsync the services in the filesystem, and ssh mysql commands to the child sponsor.
//Conventions: *_A stands for Attributes
/*_A_h_S stands for Attributes_have_Services
*_S stands for Services
*_U stands for User
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105
?>

?><-----------------------------STANDARD SESSION LIFE AND INACTIVITY CHECK----------------------------->
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $_SESSION;
        session_destroy();
    }
}
//Set the session start time
$_SESSION['start'] = time();

><?

include '../scripts/header.php';
include 'functions_Shell.php';

echo "<h3><center>Choose Categories</center></h3>";
//Display the menu ribbon
include 'menu.php';
//adjusts the header
echo "<br>

<?

<div id="wrap">

<?

// Choose a category to download
if($_SESSION['administrator'] == 1 OR $_SESSION['administrator'] == 2
    AND sending_geni_heartbeats()) {
```
if(isset($_POST['cc_submit'])) {
    include_once 'functions_Shell.php';
    include_once 'functions_Services.php';
    include_once '../scripts/connect_Services.php';
    include_once '../scripts/connect_Users.php';
    include '../scripts/constants.php';
    //Download the subset of services that the administrator of the
    // child sponsor chooses
    foreach($_POST['cc_category'] as $value) {
        //Get the directory tree from Services
        connect_Services();
        $dir_path = directory_tree($value);
        mysql_close(connect_Services());
        //download_category uses send_client_command, which requires
        // remote connection info in Users.Actors
        connect_Users();
        download_category($dir_path);
        mysql_close(connect_Users());
    }
    connect_Users();
    //Get the db connection data for the child sponsor and
    // prepare for a one-time administrator insertion
    $query = sprintf("SELECT user_ID,
                        name,
                        password,
                        guid,
                        administrator
                        FROM User
                        WHERE user_ID='%s'",
                        mysql_real_escape_string($_SESSION['user_ID']))
    or die(mysql_error() . "\n\nQuery25: " . $query);
    $result = mysql_query($query);
    $user = mysql_fetch_array($result);
    //Insert the user, who must be a client administrator on the parent
    // sponsor into the Users.User db on an authorized child sponsor,
    // if there are no other entries, such as when MSS is first
    // installed. Duplicate Key Update effectually does nothing if
    // the administrator already exists in the db. This also gives
    // the client admin the same credentials as on the parent sponsor,
    // but sets the user as a local administrator with full privileges.
    $ins_U = "\n\n\n" . User['name'] . ",
" . User['password'] . ",
" . User['guid'] . ",
"1")
    ON DUPLICATE KEY UPDATE name = name;\n"
    $command_U = "echo $ins_U
                  | mysql -u" . USERS_DB_USER . 
                  -p" . USERS_DB_PASS . 
                  -hlocalhost ".

~ 129 ~
126
127
128
129

//Deliver the insert statement for the User table
send_client_command('ssh', $command_U);

//Prepare the command to delete the Services tables
// from the child sponsor so we can reload them. We must
// delete Attributes_have_Services first because of foreign
// keys constraints.
$del = "\\"DELETE FROM Attributes_have_Services;
DETERM FROM Attributes;
DELETE FROM Services;\\"

$command = "\echo $del
| mysql -u".SERVICES_DB_USER."
-p".SERVICES_DB_PASS."
-hlocalhost ".
SERVICES_DB;

//Send the DELETE command
send_client_command('ssh', $command);

//Close Users so we can connect to services
mysql_close(connect_Users());

//Now connect to services to get the parent sponsor's
// *updated* list of attributes
connect_Services();

//Grab ALL of the attributes
$query=sprintf("SELECT *
FROM Attributes"
)
or die(mysql_error() . "\nQuery26: " . $query);

$result = mysql_query($query);

while($row = mysql_fetch_array($result)) {

    //Prepare the insert statement for delivery to
    // the child sponsor
    $ins_A = ";
    //The root of the recursive table, MSS, requires
    // the word NULL with no single quotes in the
    // parent_ID
    if($row['parent_ID'] == NULL) {
        $ins_A = "\\"INSERT INTO Attributes
VALUES (".$row['attribute_ID'].",
'. $row['attribute'].",
'. $row['description'].",
NULL);\\"
;
    }
    else {
        //Use parent_ID for everything else, and get rid of any
        // parentheses because they ruin the mysql-over-ssh queries.
        $ins_A = "\\"INSERT INTO Attributes
VALUES (".$row['attribute_ID'].",
'. $row['attribute'].",
'. addslashes(str_replace("(", "--",
str_replace(")", "--",
$row['description'])))"),",
'. $row['parent_ID'].")\n\\"
;
    }

    $command_A = "\echo $ins_A

~ 130 ~
mysql_close(connect_Services());

// We need the child sponsor connection data again for the
// send_client_commands
connect_Users();
// Deliver the insert statement for the Attribute table
send_client_command('ssh', $command_A);
mysql_close(connect_Users());

// Now we will only update the Attributes_have_Services and the
// Services tables for the services we downloaded above, thus
// completing the MSS "subset of services" requirement.
foreach($_SESSION['cc_category'] as $category) {
    connect_Services();
    // Grab all of the Attributes_have_Services
    // that match the category
    $query=sprintf("SELECT *
                     FROM Attributes_have_Services
                     WHERE Attributes_attribute_ID='%s'",
                     mysql_real_escape_string($category))
    or die(mysql_error() . "\nQuery27: " . $query);
    $result = mysql_query($query);
    while($row = mysql_fetch_array($result)) {
        // Get only the chosen services
        $query_S=sprintf("SELECT *
                          FROM Services
                          WHERE service_ID='%s'",
                          mysql_real_escape_string($row['Services_service_ID']))
        or die(mysql_error() . "\nQuery28: " . $query_S);
        $result_S = mysql_query($query_S);
        $row_S = mysql_fetch_array($result_S);
        // Prepare the insert statement for delivery to
        // the child sponsor
        $ins_S="";
        // Remove any parentheses, as above
        $ins_S = "\"\"INSERT INTO Services
                     VALUES ("".$row_S['service_ID']."",
                             "".$row_S['name']."",
                             "".$row_S['filename']."",
                             "". addslashes(
                             str_replace("=","--",
                             str_replace(""."", "--",
                             $row_S['description'])))."",
                             "".$row_S['shasum']."",
                             "".$row_S['developer']."",
                             "".$row_S['publisher'].""));\"\";
        $command_S = "echo $ins_S
                       | mysql --user=SERVICES_DB_USER.
                       --password=SERVICES_DB_PASS.
                       --host=localhost
                       SERVICES_DB;";
    }
}
SERVICES_DB;

mysql_close(connect_Services());

//Deliver the INSERT statement for the Services table
connect_Users();
send_client_command('ssh', $command_S);
mysql_close(connect_Users());

//Prepare the insert statement for delivery to
//the child sponsor
$ins_A_h_S = "";

$ins_A_h_S = "INSERT INTO Attributes_have_Services
VALUES (".$row['Attributes_attribute_ID']").",".
$.row['Services_service_ID'].");\"";

$command_A_h_S = "echo $ins_A_h_S
| mysql -u".SERVICES_DB_USER."
-p".SERVICES_DB_PASS."
-hlocalhost ".
SERVICES_DB;

//Deliver the insert statement for the
//Attributes_have_Services table
connect_Users();
send_client_command('ssh', $command_A_h_S);
mysql_close(connect_Users());

} }

} else
{
include 'functions_Services.php';
include './scripts/connect_Services.php';
include './scripts/constants.php';

connect_Services();

echo "This page allows administrators of authorized sponsors

to select entire categories and all of the services within.
\"Check\" the box next to the <i>Category</i> to download
the entire directory of services.<BR>"

// Find the top of the recursive Attributes tree to print
// all of the categories.
$query=sprintf("SELECT *
FROM Attributes
WHERE parent_ID IS NULL")
or die(mysql_error() . "\n Query24: " . $query);

$result = mysql_query($query);

$row = mysql_fetch_array($result);
?
<hr>
<table style="margin-left:2px"
border="0"
cellspacing="0"
cellpadding="4">
<tr align="left">
    <td width="197">
        <? //print attribute at the top of the tree
        echo "{1} ".$row[1]; ?>
    </td>
</tr>

<td align="left">
    <div style="border: solid 0 #060;
        border-color: rgb(248, 180, 66);
        border-left-width:2px;
        padding-left:0.5ex">
        <li> //print description at the top of the // tree
            echo $row[2]; ?></li></div>
    </td>
</tr>

<div id="foot">
    <? //the attribute MSS is "hardwired" in at 1 because it's // parent_ID=NULL, so start recursion at 2
    category_tree($row[0], 2);
    mysql_close(connect_Services());
    }
else {
    echo "This actor does not appear to be an authorized sponsor.
    If this is incorrect, please add this actor as a sponsor on the
    <i>Add Sponsor</i> page.";

    if (
    } ?>

<?php
///////// END OF PAGE ///////////
else
    // time expired or access denied; log in again
    include ('../scripts/header.php');
    ?>
Either you are not allowed to access this page, or your session has expired.
Please &lt;A href="index.php"&gt;log in&lt;/a&gt; again.

<?php
?>

<?
    echo "</table>";
    echo "</div>";
    include('../scripts/footer.php');
    echo "</div>";
?>
choose_service.php

<?php

//Purpose: Get a service as an authorized user on the local machine
//Means: Connect to the Services database as user "workers"
//       Use functions match_service_attributes()
//       service_tree()
//Conventions:
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105
?>

session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $SESSION;
        session_destroy();
    }
}

//Set the session start time
$_SESSION['start'] = time();

 ////////// START OF PAGE /////////////

if ($_SESSION['valid_user'] == true AND $_SESSION['authorized_user'] == true)
{

<?php

//choose the service to download and
// add to the user's list
include '../scripts/connect_Services.php';
include '../scripts/header.php';
include 'functions_Services.php';
include '../scripts/constants.php';
connect_Services();

//Display the menu ribbon
include 'menu.php';
//adjusts the header
echo "<BR>";

?>

<div id="wrap">

<?php

// Choose a service to download

echo "This page allows users to select services.
";\"Click\" the <i>Service ID</i> button to download new services.<BR>";

~ 135 ~
63 // Find the top of the recursive Attributes tree
64 $query = sprintf("SELECT * 
65 FROM Attributes 
66 WHERE parent_ID IS NULL")
67 or die(mysql_error() . "\n Query6: " . $query);
68 $result = mysql_query($query);
69
70 $row = mysql_fetch_array($result);
71 ?>
72
73 <hr>
74
75 <table style="margin-left:2px" border="0" cellspacing="0" cellpadding="4">
76 <h2>
77 <tr align="left">
78 <th width="198">{Level} Category</th>
79 <th><div style="border: solid 0 #060;
80 border-color: rgb(248, 180, 66);
81 border-left-width:2px;
82 padding-left:0.5ex">
83 <li>Description</li>
84 </div></th>
85 </tr>
86 </h2>
87 </table>
88
89 <table style="margin-left:3px" border="0" cellspacing="0" cellpadding="4">
90 <tr align="left">
91 <td width="197">
92 <? //print attribute at the top of the tree
93 echo "{1} ". $row[1]; ?>
94 </td>
95
96 <td align="left">
97 <div style="border: solid 0 #060;
98 border-color: rgb(248, 180, 66);
99 align: 'left';
100 border-left-width:2px;
101 padding-left:0.5ex">
102 <li><? //print description at the top of the tree
103 echo $row[2]; ?></li></div>
104 </td>
105 </tr>
106 </table>
107 <div id="foot">
108 <? //the attribute MSS is "hardwired" in at 1 because it's parent_ID=NULL,
109 // so start recursion at 2
110 service_tree($row[0], 2);
111 ?>
112
113 <?php
114 ///////// END OF PAGE /////////////
115 }
116 else
117 {
118 //time expired or access denied; log in again
119 include ('../scripts/header.php');
120 ?>
121 Either you are not allowed to access this page, or your session has expired.
122 Please <A href="index.php">log in</a> again.
123
124 <?php
125 } ?>
<?php

//Purpose: Connect to the orca database as user "workers"
//Author: John P. Quan
//Version: 1.0
//Date: 20120105

include 'constants.php';

function connect_orca()
{
    $username=GENI_DB_USER;
    $password=GENI_DB_PASS;
    $database=GENI_DB;

    $con = mysql_connect(localhost,$username,$password);
    @mysql_select_db($database) or die( "Unable to select GENI database");

    return $con;
}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document Title</title>
</head>
<body>
    <table border="1">
        <tr>
            <td>Table Data 1</td>
            <td>Table Data 2</td>
        </tr>
    </table>
</body>
</html>
connect_Services.php
1 <?php
2 //Purpose: Connect to the Services database as user "workers"
3 //Author: John P. Quan
4 //Version: 1.0
5 //Date: 20120105
6
7 include 'constants.php';
8
9    function connect_Services()
10    {
11
12       $username=SERVICES_DB_USER;
13       $password=SERVICES_DB_PASS;
14       $database=SERVICES_DB;
15
16       $con = mysql_connect(localhost,$username,$password);
17       mysql_select_db($database) or die("Unable to select database Services");
18
19       return $con;
20    }
21 ?>

connect_Users.php
1 <?php
2 //Purpose: Connect to the Users database as user "workers"
3 //Author: John P. Quan
4 //Version: 1.0
5 //Date: 20120105
6
7 include 'constants.php';
8
9    function connect_Users()
10    {
11
12       $username=USERS_DB_USER;
13       $password=USERS_DB_PASS;
14       $database=USERS_DB;
15
16       $con = mysql_connect(localhost,$username,$password);
17       mysql_select_db($database) or die("Unable to select database Users");
18
19       return $con;
20    }
21 ?>
constants.php

<?php
// Purpose: Define Global variables for this sponsor

// Means: One can put MSS files in /var/www/, or one can put them elsewhere
// in the filesystem, such as in the MSS_USER's home directory. Then,
// create a soft link to the MSS/ directory. For example:
// ln -s /home/work/MSS /var/www/MSS

// Conventions:
// Author: John P. Quan
// Version: 1.0
// Date: 20120105

// Where to store MSS services in the filesystem
define('MSS_SERVICES', '/usr/share');

// The location of the ORCA Actor Registry
define('GENI_HEARTBEATS', 'https://geni.renci.org:12443/registry/actors.jsp');

// The mysql db for the GENI control framework
define('GENI_DB', 'orca');

// The mysql db user for the GENI control framework
define('GENI_DB_USER', 'workers');

// The mysql password for the GENI control framework
define('GENI_DB_PASS', 'InsertPasswordHere');

// The mysql db for the MSS Users db
define('USERS_DB', 'Users');

// The mysql db user for the MSS Users db
define('USERS_DB_USER', 'workers');

// The mysql password for the MSS Users db
define('USERS_DB_PASS', 'InsertPasswordHere');

// The mysql db for the MSS Services
define('SERVICES_DB', 'Services');

// The mysql db user for the MSS Services
define('SERVICES_DB_USER', 'workers');

// The mysql password for the MSS Services
define('SERVICES_DB_PASS', 'InsertPasswordHere');

// The actor that performs MSS tasks on the client's behalf, such as
// connecting to the databases.
// This is a different user than Apache's user 'www-data',
// which performs SSH and SCP tasks on the sponsor's behalf.
define('MSS_USER', 'workers');

// The location of MSS PHP files and scripts in the filesystem
define('MSS_HOME', '/home/work');

// The universal location of the MSS functions and
// shell scripts in the filesystem. Apache user www-data
// must be able to access the directory and files
define('MSS_SCRIPTS', '/scripts/');

?>
cookie.php
1  <php
2  //Start the session, then make session and post variables equal.
3  session_start();
4  //Check if there are any POST variables set
5  if(isset($_POST)) {
6      //If there are POST variables set, then set a SESSION variable
7      //With the same key name
8      foreach($_POST as $key=>$value) {
9          $_SESSION[$key] = $value;
10      }
11  }
12  //Check if there are any SESSION variables set
13  if(isset($_SESSION)) {
14      //If there are SESSION variables set, then set a POST variable
15      //With the same key name, so you don't have to change your
16      //Existing code to reflect '_SESSION' instead of '_POST'
17      foreach($_SESSION as $key=>$value) {
18          $_POST[$key] = $value;
19      }
20  }
21  ?>
del_actors.php

<?php

//Purpose: Delete a sponsor or client
//Means: Connect to the Users.Actor table as user "workers"
//Conventions: da stands for "delete actor"
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105
?>

<?

//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION
        $_SESSION;
        session_destroy();
    }
}
//Set the session start time
$_SESSION['start'] = time();

///////// START OF PAGE ///////////

if ($_SESSION['valid_user'] == true AND $_SESSION['administrator'] == 1) {
    ?>
    <=
    include '../scripts/connect_Users.php';
    include '../scripts/header.php';
    include 'functions_User.php';
    connect_Users();

    echo "<h3><center><font color="red">Delete Actors</font></center></h3>";
    //Display the menu ribbon
    include 'menu.php';
    //adjusts the header
    echo "<BR>";
    ?>
    <div id="wrap">

    ?

    // Check the user name and password after "Submit"
    if (isset($_POST['da_submit'])) {
        echo "Deleted the following actors:<BR>";
        echo "<HR>";
        //Return a count of deleted actors.
        foreach( $_POST['del_actor'] as $value) {
            ?>

            ?>

            ~ 141 ~
$del = explode(" ", $value, 2);

echo "<pre> $del[1]</pre>";

//Delete the Actor from Actor_has_User first
// because of foreign keys
$query=sprintf("DELETE FROM Actor_has_User
    WHERE Actor_guid='%s'",
    mysql_real_escape_string($del[0]));
$result= mysql_query($query);

//Delete from Actor next
$query=sprintf("DELETE FROM Actor
    WHERE guid='%s'",
    mysql_real_escape_string($del[0]));
$result= mysql_query($query);

//unset the delete actor variables
unset_da_vars();
mysql_close(connect_Users());

// Retrieve the Actor List
$query = "SELECT * FROM Actor";
$result = mysql_query($query);

<?
while ($row = mysql_fetch_array($result))
{
  ?><tr
  <td align="left">
    <form id="form_assign"
      method="POST"
      action="del_actors.php">
      <input name="da_submit"
      type="Submit"
      value="Submit All">
    </form>
  </td>
</tr><?
  ?></th><font color="red">DELETE</font></th>
  <th align="left">Name</th>
  <th align="left">Actor GUID</th>
  <th align="left">IP Address</th>
  <th align="left">Port</th>
  <th align="left">Private Key Location</th>
  <th align="left">MSS User Name</th>
  <th align="left">Sponsor</th>
</tr><?
  ?></h2>
</tr>

<?
while ($row = mysql_fetch_array($result))
{
  ?><tr
  <td align="left">
    <form id="form_assign"
      method="POST"
      action="del_actors.php">
      <input name="da_submit"
      type="Submit"
      value="Submit All">
    </form>
  </td>
</tr><?
  ?></th><font color="red">DELETE</font></th>
  <th align="left">Name</th>
  <th align="left">Actor GUID</th>
  <th align="left">IP Address</th>
  <th align="left">Port</th>
  <th align="left">Private Key Location</th>
  <th align="left">MSS User Name</th>
  <th align="left">Sponsor</th>
</tr><?
  ?></h2>
</tr>

// unset the delete actor variables
unset_da_vars();
mysql_close(connect_Users());

// Retrieve the Actor List
$query = "SELECT * FROM Actor";
$result = mysql_query($query);

<?
while ($row = mysql_fetch_array($result))
{
  ?><tr
  <td align="left">
    <form id="form_assign"
      method="POST"
      action="del_actors.php">
      <input name="da_submit"
      type="Submit"
      value="Submit All">
    </form>
  </td>
</tr><?
  ?></th><font color="red">DELETE</font></th>
  <th align="left">Name</th>
  <th align="left">Actor GUID</th>
  <th align="left">IP Address</th>
  <th align="left">Port</th>
  <th align="left">Private Key Location</th>
  <th align="left">MSS User Name</th>
  <th align="left">Sponsor</th>
</tr><?
  ?></h2>
</tr>

// unset the delete actor variables
unset_da_vars();
mysql_close(connect_Users());

// Retrieve the Actor List
$query = "SELECT * FROM Actor";
$result = mysql_query($query);

<?
while ($row = mysql_fetch_array($result))
{
  ?><tr
  <td align="left">
    <form id="form_assign"
      method="POST"
      action="del_actors.php">
      <input name="da_submit"
      type="Submit"
      value="Submit All">
    </form>
  </td>
</tr><?
  ?></th><font color="red">DELETE</font></th>
  <th align="left">Name</th>
  <th align="left">Actor GUID</th>
  <th align="left">IP Address</th>
  <th align="left">Port</th>
  <th align="left">Private Key Location</th>
  <th align="left">MSS User Name</th>
  <th align="left">Sponsor</th>
</tr><?
  ?></h2>
</tr>

// unset the delete actor variables
unset_da_vars();
mysql_close(connect_Users());
he or she is an administrator

// Display the current assignments upon opening, Insert reassignments upon "Submit All".

<td align="center" width="100">
<input type="checkbox" name="del_actor[]" value="" />
</td>
<td>
<? echo $row['name']; ?>
</td>
<td>
<? echo $row['guid']; ?>
</td>
<td align="right">
<? echo $row['ip']; ?>
</td>
<td align="right">
<? echo $row['port']; ?>
</td>
<td>
<? echo $row['private_key_loc']; ?>
</td>
<td>
<? echo $row['mss_user']; ?>
</td>
<td align="center">
<?
$ip_explode = explode(":", $_SERVER['HTTP_HOST']);
if($row['sponsor'] == 1 and ($ip_explode[0] == $row['ip']))
elseif ($row['sponsor'] == 1)
echo "Client";
else echo "No"; ?>
</td>
</tr>

<?
151         }
    }
<?

Either you are not allowed to access this page, or your session has expired. Please <A href="index.php">log in</A> again.

<?php

///////// END OF PAGE /////////////
del_services.php

```php
<?php

//Purpose: Delete a user service client
//Means: Connect to the Users and Services db's as user "workers"
//Conventions: Notice the "includes embedded within the two main
//             if/else statements. "ds" stands for delete service.
//Author: John P. Quan
//Version: 1.0
//Date:    20120105
?>

<?

//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start']))
{
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive)
    {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $_SESSION;
        session_destroy();
    }
}
//Set the session start time
$_SESSION['start'] = time();

///////// START OF PAGE /////////////

if ($_SESSION['valid_user'] == true
    AND $_SESSION['authorized_user'] == true)
{
    ?><
    include '../scripts/header.php';
    echo "<h3><center><font color="red">Delete Services</font></center></h3>";
    //Display the menu ribbon
    include 'menu.php';
    //adjusts the header
    echo "<BR>";
    ?>

    <div id="wrap">
    
    <![CDATA[
    //  Check to see if the user clicked "Submit"
    if (isset($_POST['ds_submit']))
    {
        //Now include User functions
        include '../scripts/connect_Users.php';
        include 'functions_User.php';
        include 'functions_Shell.php';
        //Connect to the Users db to delete service_ID from
    ]]>"
// the user and change the service totals
connect_Users();
echo "Deleted the following services:<BR>";
echo "<HR>";

if(isset($_POST['del_service'])) {
    echo "For ".$_SESSION['username'].":<BR>";
}

//Get the values from the checkboxes, which are a four-part string.
foreach($POST['del_service'] as $value) {
    $del = explode(" ", $value, 4);
    $svc_ID = $del[0];
    $svc_path = $del[1];
    $svc_filename = $del[2];
    $svc_name = $del[3];

    echo "<pre>  $svc_name</pre>";
    //Delete from User_has_Services first
    // because of foreign keys
    $query=sprintf("DELETE FROM User_has_Services
        WHERE Services_service_ID='%s' AND
        User_user_ID='%s'",
        mysql_real_escape_string($svc_ID),
        mysql_real_escape_string($_SESSION['user_ID']))
or die(mysql_error() . "\n Query29: ". $query);
    $result= mysql_query($query);
    //Delete the service from Services next
    $query=sprintf("SELECT * FROM Services
        WHERE service_ID='%s'",
        mysql_real_escape_string($svc_id))
or die(mysql_error() . "\n Query23: ". $query);
    $result= mysql_query($query);
    while($row= mysql_fetch_array($result)) {
        //Leave the service on the Actor unless no one wants the
        // service anymore.  This is necessary to save the expense
        // of re-downloading the service.
        if($row['totals'] < 2) {
            $query=sprintf("DELETE FROM Services
                WHERE service_ID='%s'",
                mysql_real_escape_string($svc_ID));
            $result= mysql_query($query);
            del_downloaded_service($svc_path, $svc_filename);
            echo "<pre>  Removed from the Client.<pre>";
        } else {
            decrement_service_totals ($svc_ID);
        }
    }

    //Give administrators the option to delete the service for everyone
    if($_SESSION['administrator'] == 1) {
AND isset($_POST['del_all_service']) {  
    echo "For Everyone: <br>";
    
}  
// Get the values from the checkboxes, which are a four-part string.
foreach ($_POST['del_all_service'] as $value) {
    $del = explode(" ", $value, 4);
    $svc_ID = $del[0];
    $svc_path = $del[1];
    $svc_filename = $del[2];
    $svc_name = $del[3];
    echo "<pre> $svc_name</pre>";
    
    // Delete from User_has_Services first  
    // because of foreign keys
    $query=sprintf("DELETE FROM User_has_Services  
       WHERE Services_service_ID='%s'",  
       mysql_real_escape_string($svc_ID))
    or die(mysql_error() . " in Query20: " . $query);
    $result= mysql_query($query);
    
    // Delete the service from Services next
    $query=sprintf("DELETE FROM Services  
       WHERE service_ID='%s'",  
       mysql_real_escape_string($svc_ID));
    $result= mysql_query($query);
    
    // delete the service from the client
    del_downloaded_service($svc_path, $svc_filename);
}

    // unset the delete actor variables
    unset_ds_vars();
    // Get the updated list of user services
    get_user_services();
    // Close the connection to the User db
    mysql_close(connect_Users());
} else {
    // Connect to the Services db to get the service information
    // and to set the value to the service ID and name.
    include '../scripts/connect_Services.php';
    include 'functions_Services.php';
    connect_Services();
    echo "<i>Check</i> the services to delete from this client
    and <i>Click</i> the "Submit All" button.<br>"  
    <hr>
    <table style="margin-left:25px; table-layout: auto;"
       border="0"
       cellspacing="10"
       cellpadding="1"
       width="80%">
        <tr>
            <td align="left">
                <form id="form_assign"  
                        method="POST"
```php
//Retrieve the location of the service in the filesystem
$query = sprintf("SELECT * FROM view_Attributes_Services"
    or die(mysql_error() . " Query21: " . $query);
$result = mysql_query($query);
while ($row = mysql_fetch_array($result))
{
    <tr>
        <td align="center" width="100" style="background-color:red">
            <input type="checkbox"
                name="del_all_service[]"
                value="<? echo $row['service_ID'] . " 
                .directory_tree( 
                $row['attribute_ID'] . " 
                ." . $row['filename'] . " 
                ." . $row['name'] . "; ?>">
        </td>
    </tr>
    </tr>
</h2>
```


```
```
<td align="left">
  <input name="ds_submit" type="Submit" value="Submit All">
</td>
</form>
</div>

<?php

//Close the connection to Services db
mysql_close(connect_Services());

<?php

//---- END OF PAGE ----//

} else {

<?php

echo "Either you are not allowed to access this page, or your session has expired. Please <A href="index.php">log in</A> again.

<?php

~ 148 ~
<?php

//Purpose: Delete a user or administrator
//Means: Connect to the Users.User table as user "workers"
//Conventions: du stands for "delete actor"
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105
?>

<?
//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start']))
{
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive)
    {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $_SESSION;
        session_destroy();
    }
}
//Set the session start time
$_SESSION['start'] = time();

//////////// START OF PAGE /////////////

if ($_SESSION['valid_user'] == true
    AND $_SESSION['administrator'] == 1)
{
    ?>
    <![CDATA[

<h3>Deleted Users</h3>

//Display the menu ribbon
include 'menu.php';
//adjusts the header
echo "<BR>

]]></CDATA[>

<font color="red">Delete Users</font></center></h3>";
//Display the menu ribbon
include 'menu.php';
//adjusts the header
echo "<BR>

<?

if (isset($_POST['du_submit']))
{
    echo "Deleted the following users:<BR>";
    echo "<HR>";
    //Return a count of deleted actors.
    foreach( $_POST['del_user'] as $value) {

~ 149 ~
```php
$del = explode(" ", $value, 2);

//Delete the User from User_has_Services first
// because of foreign keys
$query=sprintf("DELETE FROM User_has_Services
    WHERE User_user_ID='%s'",
    mysql_real_escape_string($del[0]));
$result= mysql_query($query);

//Delete the User from Actor_has_User next
// because of foreign keys
$query=sprintf("DELETE FROM Actor_has_User
    WHERE User_user_ID='%s'",
    mysql_real_escape_string($del[0]));
$result= mysql_query($query);

//Delete from User last
$query=sprintf("DELETE FROM User
    WHERE user_ID='%s'",
    mysql_real_escape_string($del[0]));
$result= mysql_query($query);
}

//unset the delete actor variables
unset_du_vars();
mysql_close(connect_Users());

// Retrieve the User List
$query = "SELECT * FROM User";
$result = mysql_query($query);

<!DOCTYPE html>
<html>
<head>
    <title>Delete Users</title>
    <style>
        table { border-collapse: collapse; border: 1px solid #ccc; table-layout: auto; margin-left: 25px; }
        th, td { padding: 10px; border: 1px solid #ccc; text-align: left; font-weight: normal; vertical-align: middle; }
        th { background-color: #f2f2f2; }
    </style>
</head>
<body>
<form id="form_assign" method="POST" action="del_users.php">
    <input name="du_submit" type="Submit" value="Submit All"/>
</form>
    <h2>Delete Users</h2>
    <table>
        <tr>
            <th>DELETE</th>
            <th>Name</th>
            <th>User ID</th>
            <th>GUID</th>
            <th>Administrator</th>
        </tr>
        <?
            while ($row = mysql_fetch_assoc($result))
                echo "<tr>
                    <td><font color="red">DELETE</font></td>
                    <td>" . htmlspecialchars($row['Name']) . "</td>
                    <td>" . htmlspecialchars($row['User_ID']) . "</td>
                    <td>" . htmlspecialchars($row['GUID']) . "</td>
                    <td>" . htmlspecialchars($row['Administrator']) . "</td>
                </tr>
            endwhile;
        ?>
    </table>
</body>
</html>
```
<?php
//List the user names with a checkbox and whether he or she is an administrator.
//Display the current assignments upon opening, insert reassignments upon "Submit All".
<tr><td align="center" width="100">
<input type="checkbox" name="del_user[]" value="
<?
echo $row['user_ID']. " 
$row['name']; ?>"
/>
</td><td>"<?
echo $row['name']; ?>"</td><td align="center">
<?
echo $row['user_ID']; ?>
</td><td align="left">
<?
echo $row['guid']; ?>
</td><td align="center">
<? if ($row['administrator'] == 0) echo 'No';
elseif ($row['administrator'] == 1) echo 'LOCAL';
elseif ($row['administrator'] == 2) echo 'Client';
else echo 'User type unknown! Contact your Local Administrator.'; ?>
</td>
</tr>
<div id="foot">
<?
echo "</table>
";
echo "</div>
include('..//scripts/footer.php');
echo "</div>
";

<?php
//////////// END OF PAGE //////////////
}

else {
include ('..//scripts/header.php');
<?
!--Print error message and offer to login again-->
Either you are not allowed to access this page, or your session has expired.
Please <A href="index.php">log in</A> again.

<?php
} ?>

~ 151 ~
download_service.php

```php
<?php

// Purpose: Get a service from an authorized server as an authorized user on an authorized client machine.
// Means: Connect to the Services database as user "workers"
//       Use functions match_service_attributes()
//             service_tree()
// Conventions:
// Author:  John P. Quan
// Version: 1.0
// Date:    20120105
?>

<?php

//////////////////////////////////////////////////////
//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $_SESSION;
        session_destroy();
    }
}

//Set the session start time
$_SESSION['start'] = time();

//////////////////////////////////////////////////////
//If the remote user and host exist in the Actor table,
// then add the service to the user, increment the service
```

```
// totals, and download the service
if (actor_ip_exists($_SERVER['HTTP_HOST'])
    AND actor_ip_exists($_SERVER['REMOTE_ADDR'])) {
    //Increment the total number of users using the service
    increment_service_totals(
        $_SESSION['svc_download']);
    // Then add the service to the user's list
    // of services.
    add_user_service(
        $_SESSION['svc_download']);

    <hr>

    <table style="margin-left:25px; table-layout: auto;"
    border="0"
    cellspacing="10"
    cellpadding="1">
    <h2>
        <tr>
            <th>Location</th>
            <th>File</th>
        </tr>
    </h2>
    <tr>
        <td><? echo $_SESSION['svc_path']; ?></td>
        <td><? echo $_SESSION['svc_filename']; ?></td>
    </tr>
</table>

    ///Download the service to the client.
    download_service(
        $_SESSION['svc_path'],
        $_SESSION['svc_filename']);
    //Compare the shasums to ensure the download was successful
    if (compare_shasum($_SESSION['svc_path'],
        $_SESSION['svc_filename']) ) {
        echo "<font color="\"green\">Download Successful!</font><BR><HR>";
    } else {
        echo "<font color="\"red\">Downloaded SHA sum does not match Sponsor's SHA Sum!
        Download FAILED!</font><BR><HR>";
        del_downloaded_service($_SESSION['svc_path'],
            $_SESSION['svc_filename']);
    } else {
        echo "<font color='red'>Service Downloading Disabled. You are not downloading from an authorized address.<BR>
        Please contact your administrator.<color>";
    }
    //update the user services list in session
go_get_user_services();

    <div id="foot">
    </table>

~ 153 ~
else {
    // time expired or access denied; log in again
    include ('../scripts/header.php');
    ?
    Either you are not allowed to access this page, or your session has expired.
    Please [A href="index.php"]log in[/a] again.
}

<?php
}
<?

include('../scripts/footer.php');
echo "</div>

<?
?>

footer.php
1 <end><p1>
2<br>Web Administrator:
3<br>John Quan</p1></end>
4 </body>
5 </html>
functions_php.php

1  <?php
2  //Purpose: Extended or general-purpose PHP functions
3  //Means:
4  //Conventions: gen_uuid() generates a GUID
5  //Author:  John P. Quan
6  //Version: 1.0
7  //Date:    20120105
8  ?>
9
10 <?php
11 // Generate a guid to distinguish objects
12 function gen_uuid() {
13     return sprintf( '%04x%04x-%04x-%04x-%04x%04x%04x%04x',
14         mt_rand( 0, 0xffff ), mt_rand( 0, 0xffff ),
15         mt_rand( 0, 0xffff ),
16         mt_rand( 0, 0xffff ),
17         mt_rand( 0, 0xffff ),
18         mt_rand( 0, 0xffff ),
19         mt_rand( 0, 0xffff ),
20         mt_rand( 0, 0xffff ),
21         mt_rand( 0, 0xffff ),
22         mt_rand( 0, 0xffff ),
23     );
24 }
25
26
27 <?php
28 //Searches for even a partial match.
29 function key_array_search($needle = null, $haystack_array = null, $skip = 0) {
30     if($needle == null || $haystack_array == null)
31         die('Needle and haystack array are mandatory!');
32     foreach($haystack_array as $key => $eval)
33         if($skip != 0) $eval = substr($eval, $skip);
34         if(stristr($eval, $needle) !== false) return $key;
35     return FALSE;
36 }
37
38 #~ 155 ~
functions_services.php

<?php
//Purpose: Service related functions
//Author: John P. Quan
//Version: 1.0
//Date: 20120105
include_once 'functions_Shell.php';

//breadth-first recursion through Services.Attributes
function service_tree($ID, $level) {
    $query = sprintf("SELECT attribute_ID, attribute, description, parent_ID FROM Attributes WHERE parent_ID=$ID")
    or die(mysql_error() . "\nQuery30: " . $query);
    $result = mysql_query($query);
    while ($row = mysql_fetch_array($result)) {
        //add [one space]*[attribute level]
        //for readability
        for ($i = 0; $i < $level; $i++) {
            echo "&nbsp";
        }
        //print the level, attribute name, and description for each
        //attribute
        echo "{".$level."} ".$row['attribute'];
    }
    //if the attribute has services, print them
    match_service_attributes($row['attribute_ID']);
    //recurse through the attribute tree and add a level each time
    service_tree($row['attribute_ID'], $level + 1);
}

function category_tree($ID, $level) {
//breadth-first recursion through Services.Attributes
//Print the tree level, attribute, and checkbox
…

~ 156 ~
$query = sprintf("SELECT attribute_ID,
  attribute,
  description,
  parent_ID
FROM Attributes
WHERE parent_ID=$ID")
  or die(mysql_error() . "\n Query22: " . $query);

$result = mysql_query($query);
while ($row = mysql_fetch_array($result)) {
?>

<table border="0" cellspacing="0" cellpadding="4">
<tr>
<td width="200">
<?
  //add [one space]*[attribute level]
  // for readability
  for ($i = 0; $i < $level; $i++) {
    echo "&nbsp";
  }
  //print the level, attribute name, and description for each
  // attribute
  echo "({$level}.") ".$row['attribute'];
?>
<?
  //List the user names with a checkbox and whether
  // he or she is an administrator
  //Display the current assignments upon opening,
  // Insert reassignments upon "Submit All".
  echo display_category_checkbox ($row['attribute_ID']); ?>
</td>
<td><div style="border: solid 0 #060;
border-color: rgb(248, 180, 66);
border-left-width:2px;
padding-left:0.5ex">
<li><? echo $row['description']; ?>
</li>
</div>
</td>
</tr></table>
<?
  //if the attribute has services, print them
  //match_service_attributes($row['attribute_ID']);
  //recurse through the attribute tree and add a level each time
  category_tree($row['attribute_ID'], $level + 1);
} }

<? ///////////// DISPLAY_CATEGORY_CHECKBOX ///////////\\\\\\\\\\\\\\\\
//Return matching services for an attribute
function display_category_checkbox ($attribute_ID) {
  $query = sprintf("SELECT Services_service_ID
FROM Attributes_have_Services
WHERE Attributes_attribute_ID=$attribute_ID");
  $result = mysql_query($query);
  while ($row = mysql_fetch_array($result)) {
    //create a table for each service to display
    // below the attribute
    return "<input type="checkbox""
$$attribute_ID$$

```php

return NULL;
```
function display_service($service_ID) {

    //pull from the Services database
    $query = sprintf("SELECT service_ID,
                   name,
                   filename,
                   description,
                   developer,
                   publisher
                   FROM Services
                   WHERE service_ID=$service_ID");

    $result = mysql_query($query);

    //print the service as a table. This table is uniquely formatted
    $row = mysql_fetch_array($result) {
        ?>
        <tr>
            <th></th>
            <th></th>
            <th align="left">Service</th>
            <th align="left">Description</th>
            <th align="left">Developer</th>
            <th align="left">Publisher</th>
        </tr>
        <?
        while ($row = mysql_fetch_array($result)) {
            ?>
            <tr>
                $printed = FALSE;
                //If the service has not been printed already,
                //print the INSTALLED service array
                foreach ($_SESSION['svc'] as $svc_ID) {
                    if (!$printed AND $svc_ID == $row['service_ID']) {
                        ?>
                        <td width="190">
                            <? //Placeholder cell ?>
                        </td>
                        <td valign="top" align="center" width="100">
                            <font color="green">Installed</font>
                        </td>
                        <td valign="top" width="100">
                            <? echo $row['name']; ?>
                        </td>
                        <td valign="top" width="700">
                        </td>
                    }
                }
            }
        }
    }
<? echo $row['description']; ?>
</td>
<td valign="top" width="125">
<? echo $row['developer']; ?>
</td>
<td valign="top" width="150">
<? echo $row['publisher']; ?>
</td>
</tr>
</table>

<? //set $printed to true, otherwise this prints // the same service times the number of total // services the user has $printed = TRUE;
}

//if it is not printed, then it is not installed // Give the user the option to download the service if (!$printed) {
<?

<td width="190">
<? //Placeholder ?>
</td>

<? //Display the service_ID in the submit button // as the value. ?>
<td valign="top" align="center" width="100">
<form id="form_download" method="POST" action="choose_service.php">
<input type="submit" name="svc_download" value="<? //When clicked, set the button text to // "Downloading" if(isset($_SESSION['svc_path'])) {
    echo "Downloading";
} else echo $row['service_ID']; ?>">
</form>
</td>

<? echo $row['name']; ?>
</td>

<? echo $row['description']; ?>
</td>
<td valign="top" width="125">
<? echo $row['developer']; ?>
</td>
<td valign="top" width="150">
<? echo $row['publisher']; ?>
</td>

</tr>
</table>
<? //Now this service's table is definitely printed, // if it exists $printed = TRUE;

?>

<? //_DIRECTORY_TREE //Depth-first recursion to find the full path of the file based on the MSS SERVICES and it's attribute description

function directory_tree($attribute_ID, $full_path = '') {

    $query = sprintf("SELECT attribute_ID, attribute, parent_ID FROM Attributes WHERE attribute_ID=$attribute_ID");
    $result = mysql_query($query);
    $row = mysql_fetch_array($result);
    //This recurses backwards to the top of the tree, so put the next attribute BEFORE the last.

    if ($attribute_ID == NULL) {
        $temp = $row['attribute'].'/'.$full_path;
        $full_path = MSS_SERVICES.$temp;
        return (string)$full_path;
    } else {
        $temp = $row['attribute'].'/'.$full_path;
        $full_path = $temp;
        return directory_tree($row['parent_ID'], $full_path);
    }
}

?>

<? //SUBMIT_DOWNLOAD //load SESSION svc_path with the full path name

function submit_download($service_ID) {

    //match the attribute to the service
    $query = sprintf("SELECT service_ID, attribute_ID, filename, shasum FROM view_Attributes_Services WHERE service_ID=$service_ID");
    $result = mysql_query($query);
    while ($row = mysql_fetch_array($result)) {
        //Recursive directory tree + filename = full path
        $SESSION['svc_path'] = directory_tree($row['attribute_ID'], $row['filename']);
        $SESSION['svc_filename'] = $row['filename'];
        $SESSION['svc_shasum'] = $row['shasum'];
    }
}

?>
functions_shell.php

1.php
2 //Purpose: Runs shell scripts on the host
3 //Means: uses the backtick operator as user 'www-data' (The default
4 //apache user) on the host, scp's and ssh's as user MSS_USER
5 // on the client
6 //Conventions: THIS RUNS AS THE APACHE USER 'www-data', SO
7 // ASSUMING MSS_USER = 'workers'
8 // ON THE SERVER:
9 // -change www-data's default shell to bash
10 // in /etc/passwd
11 // -add these groups:
12 // usermod -a -G workers www-data
13 // (allows www-data to run the shell scripts)
14 // usermod -a -G mss www-data
15 // (all MSS users belong to group mss)
16 // -set the UID/GID on /home/work/MSS directory to:
17 // chmod ug+ts /home/work/MSS
18 // chmod 0770 /home/work/MSS
19 // -add the private key and change ownership to www-data, e.g.,
20 // -copy the Eucalyptus private key into the /home/workers/.ssh
21 // directory, then:
22 // chown www-data:www-data barrowkey.private
23 // chmod 0600 barrowkey.private
24 // ON A CLIENT EUCALYPTUS INSTANCE (Ubuntu 11.04):
25 // -as root...
26 // -add group mss
27 // addgroup mss
28 // usermod -a -G mss ubuntu
29 // -add directory /usr/share/MSS
30 // mkdir /usr/share/MSS
31 // chown ubuntu:mss /usr/share/MSS
32 // ON A PHYSICAL CLIENT:
33 // -add group mss
34 // addgroup mss
35 // -add user workers
36 // adduser workers
37 // usermod -a -G mss workers
38 // -add directory /usr/share/MSS
39 // mkdir /usr/share/MSS
40 // chown workers:mss /usr/share/MSS
41 // ON A PHYSICAL SERVER FOR A EUCALYPTUS INSTANCE:
42 // -for instance, on the MSS "Add Client" webpage, add:
43 // User Name [unique name, such as its IP, or IP-PORT]
44 // IP
45 // PORT
46 // PRIVATE KEY [the eucalyptus key, such as
47 // /home/orca/mykey.private]
48 // MSS USER [ubuntu]
49 // ON A PHYSICAL SERVER:
50 // su workers
51 // -use passwordless ssh by placing the public key in the
52 // authorized_users file. For instance:
53 // ssh-keygen
54 // ssh-copy-id -p XXXX -i ~/.ssh/id_dsa.pub workers@client.ip.’
55 // (the public key must be in /home/workers/.ssh/authorized_keys)
56 //Author: John P. Quan
57 //Version: 1.0
58 //Date: 20120105
59 ?>
60
61 include_once 'functions_php.php';

~ 162 ~
include_once 'functions_User.php';

<? /////////////////////////////////////////////////////////
//wget the ORCA Registry and check to see if the MSS_PARENT (sponsor)
//is in it.
// return TRUE || FALSE
function sending_geni_heartbeats() {
    //sh_check_heartbeats wgets the ORCA Actor Registry, greps for
    // url: [url of all actors]
    // amdiff: [if this is > 0, you're not actively donating
    // and returns an array( [0]=> url_0
    // [1]=> amdiff_0
    // [2]=> url_1
    // [3]=> amdiff_1 ... )
    $heartbeats = GENI_HEARTBEATS;
    $script = MSS_HOME.MSS_SCRIPTS;
    //Download the ORCA Actors Registry
    // amdiff is greater than 0 if the actor is NOT donating,
    // or if the actor does not exist on the page.
    $h_arr = explode( '/' , $heartbeats);
    //Get the page name at the end of GENI_HEARTBEATS. As of
    // this writing, the name of the page is "actors.jsp"
    $page = end($h_arr);
    //Run the shell script to get all actors
    // and check for sponsor's heartbeats
    $data = ` /bin/bash $script/sh_check_heartbeats $heartbeats $page`;
    $actors = explode( " ", $data);
    //In HTTP_HOST IP:PORT, exclude the port because the host is
    // serving the GENI control framework on a different one, so
    // it will not find a match with HTTP_HOST in the Actors Registry
    // e.g., GENI CF: 11080, MSS: 12080
    $ip = explode(": ", $_SERVER['HTTP_HOST']);
    //I know I (the sponsor) am sending heartbeats
    // because I $found myself in the $actors array...
    $found = key_array_search($ip[0], $actors);
    if($found) {
        //...and the next value in $found + 1 equals 0...
        if($actors[$found + 1] == 0) {
            //...so the child (user) can download the service
            return TRUE;
        } else {
            echo "<font color='red'>
            Service Downloading Disabled. No heartbeats present.<BR>
            Please contact your administrator.
            </font>";
            return FALSE;
        }
    } else {
        //...else I am not in the Actor Registry, so the user cannot
        // download new services.
        else {
            echo "<font color='red'>
            ~ 163 ~
            
            ~ 163 ~
Service Downloading Disabled. No heartbeats present.<BR>
Please contact your administrator.<BR>
<color>";
return FALSE;
)
?>
<?/////////// DOWNLOAD_SERVICE/////////////////////////////////
// Download the service to the client.
function download_service($svc_path, $svc_filename) {
    //Only download if the service does not exist on the client.
    // We can do this because new releases of a service will be
    // named differently.
    $command = "$ls -hal $svc_path
                 | grep $svc_filename
                 | awk '{print $5}'';
    $return = send_client_command('ssh', $command);
    //file_size returns the file size or a single character. I was
    // not able to figure out what this character is in a timely
    // manner.
    if(strlen($return) < 2) {
        //Make the MSS directory in MSS_SERVICES if it doesn't exist
        $command = "mkdir -p $svc_path";
        send_client_command('ssh', $command);
        // Rsync the file to the client
        send_client_command('rsync', NULL, $svc_path, $svc_filename);
    }
    else echo "File $svc_filename previously installed. ";
}
?>
<?/////////// DOWNLOAD_SERVICE/////////////////////////////////
// Download the service to the client.
function download_category($dir_path) {
    //Prepare to take the last directory name off of
    // the string
    $tmp = explode("/", $dir_path);
    //The dir_path ends in a "/", so you have to pop
    // it once for the copy directory...
    array_pop($tmp);
    //Put the shortened path back together
    $copy_this_dir = implode("/", $tmp);
    //...and a second time for the copy location
    $category = array_pop($tmp);
    //Put the shortened path back together
    $loc = implode("/", $tmp);
    //Echo the results
    echo "Copied the ".$category." category to ".$loc."<BR>">
    //Make the MSS directory in MSS_SERVICES if it doesn't exist
    $command = "mkdir -p ".$dir_path;
    send_client_command('ssh', $command);
    //Recursive secure copy the directory to the client

~ 164 ~
send_client_command('dir', NULL, $loc, $copy_this_dir);
 />

function compare_shasum($svc_path, $svc_filename) {

    $command = "shasum ".$svc_path.$svc_filename;
    $shasum = send_client_command('ssh', $command);
    if(strcmp($shasum, $_SESSION['svc_shasum'])) return TRUE;
    else return FALSE;
}

function del_downloaded_service($svc_path, $svc_filename) {

    $command = "rm ".$svc_path.$svc_filename;
    send_client_command('ssh', $command);
}

function send_client_command($c_type = 'ssh', $command=NULL, $path=NULL, $copy_this=NULL) {

    $script = MSS_HOME.MSS_SCRIPTS;

    $user=(string)get_client_mss_user();

    $address=(string)$_SERVER['REMOTE_ADDR'];

    $port=(string)get_client_port();

    $private_key=(string)get_client_private_key();

    if($c_type === 'ssh') {

        $ssh_command = "".$command."";

        //...and send using the BASH script sh_ssh_command
```bash
$data='/bin/bash $script/sh_ssh_command $user \\
   $address \\
   $port \\
   $private_key \\
   $ssh_command';

    return $data;
}

//Run rsync to download a service to the client
elseif($c_type === 'file') {

   $full_path = $path.$copy_this;
   $loc = $path.".";

   //...and send using the BASH script sh_rsync_command
   $data='/bin/bash $script/sh_rsync_command $user \\
         $address \\
         $port \\
         $private_key \\
         $full_path $loc';

    return $data;
}

//Run rsync to download a directory to the client
elseif($c_type === 'dir') {

   //This is the full path of the directory
   $full_path = $copy_this;
   //This is the directory just above the one you
   // want to copy
   $loc = $path;

   //...and send using the BASH script sh_rsync_command
   $data='/bin/bash $script/sh_rsync_command $user \\
         $address \\
         $port \\
         $private_key \\
         $full_path $loc';

    return $data;
}
else {
   echo "Type must be 'file', 'dir', or 'ssh'.<BR>");
   return -1;
}
} ?>

~ 166 ~
functions_User.php

```php
<?php

//Purpose: User related functions
//Author: John P. Quan
//Version: 1.0
//Date: 20120105

include_once '../scripts/connect_Users.php';
include_once "../scripts/connect_orca.php";

connect_Users();
?>

<?

///////////////////////////////////////////////////////////////////////////////////////////////////////

//Compare the session actors array with the server IP
// to determine whether the user can access the website

function check_authorized_user() {

    //Allows for special case of no actor/user assignments,
    // such as when MSS is first installed.
    //This always allows local administrators in to the sponsor, even
    // when no users are assigned to actors.
    //The site will not allow access for others
    // unless a local administrator assigns the user to the sponsor and
    // at least one actor.
    if($_SESSION['administrator'] == 1) {
        $_SESSION['authorized_user'] = "true";
    }

    //See if the client IP is a sponsor to set whether the
    // administrator can download to it. The local admin can
    // always choose to make this IP a sponsor, and the act of
    // doing so cements which IPs the admin wants to allow to download.

    $ip_explode = explode(":", $_SERVER['REMOTE_ADDR']);

    foreach($_SESSION['actors'] as $value) {
        $query = sprintf("SELECT sponsor, guid
                        FROM Actor
                        WHERE ip='%s'
                        AND guid='%s'",
                        mysql_real_escape_string($ip_explode[0]),
                        mysql_real_escape_string($value))
        or die(mysql_error() . "\nQuery7: " . $query);
        $result = mysql_query($query);

        //If the server IP address is in the user's assigned
        // session actors, return TRUE, else return FALSE.
        while($actor = mysql_fetch_array($result)) {
            //Find out if one of the authorized sponsors is this server
            if($actor['sponsor'] == 1
                and authorized_sponsor($actor['guid'])) {
                $_SESSION['authorized_sponsor'] = "true";
            }
        }
    }
    return TRUE;
}

else {
    //The person who logged on is not an administrator, but
}?
```
// he or she must be a client administrator or a user, so
// find out if he or she is allowed access.
$ip_explode = explode(":", $_SERVER['HTTP_HOST']);

foreach($_SESSION['actors'] as $value) {
    $query = sprintf("SELECT sponsor, ip, guid
FROM Actor
WHERE ip='%s'
AND guid='%s'",
    mysql_real_escape_string($ip_explode[0]),
    mysql_real_escape_string($value))
or die(mysql_error() . "\nQuery31: " . $query);
    $result = mysql_query($query);

    //If the server IP address is in the user's assigned
    // session actors, return TRUE, else return FALSE.
    while($actor = mysql_fetch_array($result)) {
        //Find out if one of the authorized sponsors is this server
        if($actor['sponsor'] == 1
            and authorized_sponsor($actor['guid'])) {
            $_SESSION['authorized_sponsor'] = "true";
        }
     }

    //See if the user is authorized on this actor
    $query = sprintf("SELECT Actor_guid, User_user_ID
FROM Actor_has_User
WHERE Actor_guid='%s'",
    mysql_real_escape_string($actor['guid']))
or die(mysql_error() . "\nQuery8: " . $query);
    $result = mysql_query($query);

    while( $A_h_U = mysql_fetch_array($result)) {
        if($A_h_U['User_user_ID'] == $_SESSION['user_ID']) {
            $_SESSION['authorized_user'] = "true";
            return TRUE;
        }
    }
    $_SESSION['authorized_user'] = "false";
    return FALSE;
}

?? function authorized_sponsor($guid) {

    //Close the Users connection so authorized_sponsor() can
    // connect to the GENI db
    mysql_close(connect_Users());

    //If the actor guid matches a sponsor guid
    // return TRUE, else return FALSE.
    connect_orca();

    ~ 168 ~
$query = sprintf("SELECT act_guid
FROM Actors
WHERE act_guid='%s'",
    mysql_real_escape_string($guid))
or die(mysql_error() . "\n Query10: " . $query);
$geni_result = mysql_query($query);
while($geni_row = mysql_fetch_array($geni_result)) {
    //Find out if one of the authorized sponsors is this server
    if(isset($geni_row['act_guid'])) {
        mysql_close(connect_orca());
        connect_Users();
        return TRUE;
    }
    mysql_close(connect_orca());
    connect_Users();
    return FALSE;
}

// Add the service to the user's list
// of services and update the total
// number of users using the service
function add_user_service($service_ID) {
    //Insert the user_ID and service_ID into
    //User_has_Services
    $user_ID = $_SESSION['user_ID'];
    $query = mysql_query(
        "INSERT INTO User_has_Services (User_user_ID,
        Services_service_ID)
VALUES(    '$user_ID',
        '$service_ID' ) ")
    or die(mysql_error() . "\n Query9: " . $query);
}

//Increment the service totals if the user downloads the service
function increment_service_totals($service_ID) {
    if(!service_ID_exists($service_ID)) {
        $query = mysql_query(
            "INSERT INTO Services (service_ID)
VALUES(      '$service_ID' ) ")
    or die(mysql_error() . "\n Query10: " . $query);
    }
    //find the current service total
    $query=sprintf("SELECT service_ID,
    totals
FROM Services
WHERE service_ID=$service_ID")//,
    or die(mysql_error() . "\n Query11: " . $query);
$result = mysql_query($query);
$i=0;
while($row = mysql_fetch_array($result)) {
    $i = $row['totals'];
    $i++;
    $update = sprintf("UPDATE Services
             SET totals=$i
             WHERE service_ID=$service_ID"
     or die(mysql_error()) . "\n Query12: " . $update);
    mysql_query($update);
}

$ip_explode = explode( ":", $ip);
//Check to see if the parent or child ip is in the Users.Actor database
$query = sprintf("SELECT ip
             FROM Actor
             WHERE ip='%s'", $ip_explode[0]);
$result = mysql_query($query);
while($row = mysql_fetch_array($result)) {
    if($row['totals'] == 1) {
        $query = mysql_query("DELETE FROM Services
             WHERE service_ID=$service_ID"
     or die(mysql_error()) . "\n Query14: " . $query);
    } else {
        $i = $row['totals'];
        $i--;
        $update = sprintf("UPDATE Services
             SET totals=$i
             WHERE service_ID=$service_ID"
     or die(mysql_error()) . "\n Query15: " . $update);
    }
    mysql_query($update);
}
mysql_real_escape_string($ip_explode[0])
or die(mysql_error() . "\n Query16: " . $query);

$result = mysql_query($query);
while($row = mysql_fetch_array($result)) {
    if(isset($row['ip'])) return TRUE;
}
return FALSE;

function service_ID_exists($service_ID) {
    //Check to see if the service is in the Users.Service database
    $query = sprintf("SELECT service_ID
                    FROM Services
                    WHERE service_ID='%s'",
                    mysql_real_escape_string($service_ID))
    or die(mysql_error() . "\n Query17: " . $query);

    $result = mysql_query($query);
    while($row = mysql_fetch_array($result)) {
        if(isset($row['service_ID'])) return TRUE;
    }
    return FALSE;
}

function get_client_port() {
    foreach($_SESSION['actors'] as $value) {
        //Get the client port from the Users.Actor database
        $query = sprintf("SELECT port
                        FROM Actor
                        WHERE ip='%s'
                        AND guid='%s'",
                        mysql_real_escape_string($SERVER['REMOTE_ADDR']),
                        mysql_real_escape_string($value))
        or die(mysql_error() . "\n Query32: " . $query);

        $result = mysql_query($query);
        while($row = mysql_fetch_array($result)) {
            if(isset($row['port'])) return $row['port'];
        }
    }
    return FALSE;
}

function get_client_private_key() {
    foreach($_SESSION['actors'] as $value) {
        //Get the client private key from the Users.Actor database
        $query = sprintf("SELECT private_key_loc
                        FROM Actor
                        WHERE ip='%s'
                        AND guid='%s'",
                        mysql_real_escape_string($SERVER['REMOTE_ADDR']),
                        mysql_real_escape_string($value))
        or die(mysql_error() . "\n Query33: " . $query);

        $result = mysql_query($query);
        while($row = mysql_fetch_array($result)) {
            if(isset($row['private_key_loc'])) return $row['private_key_loc'];
        }
    }
}
FROM Actor
    WHERE ip='%s'
    AND guid='%s'",
    mysql_real_escape_string($_SERVER['REMOTE_ADDR']),
    mysql_real_escape_string($value))
    or die(mysql_error() . "\nQuery33: " . $query);

$result = mysql_query($query);
while($row = mysql_fetch_array($result)) {
    if(isset($row['private_key_loc'])) return $row['private_key_loc'];
}
return FALSE;

function get_client_mss_user() {
    foreach($_SESSION['actors'] as $value) {
        //Get the client username from the Users.Actor database
        $query = sprintf("SELECT mss_user
            FROM Actor
            WHERE ip='%s'
            AND guid='%s'",
            mysql_real_escape_string($_SERVER['REMOTE_ADDR']),
            mysql_real_escape_string($value))
        or die(mysql_error() . "\nQuery19: " . $query);

        $result = mysql_query($query);
        while($row = mysql_fetch_array($result)) {
            if(isset($row['mss_user'])) return $row['mss_user'];
        }
    return FALSE;
}

function get_user_services() {
    unset_svc_vars();
    if(!isset($_SESSION['svc'])) {
        $_SESSION['svc'] = array();
    }
    //List the user's services as a array in SESSION
    $query = sprintf("SELECT User_user_ID,
        Services_service_ID
        FROM User_has_Services
        WHERE User_user_ID='%s'",
        mysql_real_escape_string($_SESSION['user_ID']));
    $result = mysql_query($query);
    while($row=mysql_fetch_array($result)) {
        array_push($_SESSION['svc'], $row['Services_service_ID']);
    }
}
function get_user_actors($user_ID) {
    // List the user's services as a array in SESSION
    $_SESSION['actors'] = array();
    $query = sprintf("SELECT Actor_guid,
        User_user_ID
        FROM Actor_has_User
        WHERE User_user_ID='%s'",
        mysql_real_escape_string($user_ID));
    $result = mysql_query($query);
    while($row=mysql_fetch_array($result)) {
        array_push($_SESSION['actors'], $row['Actor_guid']);
    }
}

function get_user_assignments($guid) {
    // List all users under each actor
    $query = sprintf("SELECT user_ID,
        name,
        administrator
        FROM User");
    $result = mysql_query($query);
    while($row=mysql_fetch_array($result)) {
        <table border="0" cellspacing="10" cellpadding="1">
            <tr>
                <td align="center" width="100">
                    <?php
                        echo '<input type="checkbox" name="au_assign["';
                        set_au_value($guid, $row['user_ID']);
                        get_au_checked($guid, $row['user_ID']);
                    ' />
                </td>
                <td width="200">
                    <?php
                        echo $row['name'];
                    ?></td>
                <td style="font-family: monospace;">
                    <?php
                        if ($row['administrator'] == 0) echo '';
                        elseif ($row['administrator'] == 1) echo 'LOCAL Admin';
                        elseif ($row['administrator'] == 2) echo 'Client Admin';
                        else echo 'User type unknown!
Contact your Local Administrator.';
                    ?></td>
            </tr>
        </table>
    }
function set_au_value($guid, $user_ID) {
    return "value=\"".$guid." ".user_ID."\";?
}

function get_au_checked($guid, $user_ID) {
    return "checked=\"checked\";?
}

function null_nu_vars() {
    unset($_SESSION['nu_name']);
    unset($_SESSION['nu_password1']);
    unset($_SESSION['nu_password2']);
    unset($_SESSION['nu_administrator']);
    unset($_SESSION['nu_mysql_user']);
    unset($_SESSION['nu_mysql_password1']);
    unset($_SESSION['nu_mysql_password2']);
    unset($_SESSION['nu_mysql_host']);
    unset($_SESSION['nu_submit']);?
}

function unset_svc_vars() {
    unset($_SESSION['svc']);
    unset($_SESSION['svc_download']);?
}
unset($_SESSION['svc_path']);
unset($_SESSION['svc_filename']);
unset($_SESSION['svc_shasum']);

//Unset page variables
unset($_POST['svc']);
unset($_POST['svc_download']);
unset($_POST['svc_path']);
unset($_POST['svc_filename']);
unset($_POST['svc_shasum']);

function unset_na_vars()
{
  //Unset session variables
  unset($_SESSION['na_guid']);
  unset($_SESSION['na_name']);
  unset($_SESSION['na_ip']);
  unset($_SESSION['na_port']);
  unset($_SESSION['na_private_key_loc']);
  unset($_SESSION['na_mss_user']);
  unset($_SESSION['na_sponsor']);
  unset($_SESSION['na_submit']);

  //Unset page variables
  unset($_POST['na_guid']);
  unset($_POST['na_name']);
  unset($_POST['na_ip']);
  unset($_POST['na_port']);
  unset($_POST['na_private_key_loc']);
  unset($_POST['na_mss_user']);
  unset($_POST['na_sponsor']);
  unset($_POST['na_submit']);
}

function unset_au_vars()
{
  //Unset session variables
  unset($_SESSION['au_assign']);
  unset($_SESSION['au_submit']);

  //Unset page variables
  unset($_POST['au_assign']);
  unset($_POST['au_submit']);
}

function unset_da_vars()
{
  //Unset session variables
  unset($_SESSION['del_actor']);
  unset($_SESSION['da_submit']);

  //Unset page variables
  unset($_POST['del_actor']);
  unset($_POST['da_submit']);
}
567    unset($_POST['del_actor']);
568    unset($_POST['da_submit']);
569 }
570 ?>
571
572 <? ////////////////////////////////////////////////////////////////////////////// UNSET_DU_VARS //////////////////////////////////////////////////////////////////////////
573 // unset only the DEL_USERS variables
574 function unset_du_vars()
575 {
576    //Unset session variables
577    unset($_SESSION['del_user']);
578    unset($_SESSION['du_submit']);
579
580    //Unset page variables
581    unset($_POST['del_user']);
582    unset($_POST['du_submit']);
583 }
584 ?>
585
586 <? ////////////////////////////////////////////////////////////////////////////// UNSET_DS_VARS //////////////////////////////////////////////////////////////////////////
587 // unset only the DEL_SERVICES variables
588 function unset_ds_vars()
589 {
590    //Unset session variables
591    unset($_SESSION['del_service']);
592    unset($_SESSION['del_all_service']);
593    unset($_SESSION['ds_submit']);
594
595    //Unset page variables
596    unset($_POST['del_service']);
597    unset($_POST['del_all_service']);
598    unset($_POST['ds_submit']);
599 }
600 ?>
601
602 <? ////////////////////////////////////////////////////////////////////////////// UNSET_CC_VARS //////////////////////////////////////////////////////////////////////////
603 // unset only the CHOOSE_CATEGORY variables
604 function unset_cc_vars()
605 {
606    //Unset session variables
607    unset($_SESSION['cc_category']);
608    unset($_SESSION['cc_submit']);
609
610    //Unset page variables
611    unset($_POST['cc_category']);
612    unset($_POST['cc_submit']);
613 }
614 ?>
header.php

```php
<?php
//Purpose: Connect to Users with username and password.
//Log out after 20 minutes of inactivity.
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105

//Start a new Login SESSION
```

index.php

```php
<?php
//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire( 20 );
session_start();
$inactive = 1200;
if(isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION
    }
    
    $session_life = time() - $_SESSION['start'];
    if($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION
    }
```
session_destroy();

$_SESSION['start'] = time();

// BELOW IS UNNECESSARY FOR THE INDEX.PHP PAGE
if($_SESSION['valid_user'] == true AND $_SESSION['authorized_user'] == true)
{
//
?>

<?php
// STANDARD AREA TO INCLUDE FILES AND INITIALIZE DATA
include '../scripts/header.php';
include '../scripts/connect_Users.php';
include 'functions_User.php';
connect_Users();

EXECUTE THE STORED PROCEDURE SSS_READ_PROVIDER
$_SESSION['valid_user'] = 'false';
$_SESSION['username'] = '';
$_SESSION['user_ID'] = 0;
$_SESSION['password'] = '';
$_SESSION['guid'] = '0';
$_SESSION['administrator'] = '';

// INITIALIZE USER DATA

// PROVIDE YOUR MSS CREDENTIALS:

<form method="post" action="index.php">
<table style="margin-left:3px" border="0"
cellspacing="10"
cellpadding="1">
<tr align="left">

<td width="10">
<? // Placeholder ?>

</td>
<th>
Username:
</th>
<td>
<input type="text"
name="username"
size="20">
</td>
</tr>
</form>
```php
<?php

//grab the input
$username = $_POST['username'];
$password = $_POST['password'];

//compare the input to the users in the Users database
$query = sprintf("SELECT user_ID, name, password, guid, administrator FROM User WHERE name='%s' AND password='%s'";

mysql_real_escape_string($username),
mysql_real_escape_string($password));

$result= mysql_query($query);

//Print on error...
if (!$result)
{
    $message = 'Invalid query: ' . mysql_error() . "\n";
    $message .= 'Whole query: ' . $query;
    die($message);
    // ...and make sure invalid users can't get in to other pages
    $_SESSION['valid_user'] = "false";
    session_unset();
    ?>
    Error in username or password.
    Please <A href="index.php">log in</A> again.
    <?
    }
else

~ 179 ~
//Load the rest of the user’s data into the session...
$row = mysql_fetch_array($result);

//...if they give the right credentials
if($row['name'] == $username AND $row['password'] == $password)
{
  $_SESSION['user_ID'] = $row['user_ID'];
  $_SESSION['username'] = $row['name'];
  $_SESSION['password'] = $row['password'];
  $_SESSION['guid'] = $row['guid'];
  $_SESSION['administrator'] = $row['administrator'];
  // allow a valid_user to view the website
  $_SESSION['valid_user'] = "true";
  $valid_user = "true";
}
else $_SESSION['valid_user'] = "false";

//Get the user's actors for which he or she is authorized to download services.
get_user_actors($_SESSION['user_ID']);

//Find out if the server address matches the list of sponsors the user is allowed to download from.
check_authorized_user();

$authorized_user = $_SESSION['authorized_user'];

//Show the available web pages if you are a valid, authorized user
if($valid_user == "true" and $authorized_user == "true")
{
  include 'menu.php';
  //adjusts the header
  echo "<BR>";

  ??
  get_user_services();
  $mss_location=MSS_SERVICES;
  ?>

<div id="wrap">
  ?? // Give the instructions for each page
  echo "Here are the directions for using MSS:<BR>";
  "<hr>
  <table style="margin-left:3px"
    border="0"
    cellspacing="0"
    cellpadding="4">
    <tr>
      <td align="top" width="197">
        ??
        echo "<b>Sponsor</b>";
        "
      </td>
      <div style="border: solid 0 #060;" border-color: rgb(248, 180, 66);
      border-left-width:2px;
      padding-left:0.5ex">
        ?? //
        echo "This is the actor in your Global Environment for Network Innovation (GENI) control framework.
        Your sponsor must be actively donating to GENI in order to download new services.";
        ?>
    </li>
  </div>
</div>
</td>
</tr>
<tr>
<td valign="top" width="197">

```php
<?
echo '<b>Sponsored Services</b>

<li> /*
// echo "The subset of GENI services your sponsor provides.";
?>
</li></div>
</td>
</tr>
<tr>
<td valign="top" width="197">

```php
<?
echo '<b>Choose Services</b>

<li> /*
// echo "Use this page to download services to your computer.";
?>
</li></div>
</td>
</tr>
<tr>
<td valign="top" width="197">

```php
<? if($_SESSION['administrator'] == 1 OR $_SESSION['administrator'] == 2) {
if($_SESSION['authorized_sponsor'] == "true") { ?>
<tr>
<td valign="top" width="197">

```php
<?
echo '<b>Choose Categories</b>

<li> /*
// echo "This option is only available to an authorized sponsor, which is a server that hosts a subset of its parent sponsor. The sponsor must be identified as such by using the <i>Add Sponsor</i> page. You must use the GUID of a currently donating GENI control framework actor in order to download new services from your parent sponsor. For instance, Duke
```
University might serve ORCA-related services and have many child sponsors, which in turn would be a parent sponsor of many child sponsors, and so on. A high level view of this arrangement might look like this:

```<pre>
GENI MSS CENTER --> PlanetLab --> .. --> ProtoGENI --> .. --> ORCA -- physical client1 --> physical client2 --> physical client..N --> virtual client1 --> virtual client2 --> virtual client..N --> orca-uaf-2 -- physical client1 --> physical client2 --> physical client..N --> virtual client1 --> virtual client2 --> virtual client..N --> orca-barrow-0 -- physical client1 --> physical client2 --> physical client..N --> virtual client1 --> virtual client2 --> virtual client..N</pre>
```
~ 183 ~
name one would use to remotely connect to the  
computer. For instance, one must often log in  
to VMs as user "root" or "ubuntu".  
(6) In addition:<br>

MSS DELIVERS SERVICES USING SECURE SHELL (SSH) AND  
SECURE COPY (SCP) AS THE APACHE USER 'www-data', SO  
ASSUMING YOUR MSS USER IS NAMED 'workers'  
<u>ON THE SPONSOR</u>:  
- change www-data's default shell to bash  
in /etc/passwd  
- add these groups:  
usermod -a -G workers www-data  
(allow www-data to run the shell scripts)  
usermod -a -G mss www-data  
(all MSS users belong to group mss)  
- set the UID/GID on /home/work/MSS directory to:  
chmod ug+s /home/work/MSS  
chmod 0770 /home/work/MSS  
- set the ssh key directory so www-data can access:  
chmod 0750 /home/work/.ssh  
<u>ON A CLIENT EUCALYPTUS INSTANCE</u> (such as Ubuntu 11.04):  
- as root...  
- add group mss  
addgroup mss  
usermod -a -G mss ubuntu  
- add directory "/"."$mss_location"]/MSS"  
mkdir "/"."$mss_location"]/MSS"  
chown ubuntu:mss "/"."$mss_location"]/MSS"  
<u>ON A PHYSICAL CLIENT</u>:  
- add group mss  
addgroup mss  
- add user workers if the user does not exist  
adduser workers  
usermod -a -G mss workers  
- add directory "/"."$mss_location"]/MSS"  
mkdir /usr/share/MSS  
chown workers:mss /usr/share/MSS  
<u>ON THE SPONSOR</u>:  
- su workers  
- use passwordless ssh by placing the public key in the  
authorized_users file. For instance:  
ssh-keygen  
ssh-copy-id '-p XXXX -i ~/.ssh/id_dsa.pub workers@client.ip.'  
(the public key must be in /home/workers/.ssh/authorized_keys)  
</pre>
echo "Follow these steps to establish a sponsor, from which clients will download services:<BR>
(1) Add the sponsor's <b>GUID</b>, which one can find on the <i>Sponsor</i> page.<BR>
(2) Add the sponsor's unique <b>Name</b>. For instance, the name could be its hostname, such as "orca-barrow-0" or its IP address, such as "1.2.3.4".<BR>
(3) Add the sponsor's <b>IP address</b>.<BR>
(4) Add the sponsor's <b>Port</b>. The default port is 22, but virtual machine (VM) hosts often forward a port to the VM's port 22, such as port 6001.<BR>
(5) Add the sponsor's <b>Private Key Location</b> for this actor. VMs often have a specific private key to use that is different than the sponsor's private key.<BR>
(6) Add the <b>MSS User Name</b>, which is the name one would use to remotely connect to the computer. For instance, one must often log in to VMs as user "root" or "ubuntu".<BR>
(7) In addition:<BR><pre>MSS SPONSOR IS A TYPICAL LINUX APACHE MYSQL & PHP (LAMP) SERVER
PLEASE REFER TO THE DOCUMENTATION ON THE SPONSOR SETUP</pre>";?>
</td></tr>
<tr><td valign="top" width="197">
<?
 echo "<b>Assign Users</b>";
?>
</td></tr>
<tr><td valign="top" width="197">
<?
 echo "Assign a specific user or users to a client or clients. Assign the user to the sponsor from which he or she is authorized to download services.";
?>
</td></tr>
<tr><td valign="top" width="197">
<?
 echo "<b>Delete Actors</b>";
?>
</td></tr>
<tr><td valign="top" width="197">
<?
 echo "Choose the actor or actors to delete from this
<table>
<thead>
<tr>
<th>Sponsor.</th>
<th>Delete Users</th>
<th>Delete Services</th>
<th>Log Off</th>
</tr>
</thead>
<tbody>
<tr>
<td>sponsor.</td>
<td>echo &quot;Delete Users&quot;;</td>
<td>// echo &quot;Choose the user or users to delete from this sponsor.&quot;;</td>
<td>??</td>
</tr>
</tbody>
</table>
<li><? //
  echo "Closes the current session and allows one 
  to log in again or as a different user."
?>
</li></div>
</td>
</tr>

<?php
//refresh the page to load user services into the svc array
if (!isset($_SESSION['svc']))
{
  //Create an array to hold the user's services in SESSION
  $_SESSION['svc'] = array();
  //Get the user's services
  get_user_services();
}
?>

<?php
echo "</table>";
echo "</div>";
include('../scripts/footer.php');
echo "</div>";
?>
list_actors.php

<?php
//Purpose: Connect to the Services database as user "workers"
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105
?>

<?
//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start']))
{
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive)
    {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $SESSION;
        session_destroy();
    }
}
//Set the session start time
$_SESSION['start'] = time();

////////////// START OF PAGE ///////////////

if ($_SESSION['valid_user'] == true
AND $_SESSION['authorized_user'] == true)
{
    ?>
    include '../scripts/connect_Users.php';
    include '../scripts/header.php';

    connect_Users();

    // Retrieve the Actor List
    $query = "SELECT * FROM Actor";
    $result = mysql_query($query);

    echo "<h3><center>Current Actors</center></h3>";
    //Display the menu ribbon
    include 'menu.php';
    //adjusts the header
    echo "<BR>";
    ?>
    <div id="wrap">

    <? echo "List of actors that are authorized to download
    from the sponsor:<BR>" ?>
    <HR>
    <!--Create a table of current Services-->
    <table style="margin-left:25px; table-layout: auto;"
    border="0"
cellspacing="10"

    ~ 188 ~
<table>
<thead>
<tr>
<th>Actor</th>
<th>Actor GUID</th>
<th>IP Address</th>
<th>Port</th>
<th>Private Key Location</th>
<th>MSS User Name</th>
<th>Sponsor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```php
while ($row = mysql_fetch_array($result)) {
    <tr>
        <td><? echo $row['name']; ?></td>
        <td><? echo $row['guid']; ?></td>
        <td><? echo $row['ip']; ?></td>
        <td><? echo $row['port']; ?></td>
        <td><? echo $row['private_key_loc']; ?></td>
        <td><? echo $row['mss_user']; ?></td>
        <td align="center">
            <?
                $ip_explode = explode(":", $_SERVER['HTTP_HOST']);
                if ($row['sponsor'] == 1 and ($ip_explode[0] == $row['ip']))
                    echo "LOCAL";
                elseif ($row['sponsor'] == 1)
                    echo "Client";
                else echo "No";
            ?></td>
    </tr>
}</h2>
</div>

```
list_services.php

<?php

//Purpose: Connect to the Services database as user "workers"
//Author:  John P. Quan
//Version: 1.0
//Date:    20120105
?>

<?
//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

inactive = 1200;
if (isset($_SESSION['start'])) {
    session_life = time() - $_SESSION['start'];
    if (session_life > $inactive) {
        header("Location: user_logout.php");
        $SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $SESSION;
        session_destroy();
    }
}
//Set the session start time
$_SESSION['start'] = time();

//////// START OF PAGE ///////////

if ($_SESSION['valid_user'] == true
    AND $_SESSION['authorized_user'] == true)
{
    ?><
    include '../scripts/connect_Services.php';
    include '../scripts/header.php';
    connect_Services();
    // Retrieve the Services List
    $query = "SELECT * FROM Services";
    $result = mysql_query($query);
    $num = mysql_numrows($result);
    echo "<h3><center>Current Services</center></h3>";
    //Display the menu ribbon
    include 'menu.php';
    //adjusts the header
    echo "<BR>";
    ?>
    <div id="wrap"高于在"wrap"

<? echo "List of services on the sponsor:<BR>" ?>

<table style="margin-left:25px; table-layout: auto;"
    border="0"
cellsapcing="10"
cellpadding="1"
width="97%">

~ 190 ~
<h2>
<tr>
<th align="left">Service ID</th>
<th align="left">Service</th>
<th align="left">File Name</th>
<th>Description</th>
<th>SHA Sum</th>
<th align="left">Developer</th>
<th align="left">Publisher</th>
</tr>
<?
$i = 0;
while ($i < $num) {
    $service_ID = mysql_result($result, $i, "service_ID");
    $name = mysql_result($result, $i, "name");
    $filename = mysql_result($result, $i, "filename");
    $description = mysql_result($result, $i, "description");
    $shasum = mysql_result($result, $i, "shasum");
    $developer = mysql_result($result, $i, "developer");
    $publisher = mysql_result($result, $i, "publisher");
    ?><td valign="top" align="center" width="90"><? echo $service_ID; ?></td>
    <td valign="top"><? echo $name; ?></td>
    <td valign="top"><? echo $filename; ?></td>
    <td valign="top"><? echo $description; ?></td>
    <td valign="top"><? echo $shasum; ?></td>
    <td valign="top"><? echo $developer; ?></td>
    <td valign="top"><? echo $publisher; ?></td>
<?
}$i++;
}echo "</table>";
echo "</div>";
mysql_close(connect_Services());
include('../scripts/footer.php');
echo "</div>
<?php
///// END OF PAGE //////////////
else {
include('../scripts/header.php');
include ('../scripts/footer.php');
~ 191 ~
Either you are not allowed to access this page, or your session has expired. Please [log in](index.php) again.

```php
<?php
//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION
        $_SESSION;
        session_destroy();
    }
}
//Set the session start time
$_SESSION['start'] = time();

~/~
<?
include '../scripts/connect_orca.php';
include '../scripts/header.php';
connect_orca();

// Retrieve the ORCA Sponsor
$query = "SELECT act_id,
        act_name,
        act_guid
FROM Actors";
$result = mysql_query($query);

echo "<h3><center>Sponsor</center></h3>";
// Display the menu ribbon
include 'menu.php';
// Adjusts the header
echo "<BR>

<?
<? echo "List of sponsors through which local users can download new services:<BR>" ??
<br>
<!-- Create a table of current Services -->
<table style="margin-left:25px; table-layout: auto;"
border="0"
cellspacing="10"
cellpadding="1">
<tr>
<th>Actor ID</th>
<th align="left">Sponsor</th>
<th>GUID</th>
</tr>

<?
while ($row = mysql_fetch_array($result))
{
  ??
  <tr>
    <td><center><? echo $row['act_id']; ?></center></td>
    <td><? echo $row['act_name']; ?></td>
    <td><center><? echo $row['act_guid']; ?></center></td>
  </tr>
</tr>
</h2>
</div id="foot">

<?
echo "</table>

<?
echo "</div>

mysql_close(connect_orca());
include('../scripts/footer.php');
echo "</div>

<?php
Either you are not allowed to access this page, or your session has expired. Please <A href="index.php">log in</a> again.

<?php

session_cache_expire(20);
session_start();

ini_set('session.gc_maxlifetime', 20); // Set the session.gc_maxlifetime to 20 seconds

if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
        $_SESSION['valid_user'] = 'false';
        //CLOSE PREVIOUS SESSION*
        $_SESSION = array();
        session_destroy();
    }
}

~ 194 ~
//Set the session start time
$_SESSION['start'] = time();

///////// START OF PAGE /////////////////
if ($_SESSION['valid_user'] == true
    AND $_SESSION['administrator'] == 1) {

<?
include '../scripts/connect_Users.php';
include '../scripts/header.php';
connect_Users();
// Retrieve the Services List
$query = "SELECT * FROM User";
$result = mysql_query($query);
$num = mysql_num_rows($result);
echo "<h3><center>Current Users</center></h3>";
//Display the menu ribbon
include 'menu.php';
//adjusts the header
echo "<BR>

<?
<div id="wrap"

<? echo "List of users who are authorized to download
from the sponsor:<BR>" ?>
<HR>
<table style="margin-left:25px; table-layout: auto;"
    border="0"
    cellspacing="10"
    cellpadding="1" >
<h2>
<tr>
<th align="left">User ID</th>
<th align="left">Name</th>
<th>GUID</th>
<th align="left">Administrator</th>
</tr>

<?
$i = 0;
while ($i < $num) {

$user_ID = mysql_result($result, $i, "user_ID");
$name = mysql_result($result, $i, "name");
$guid = mysql_result($result, $i, "guid");
$administrator = mysql_result($result, $i, "administrator");
if ($administrator == 0) $administrator = 'No';
elseif ($administrator == 1) $administrator='LOCAL';
elseif ($administrator == 2) $administrator='Client';
else $administrator='User type unknown!
    Contact your Local Administrator.';

<?
<td align="center">
    <? echo $user_ID; ?></td>
<td><? echo $name; ?></td>
</td><td><? echo $guid; ?></td>
<td align="center">
<? echo $administrator; ?></td>
</tr>
</table>
</div>

<?
$i++;
}
</div>
</table>
</div>

//******/ END OF PAGE //*************
}
else {

include ('../scripts/header.php');

<!--Print error message and offer to log in again-->
Either you are not allowed to access this page, or your session has expired.
Please &lt;A href="index.php">log in</a> again.

<?php } ?>
menu.php

```php
<?php

//Purpose: Acts as a menu bar for Available pages
//Author: John P. Quan
//Version: 1.0
//Date: 20120105

if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
    }
}

//STANDARD SESSION LIFE AND INACTIVITY CHECK
session_cache_expire(20);
session_start();

$inactive = 1200;
if (isset($_SESSION['start'])) {
    $session_life = time() - $_SESSION['start'];
    if ($session_life > $inactive) {
        header("Location: user_logout.php");
    }
    $_SESSION['valid_user'] = 'false';
    //CLOSE PREVIOUS SESSION*
    session_destroy();
}

//Set the session start time
$_SESSION['start'] = time();

include_once 'functions_php.php';

///////// START OF PAGE ///////////

if ($_SESSION['valid_user'] == true AND $_SESSION['authorized_user'] == true) {
    <? echo "&nbsp&nbsp";
    //everyone sees these pages
    ?><A HREF="index.php">Instructions</A> -
    <A HREF="list_sponsor.php">Sponsor</A> -
    <A HREF="list_services.php">Sponsored Services</A> -
    <A HREF="choose_service.php">Choose Services</A> -
    <!--only show the administrators these pages
    if ($_SESSION['administrator'] > 0) {
        <!--only show choose_category for authorized_sponsors
        if ($_SESSION['authorized_sponsor'] == "true") { ?>
        <A HREF="choose_category.php">Choose Categories</A> -
        <? }
    }
    //Only the local administrators see these pages
    if ($_SESSION['administrator'] == 1) {
        <!--phpinfo-FOR TESTING ONLY!"</A> -
        <A HREF="list_users.php">List Users</A> -
        <A HREF="list_actors.php">List Actors</A> -
        <A HREF="add_user.php">Add User</A> -
        <A HREF="add_client.php">Add Client</A> -
        <A HREF="add_sponsor.php">Add Sponsor</A> -
    }<?
    //Do not show assign_users.php for the text-based browser W3M
```
// because it does not interpret the page
// correctly. Administrators can log in from his or her computer

if(key_array_search("w3m", $_SERVER) == FALSE) { ?>
   <A HREF="assign_users.php">Assign Users</A> -
   <A HREF="del_actors.php">Delete Actors</A> -
   <A HREF="del_users.php">Delete Users</A> -
   <A HREF="del_services.php">Delete Services</A> -
   <A HREF="user_logout.php">Log Off</A>
<?php
</br>
</br>
<?php
<?

          /*
          *  Document : style.css
          *  Created on: Jan 8, 2012, 7:17:41 PM
          *  Author    : johnpquan
          *  Description: Provides a style for web pages.
          */
          */

* {
* margin:0;
* padding:0;
* }
html, body {
* height:100%;
* background:rgb(128, 187, 213);
* }
user_logout.php

<?php

//Purpose: Show the user as logged out of the session
//Author: John P. Quan
//Version: 1.0
//Date: 20120105
//SESSION has ended

include '../scripts/header.php';

echo "User logged out of session.";

session_start();
session_unset();
session_destroy();
session_write_close();
setcookie(session_name(), '', 0, '/');
session_regenerate_id(true);

?>

Please <A href="index.php">log in</a> again.
BASH Scripts

sh_check_heartbeats
1  #!/bin/bash
2  
3  # Download the ORCA Actors Registry
4  # amdiff is greater than 0 if the actor is NOT donating,
5  # or if the actor does not exist.
6  
7  rm actors.jsp
8  
9  ACTORS=`wget --no-check-certificate $1; egrep '(url:\|amdiff:\)' $2 | cut -d'\'' -f2; chown workers:workers $2`
10 
11  echo $ACTORS

sh_rsync_command
1  #!/bin/bash
2  
3  # Use SSH to send rsync commands, such as downloading the service,
4  # because this is often the only port open for Eucalyptus
5  # instances (VMs).
6  
7  OPTS="-q -o PreferredAuthentications=publickey -o HostbasedAuthentication=no -o PasswordAuthentication=no -o StrictHostKeyChecking=no"
8  
9  USER=$1
10  IP=$2
11  PORT=$3
12  KEY=$4
13  FILE=$5
14  LOC=$6
15  
16  RETURN=`rsync --delete --log-file=../scripts/logs/rsync/$(date +%Y%m%d)_mss_rsync.log -e 'ssh ${OPTS} -i ${KEY} -p ${PORT} -l ${USER}'' -avzp ${FILE} ${IP}:${LOC}`
17  
18  echo $RETURN

sh_ssh_command
1  #!/bin/bash
2  
3  # Use SSH to send commands, such as downloading the service,
4  # because this is often the only port open for Eucalyptus
5  # instances (VMs).
6  
7  OPTS="-q -o PreferredAuthentications=publickey -o HostbasedAuthentication=no -o PasswordAuthentication=no -o StrictHostKeyChecking=no"
8  
9  USER=$1
10  IP=$2
11  PORT=$3
12  KEY=$4
13  COMMAND=$5
14  
15  RETURN=`ssh ${OPTS} -i ${KEY} -p ${PORT} ${USER}@${IP} ${COMMAND}`
16  
17  echo $RETURN

~ 201 ~
Appendix C

Hypervisor
The following sections provide the files from UAF’s remote server, orca-barrow-0, with only the uncommented lines displayed because the commented lines have no affect on the system’s configuration. In addition, any passwords, encryption keys, or other security information is masked or changed. orca-barrow-0 is a single server with the Linux distribution Debian 6 and paravirtualized Xen installed, and that resides in a server room in Barrow, AK. The files consider its public IP address to be “public.IP.address,” and it is private, internal IP address ranges as displayed in Figure 2. ORCA Remote Server.

Networking

/etc/network/interfaces

auto lo
iface lo inet loopback

allow-hotplug eth0
iface eth0 inet static
    address 192.168.1.100
    netmask 255.255.255.0
    gateway 192.168.1.1
    network 192.168.1.0
    broadcast 192.168.1.255

allow-hotplug eth1
iface eth1 inet static
    address 192.168.1.10
    netmask 255.255.255.0
    broadcast 192.168.1.255

auto dummy0
iface dummy0 inet manual
    pre-up ifconfig $IFACE up
    post-down ifconfig $IFACE down

auto xenbrdummy0
iface xenbrdummy0 inet static
    address 10.10.10.1
    network 10.10.10.0
    netmask 255.255.255.0
    broadcast 10.10.10.255
    bridge_ports dummy0
    bridge_stp off
    bridge_fd 0
    bridge_maxwait 0

Xen

/etc/xen-tools/xen-tools.conf

lvm = orca-barrow-0
install-method = debootstrap

size   = 3Gb   # Disk image size.
memory = 512Mb # Memory size
swap = 1Gb  # Swap size
fs = ext3    # use the EXT3 filesystem for the disk image.
dist = `xt-guess-suite-and-mirror --suite`  # Default distribution to install.
image = sparse  # Specify sparse vs. full disk images.
gateway = public.IP.address
netmask = 255.255.255.0
broadcast = public.IP.broadcast.address

passwd = 1
kernel = /boot/vmlinuz-`uname -r`
initrd = /boot/initrd.img-`uname -r`

arch = amd64

mirror = `xt-guess-suite-and-mirror --mirror`

ext3_options = noatime,nodiratime,errors=remount-ro
ext2_options = noatime,nodiratime,errors=remount-ro
xfs_options = defaults
reiserfs_options = defaults
btrfs_options = defaults

/etc/xen/xend-config.sxp
(xend-http-server yes)
(xend-unix-server yes)
(xend-unix-path /var/lib/xend/xend-socket)
(xend-address localhost)

(network-script network-ORCA)
(vif-script vif-ORCA)

(dom0-min-mem 196)
(enable-dom0-ballooning yes)
(total_available_memory 0)
(vncpasswd '')

/etc/xen/orca-barrow-0.cfg
kernel = '/boot/vmlinuz-2.6.32-5-xen-amd64'
ramdisk = '/boot/initrd.img-2.6.32-5-xen-amd64'
vcpus = '1'
memory = '512'
root = '/dev/xvda2 ro'
disk = [
    'phy:/dev/orca-barrow-0/dnat-server-disk,xvda2,w',
    'phy:/dev/orca-barrow-0/dnat-server-swap,xvda1,w',
]
name = 'orca-barrow-0'
vif = ['ip=192.168.1.1, mac=00:16:3E:7B:6E:12, bridge=eth0',
      'ip=public.IP.address, mac=00:16:3E:7B:6E:11, bridge=eth1']
on_poweroff = 'destroy'
on_reboot = 'restart'
on_crash = 'restart'
on_xend_start = 'start'
on_xend_stop = 'shutdown'

/etc/xen/scripts/vif-ORCA
#!/bin/sh

dir=$(@dir)
IFNUM=${vif} | awk -F. '{ print $2 }'

if [[ "$IFNUM" == "0" ]]; then
  "$dir/vif-route" "$@"
else
  "$dir/vif-bridge" "$@"
fi

/etc/xen/scripts/network-ORCA
#!/bin/sh
/etc/xen/scripts/network-bridge "$@" vifnum=0 netdev=eth0
/etc/xen/scripts/network-bridge "$@" vifnum=1 netdev=eth1

ORCA

/opt/Camano-3.1/config/ec2.site.properties
ec2.img.proxy.use=true
ec2.img.proxy.url=http://192.168.1.100/axis2/services/IMAGEPROXY
ec2.img.proxy.timeout=3600
ec2.ami.name=emi-499D16C3
ec2.aki.name=eki-A3B01BDD
ec2.ari.name=eri-991917F8
ec2.instance.type=m1.small
ec2.ssh.key=barrowkey
ec2.use.public.addressing=true
ec2.connection.timeout=60
ec2.request.timeout=120
ec2.ping.retries=60
ec2.ssh.retries=10
ec2.startup.retries=5
ec2.use.proxy=true
proxy.type=SHOREWALL-DNAT
proxy.proxy.ip=192.168.1.1
proxy.user=orca
proxy.ssh.key=/opt/Camano-3.1/config/orca-proxy-ssh-key
proxy.script.path=/opt/iptables-scripts

Initialization Scripts

/etc/init.d/start_ORCA.sh
  case "$1" in
    start)
      echo -n "Starting orca-barrow-0 in 2 minutes"
      #To run it as root:
      sleep 10
      xm destroy orca-barrow-0
      sleep 4
      xm create /etc/xen/orca-barrow-0.cfg
  ~ 204 ~
sleep 10
/opt/eucalyptus-2.0/etc/init.d/eucalyptus-cc start
/opt/eucalyptus-2.0/etc/init.d/eucalyptus-nc start
/etc/init.d/eucalyptus-cloud start
cd /opt/Camano-3.1/tomcat-7; ./start.sh
cd /opt/imageproxy/axis2-1.5.4/bin; nohup ./axis2server.sh &
cd /opt/imageproxy/bin; nohup ./seeding /opt/imageproxy
/opt/imageproxy/axis2-1.5.4/imageproxy.db > /opt/imageproxy/logs/seeding.log &
> .

stop)
    echo -n "Stopping orca-barrow-0"
    #To run it as root:
    cd /opt/Camano-3.1/tomcat-7; ./stop.sh
    sleep 4
    /etc/init.d/eucalyptus-cloud stop
    sleep 6
    /opt/eucalyptus-2.0/etc/init.d/eucalyptus-cc stop
    sleep 3
    /opt/eucalyptus-2.0/etc/init.d/eucalyptus-nc stop
    sleep 10
    xm destroy orca-barrow-0
    echo "."
    ;;

)*
    echo "Usage: /sbin/service start_ORCA.sh {start|stop}"
    exit 1

esac

exit 0

Virtual Router
The following sections provide the files from UAF’s remote server, orca-barrow-0, with only the uncommented lines displayed because the commented lines have no affect on the system’s configuration. In addition, any passwords, encryption keys, or other security information is masked or changed. orca-barrow-0 is a single server that resides in a server room in Barrow, AK. The files consider its public IP address to be “public.IP.address,” and it is private, internal IP address ranges as displayed in Figure 2. ORCA Remote Server.

ORCA
A single-server remote server installation is very similar to a canonical ORCA cluster behind a DNAT Server [1], except that it does not use Shorewall to control iptables on the DNAT Server. Instead, the remote server uses modified RENCI scripts to control iptables on the virtual DNAT Server directly. Thank you to Dr. Brian Hay, University of Alaska Fairbanks, for modifying the /opt/iptables-scripts files.

/opt/iptables-scripts/README.txt
These scripts were created to circumvent using Shorewall DNAT server, as RENCI prescribes. In order to "fit" an entire ORCA installation on one server, UAF ORCA architecture uses a Virtual Machine named "dnat-server" as a router, which is turned on in the host server's startup sequence. This set up seems to create problems for Shorewall, and so we created this work-around by modifying /opt/shorewall/dnat-scripts created by RENCI. Instead of using shorewall to modify the dnat-server iptables, these scripts do so directly. Like Shorewall, iptables-scripts write a rule into iptables and then restarts iptables.
Unlike Shorewall, iptables is first populated with DNAT and SNAT rules so that Dom0, which contains ORCA, Eucalyptus, and Image Proxy, can communicate with the dnat-server DomU. The virtual dnat-server then routes these communications to the Internet. Please refer to /etc/iptables.rules on the dnat-server for an example.

```
# /opt/iptables-scripts/execCmd.sh
set -e

SCRIPT=`readlink -f $0`
INSTALLPATH=`dirname $SCRIPT`

. $INSTALLPATH/lib/helpers.sh
. $INSTALLPATH/lib/iptables-helpers.sh

if [ "$1" = "" ] || [ "$2" = "" ] || [ "$3" = "" ]; then
  echo "STATUS=ERROR; MSG="Expecting [ADD|DEL] ip port"
  exit 1
fi

CMDCONT="$1 $2 $3"
RES=$(cmdHelper $CMDCONT) || {
  echo "STATUS=ERROR; $RES"
  return 1
}

```

```
# /opt/iptables-scripts/lib/iptables-helpers.sh

DNAT_FIRST_PORT=6001
DNAT_LAST_PORT=6999
PUBIP="public.IP.address"
SHOREWALL=`which shorewall-wrapper`
DNAT_CHAIN="PREROUTING"
RULESFILE="/tmp/rules"

function cmdHelper () {
  CMD=$1
  HOST=$2
  HOSTPORT=$3
  RES="NOTSET"

  if [ "$CMD" = "ADD" ]; then
    RES=$(addDNATRule $HOST $HOSTPORT) || {
      echo "$RES"
      return 1
    }
    echo "$RES"
    return 0
  fi

  if [ "$CMD" = "DEL" ]; then
    RES=$(delDNATRule $HOST $HOSTPORT) || {
      echo "$RES"
      return 1
    }
    echo "$RES"
    return 0
  fi

  echo "Invalid syntax"
  return 1
}

function getPorts () {

```
if [ "$1" == "" ]; then
    return 1
fi
PORTS=`sudo /sbin/iptables -t nat -n -L $1 2>/dev/null| gawk 'BEGIN { FS = "[ \	:]+" }$7 ~ /dpt/ {print $8}'` || {
    return 1
}
echo $PORTS
return 0

function findFreeDnatPort () {
    USEDPORTS=$(getPorts $DNAT_CHAIN) || {
        return 1
    }
    FREEPORT=`echo "$1" | gawk "v used="$USEDPORTS" v first="$DNAT_FIRST_PORT" v last="$DNAT_LAST_PORT" BEGIN { RS=" "; }
{n=sprintf(arr, " ")
asort(arr)
port=first
found=-1
i=1
while(found<first && port<=last && i<=n) {
    if (arr[i]<port) {
       i++
    } else if (port==arr[i]) {
       i++
       port++
    } else {
       found=port
    }
}
print found
}'`
test $FREEPORT -eq -1 && {
    echo "$1"
    return 1
}
echo $FREEPORT
return 0

function checkIP () {
    RES=`echo $1 | gawk 'BEGIN { RS=" " }
$0 ~ /[0-9]+/ { print $0 }'` || {
        return 1
    }
    echo "$1"
    return 0
}

function checkPort () {
    RES=`echo "$1" | gawk -v port=$1 'BEGIN { RS=" " }$
    (port>0 && port < 65535)
    print port
)'

~ 207 ~
if [ "$RES" = "" ]; then
    return 1
fi
echo $RES
return 0
}

function checkDNATRule () {
    #FN=$RULESFILE
    HOST=$1
    R=$(checkIP $HOST) || {
        echo "Invalid host $HOST"
        return 2
    }
    HOSTPORT=$2
    R=$(checkPort $HOSTPORT) || {
        echo "Invalid port $HOSTPORT"
        return 2
    }
    #PAT="DNAT\n$DNAT_CHAIN:$HOST:$HOSTPORT"
    #RES=`gawk -v pat="$PAT" '$0 ~ pat { print $5 }' $FN'
    PAT="DNAT.*dpt:[0-9]* to:$HOST:$HOSTPORT"
    RES=`sudo /sbin/iptables -t nat -L $DNAT_CHAIN -n | gawk -v pat="$PAT" '$0 ~ pat { print $7 }' | gawk -F ":" '{ print $2 }'` || {
        if [ "$RES" = "" ]; then
            return 0
        else
            echo $RES
            return 1
        fi
    }
}

function addDNATRule () {
    FN=$RULESFILE
    HOST=$1
    R=$(checkIP $HOST) || {
        echo "MSG="Invalid host $HOST"
        return 2
    }
    HOSTPORT=$2
    R=$(checkPort $HOSTPORT) || {
        echo "Invalid port $HOSTPORT"
        return 2
    }
    lock_file $FN
    # check if rule exists
    RES=$(checkDNATRule $HOST $HOSTPORT) || {
        echo "MSG="RULE EXISTS"; HOST=$HOST; PORT=$HOSTPORT;
        PUBIP=$PUBIP; FWDPORT=$RES
        unlock_file $FN
        return 0
    }
    # find free port
    LOCPORTE=$(findFreeDnatPort) || {
        echo "MSG="Unable to find free port"
        unlock_file $FN
        return 1
    }
    #echo "DNAT net $DNAT_CHAIN:$HOST:$HOSTPORT tcp $LOCPORTE" >> $FN

    ~ 208 ~
```bash
sudo /sbin/iptables -t nat --append $DNAT_CHAIN --destination $PUBIP --
proto tcp --dport $LOCPORT -j DNAT --to-destination $HOST:$HOSTPORT
unlock_file $FN
  echo "MSG=""ADDED""; HOST=$HOST; PORT=$HOSTPORT; PUBIP=$PUBIP;
  FWDPORT=$LOCPORT"
  return 0
}
function delDNATRule ()
{
  FN=$RULESFILE
  HOST=$1
  R=$(checkIP $HOST) || {
    echo "MSG=""Invalid host $HOST"
    return 2
  }
  HOSTPORT=$2
  R=$(checkPort $HOSTPORT) || {
    echo "MSG=""Invalid port $HOSTPORT"
    return 2
  }
  lock_file $FN
  RULES_TO_DELETE=`sudo /sbin/iptables
  -t nat -L $DNAT_CHAIN -n | grep -v "^Chain" | grep -v "^target" | grep -n "DNAT.*$PUBIP.*dpt:[0-9]*
to:$HOST:$HOSTPORT" | gawk -F ":" '{ print $1 }' | sort -r`
  #echo $RULES_TO_DELETE
  if [ "$RULES_TO_DELETE" = "" ]; then
    echo "MSG=""NO RULE TO DELETE""; HOST=$HOST; PORT=$HOSTPORT"
  else
    for r in $RULES_TO_DELETE; do
      sudo /sbin/iptables -t nat --delete $DNAT_CHAIN $r
      echo "MSG=""DELETED""; HOST=$HOST; PORT=$HOSTPORT"
    done
  fi
  unlock_file $FN
  #PAT="DNAT\t\tnet\t\t$DNAT_CHAIN:$HOST:$HOSTPORT"
  #lock_file $FN
  #FNTMP=/tmp/`basename $FN`.new
  #cat $FN | sed '/^'$PAT'/{ d }' > $FNTMP
  #cp $FNTMP $FN
  #rm -f $FNTMP
  #unlock_file $FN
  #echo "MSG=""DELETED""; HOST=$HOST; PORT=$HOSTPORT"
  return 0
}
function kickShorewall ()
{
  #$SHOREWALL restart > /dev/null 2>&1 || {
    # note that refresh does not invalidate existing connections,
    # so even the deleted connection stays on
    #return 1
  #}
  return 0
}
```
Networking

/etc/network/interfaces
auto lo
  iface lo inet loopback
auto eth0
  iface eth0 inet static
    address 192.168.1.1
    broadcast 192.168.1.255
    netmask 255.255.255.0
auto eth1
  iface eth1 inet static
    # networking for Barrow, AK
    address public.IP.address
    netmask public.IP.netmask
    broadcast public.IP.broadcast.address
    gateway public.IP.gateway

/etc/iptables.rules
*nat
  :PREROUTING ACCEPT [5:440]
  :POSTROUTING ACCEPT [0:0]
  :OUTPUT ACCEPT [0:0]
-A PREROUTING -d public.IP.address/32 -p tcp -m tcp --dport 12080 -j DNAT --to-destination 192.168.1.100:12080
-A PREROUTING -d public.IP.address/32 -p tcp -m tcp --dport 11080 -j DNAT --to-destination 192.168.1.100:11080
-A PREROUTING -d public.IP.address/32 -p tcp -m tcp --dport 11443 -j DNAT --to-destination 192.168.1.100:11443
-A PREROUTING -d public.IP.address/32 -p tcp -m tcp --dport 11081 -j DNAT --to-destination 192.168.1.100:11081
-A PREROUTING -d public.IP.address/32 -p tcp -m tcp --dport 8443 -j DNAT --to-destination 192.168.1.100:8443
-A PREROUTING -d public.IP.address/32 -p tcp -m tcp --dport 12443 -j DNAT --to-destination 192.168.1.100:12443
-A PREROUTING -d public.IP.address/32 -p tcp -m tcp --dport 222 -j DNAT --to-destination 192.168.1.100:22
-A POSTROUTING -s 192.168.1.100/32 -j SNAT --to-source public.IP.address
-A POSTROUTING -s 192.168.1.101/32 -j SNAT --to-source public.IP.address
-A POSTROUTING -s 192.168.1.102/32 -j SNAT --to-source public.IP.address
-A POSTROUTING -s 192.168.1.103/32 -j SNAT --to-source public.IP.address
-A POSTROUTING -s 192.168.1.104/32 -j SNAT --to-source public.IP.address
COMMIT
*filter
  :INPUT ACCEPT [3083:279267]
  :FORWARD ACCEPT [96899:11149697]
  :OUTPUT ACCEPT [18425:2683358]
-A INPUT -m state --state ESTABLISHED -j ACCEPT
-A INPUT -d public.IP.address/32 -p tcp -m tcp --dport 22 -j ACCEPT
COMMIT

Initialization scripts

/etc/init.d/start_iptables.sh
  case "$1" in
    start)
      echo -n "Starting iptables"
      sleep 10
      iptables-restore /etc/iptables.rules
      echo "."
  esac
stop)
  echo -n "Stopping dnat-server"
  /sbin/iptables --flush
  /sbin/iptables --delete-chain
  /sbin/iptables -t nat --flush
  /sbin/iptables -t nat --delete-chain
  echo ".";
  ;

*)
  echo "Usage: /sbin/service start_iptables.sh {start|stop}"
  exit 1
esac

exit 0

/etc/init.d/send_mailPowerLoss.sh
  case "$1" in
    start)
      echo -n "Mailing Administrator about power interruption"
      sendmail -v jquan2@alaska.edu < /etc/mail/ORCA_RESTARTED_MSG.txt
      echo ".";
      ;;
    stop)
      echo -n "Stopping send_mailPowerLoss.sh"
      /etc/init.d/sendmail stop
      echo ".";
      ;;
  *)
    echo "Usage: /sbin/service send_mailPowerLoss.sh {start|stop}"
    exit 1
esac

exit 0
Volume IV
Software Testing for Mutualistic Software Services (MSS) Version 1.0
1. Introduction

This volume tests MSS according to the requirements in *Software Architecture for Mutualistic Software Services (MSS) Version 1.0*, which were synthesized from the *Software Requirements Specification for Mutualistic Software Services (MSS) Version 1.0*. The tests compare the intended actions listed in 5.3 Component Model of the Software Architecture document with the actual outcome when using MSS. Section 3. Tested Components below presents screenshots where appropriate as an example of the test case.

2. Test Criteria

Testing will use the relationship tree provided in the software architecture’s section 5. Architectural Plan as a testing template. Here is the testing relationship tree:

```
The ORCA Actor Registry (node 1)  
  ├── Heartbeats  
  │   ├── UAF - orca-uaf-2 (node 2)  
  │   │   └── Heartbeats  
  │   │       └── Barrow, AK - orca-barrow-0 (node 3)  
  │   │           ├── Admin Barrow  
  │   │           │   └── John Quan’s MacBook (physical)  
  │   │           │       ├── John Quan  
  │   │           │       │   └── 199.165.126.136-6001 (virtual)  
  │   │           │       │       └── User Barrow  
  │   │           │       └── John Quan’s MacBook (physical)  
  │   │           │                   └── John Quan  
  │   │           └── 199.165.76.87 (virtual)  
  │                   └── User Uaf  
```

In addition, testing must evaluate the following basic criteria:

- Work on multiple browsers, to include the text-based browser w3m
- Work on a canonical ORCA cluster
- Work on a Remote Server
- Support virtual resources
- Support physical resources
- Check for heartbeats at the MSS-ORIGIN
- Only allow MSS access to authorized users on authorized computers

The *Software Architecture for Mutualistic Software Services (MSS) Version 1.0*, Appendix B contains the component diagram. The component diagram CM-1 represents the external connections, and the data flow model DM-1 represents the generation and flow of data for each MSS-ENTITY.
3. Tested Components

3.1 Databases

3.1.1 Service Data
- Holds the subset of service attribute data of its MSS-ENTITY
  - See the Services Database in the Software Design for Mutualistic Software Services (MSS) Version 1.0.

3.1.2 CF Data
- Holds the MSS-ENTITY data for itself. For instance, ORCA uses database orca

3.1.3 User Data
- Holds the authorized user’s data for itself and its child MSS-AFFILIATES (actors)
  - See the Users Database in the Software Design for Mutualistic Software Services (MSS) Version 1.0.

3.2 Components

3.2.1 CF
- MSS-ORIGIN presents an interface to its children to receive heartbeats
  - MSS-AFFILIATES deliver heartbeats to its MSS-ORIGIN
  - See Figure 3. The ORCA Actor Registry.
- Provides a CF interface with a list of current donors. For instance, ORCA provides the ORCA Actor Registry [18]
  - See Figure 3. The ORCA Actor Registry.
- Provides a list of available resources to GENI experimenters and users
  - See Figure 3. The ORCA Actor Registry.
- Enables GENI experimenters and users to request resources
3.2.2 User Interface

- Uses the Authentication process to verify actor and user requests

Figure 4. MSS does not interfere with CF functionality.

Figure 5. The administrator must authorize clients
Figure 6. The administrator must specify a sponsor.

Figure 7. The administrator must authorize users.
Figure 8. The administrator assigns users to actors.
• Assign Users does not display for text-based web browsers, but it does for graphical web browsers. This is only a partial fulfillment of the requirement, but it does not seem to detract from the functionality of MSS. MSS incorporates text-based web browsing functionality for Eucalyptus Instance users and not administrators.

Figure 9. "Assign Users" does not display for w3m.
Figure 10. MSS only allows authorized users on authorized computers.

Figure 11. w3m version of "not authorized" response.
Figure 12. MSS only accepts authorized users.

Figure 13. The administrator can remove actor authorization.
Figure 14. The administrator can remove user authorization.

Figure 15. MSS displays authorized users to the administrator.

- Receives a list of MSS-ENTITY resources from the CF
• Receives user service information and service requests
Figure 18. MSS incorporates user session expiration.

- Provides service delivery information to the Service Interface

NOTE: All users are logged out of the system and the session is destroyed after 20 minutes of inactivity.
Figure 19. The Download Service page provides success/failure information.

- Triggers the Service Interface to deliver services
  - See Figure 21. User can request to download service ID 1
- Checks for heartbeats on the CF interface
  - See the `sh_check_heartbeats` command in the *Software Design for Mutualistic Software Services (MSS) Version 1.0*.
- Triggers the Service Interface to download services.
  - See Figure 21. User can request to download service ID 1

### 3.2.3 Service Interface

- Receives a service information request from the User Interface
Figure 20. Administrators choose a subset of services to advertise.

- Receives a service download request from the User Interface

Figure 21. User can request to download service ID 1
Figure 22. All users can uninstall services, but only administrators can do so immediately.

Figure 23. MSS maintains permissions on "Delete Services"

- Uses the Encryption process to encrypt services
- See the Bash Scripts `sh_rsync_command` and `sh_ssh_command` in the *Software Design for Mutualistic Software Services (MSS) Version 1.0*.
- Uses the Decryption process to decrypt services
  - See the Bash Scripts `sh_rsync_command` and `sh_ssh_command` in the *Software Design for Mutualistic Software Services (MSS) Version 1.0*.
- Uses the Validation process to validate services
  - See the `compare_shasum()` function in `functions_shell.php` of the *Software Design for Mutualistic Software Services (MSS) Version 1.0*.
- Receives services from parent MSS-ENTITY

---

**Figure 24. Service query of the sponsor.**

- Delivers services to child MSS-AFFILIATES
3.2.4 **User Web Site**

- (Version 2.0) Delivers service content to service users
- (Version 2.0) Enables MSS-CENTER/MSS-DEVELOPERS to add service content
- (Version 2.0) Delivers GENI information as required by MSS-CENTER
- (Version 2.0) Enables users to email MSS-CENTER/MSS-DEVELOPERS

3.2.5 **Administrator Web Site**

- (Version 2.0) Delivers administrative content to MSS administrators
Figure 26. Delivers tailored instructions according to user permissions.

- (Version 2.0) Allows MSS-CENTER/MSS-DEVELOPERS to add administrative content
- (Version 2.0) Delivers GENI information as required by MSS-CENTER
- (Version 2.0) Enables users to email MSS-CENTER/MSS-DEVELOPERS

### 3.2.6 Developer Uploads

- (Version 2.0) Allows enrolled MSS-DEVELOPERS to add services
- (Version 2.0) Allows enrolled MSS-DEVELOPERS to add service descriptions
- (Version 2.0) Enables MSS-CENTER to approve services
- (Version 2.0) Enables MSS-CENTER to approve service descriptions

### 3.3 Service Repository

#### 3.3.1 File System

- Holds MSS services
Figure 27. MSS Services subset downloaded to an Apple MacBook.

Figure 28. The services subset matches the Attribute Tree.

Note: Eucalyptus Instance 199.165.126.136-6001 after Choose Services -> sBGP.zip
Figure 29. Eucalyptus Instance downloads from MSS on a canonical ORCA cluster.

- (Version 2.0) Holds all MSS services for MSS-CENTER
- Holds a subset of parent services, to include the entire subset, for its MSS-AFFILIATES
Figure 30. MSS delivers a services subset and rebuilds the Services DB.
4. Evaluation

The MSS system meets all criteria listed for Version 1.0, excepting that the page *Assign Users* does not display in the text-based browser w3m. However, the testing committee believes that this is an unnecessary requirement because only administrators use this page and not Eucalyptus Instance users, for which the w3m requirement exists. *Assign Users* works well in graphical web browsers, and this committee doubts that an administrator would prefer using a text-based browser with a limited feature set to a full-featured graphical web browser, such as Internet Explorer, Firefox, Safari, or Chrome.

The testing committee did find several areas for improvement in MSS, which are listed below in order of importance:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>No separate password() function</td>
<td><em>add_user.php</em> has the administrator type in a new user’s password twice to verify a correct entry, but it performs no strong password checking. One could make the user’s password as simple as “a” or even blank.</td>
<td>Version 1.0 does not include a strong password-checking requirement, but we believe every web-facing server should perform this. This code is widely available, and a web search for “php strong password checker” listed many links.</td>
</tr>
<tr>
<td>Rebuilding the client’s Services database</td>
<td>This functionality is effective, but incredibly slow. For instance, rebuilding the <em>orca-barrow-0</em> database from <em>orca-uaf-2</em> took 2 minutes 40 seconds, even with a very small attribute tree. This is mostly due to a very slow network connection, but the system is designed to open an SSH connection for each entry in all of the Services tables.</td>
<td>Connect only twice from sponsor to client when rebuilding the Services database, instead of connecting for each entry (14 separate connections in testing). Developers can accomplish this by writing the rebuild to a SQL file, passing the file to the client (connection one) and then sending an SSH command (connection two) to load the file into SQL. The SQL code should combine all of the SQL statements into one file. Since each connection to Barrow takes about 5 seconds, this would reduce the total Services database rebuild to about 10 seconds.</td>
</tr>
<tr>
<td>Firefox display malfunction</td>
<td>Firefox does not display quotes in the instructions correctly.</td>
<td>Define the format between the <code>&lt;pre&gt;&lt;/pre&gt;</code> tags in the user instruction set.</td>
</tr>
<tr>
<td>Attribute tree display malfunction</td>
<td>w3m does not display “{1} MSS The root of MSS” correctly. The words “The root of MSS” are aligned all the way to the right of the page. All graphical web browsers present the attribute tree properly.</td>
<td>By viewing the source HTML, one sees that there are too many <code>&lt;table&gt;</code> end tags. Format this section so that the attribute tree does not contain unnecessary tags. The testing committee was unable to repair this malfunction after a short time of troubleshooting.</td>
</tr>
</tbody>
</table>