EAI Endorsed Transactions
on Industrial Networks And Intelligent Systems Research Article **ICST.ORG**

Security Issues in ProtoGENI

Fnu Shalini[1], Yang Xiao[1]*, Bo Sun[2]

[1]Department of Computer Science, The University of Alabama, Tuscaloosa, AL 35487-0290 USA
[2]Department of Computer Science, Lamar University, P.O. Box 10056, Beaumont, TX 77710, USA

Abstract

Network security consists of primary concerns in future Internet development due to the ever increasing threats to current Internet. ProtoGENI is a federated testbed facility supporting slice-based experiments to manage, utilize, and monitor the resources for innovative network research. Security research in ProtoGENI is crucial because experiments conducted in manipulated or corruptted test environment can mislead about security mechanism's capabilities in a system. In this paper, existing ProtoGENI security mechanisms and functions are tested and analyzed through different experiments to find out the exploitable attacking loopholes. Experiments elaborate the existing functioning and security issues that can cause non-functional, semi non-functional or malfunctioned systems. Results indicate threats to ProtoGENI resources and run-time interactions. Cross-experiment communication in Emulab wireless nodes have the capability of assisting in verifying isolation between ProtoGENI slices. Host security is one of the security components, which can be enhanced by modifying default security settings including SSH port number and root login rights. Documentation of experiment environment, experiment design, results, and analysis including many observations is helpful to understand the basic functioning and security issues to improve the overall functioning and security of ProtoGENI.

Keywords: Security, GENI, ProtoGENI

Received on 30 September 2015, accepted on 19 October 2015, published on 01 January 2016

doi: 10.4108/eai.1-1-2016.150807

*Corresponding author. Email:yangxiao@ieee.org

1. Introduction

Information stealing or misuse of available information has gone to an extent to warn the Internet community to seriously think about security and accountability in Internet [1-2]. There is a notable increase in reported cybercrimes every year and the associated financial losses are only one side of these cybercrimes [3].

Internet threats such as IP spoofing [4], malicious mobile code [4], obfuscated code, throttling limiting the rate of delivery, malicious/tracking cookies, browser hijackers, and distributed DoS (denial of service) [6-7] motivated researchers to work on reducing security in terms of deterrence, detection mechanisms, and accountability (detection, traceability, and universal cyber laws) [8] in the Internet. Most of the security problems are based on the exploitation of fundamental flaws in current Internet's architecture, which did not consider the security as its primary concern. This basic problem encourages attackers to maximize the attacking effect by studying vulnerabilities of software applications and Operating System.

Does everyone want a secure Internet in the future? Does the Internet user community really understand the threats? It is more like a burden on research community who can see clearly the threats, the causes, the consequences, and the damage. Researchers are working around the world independently as well as in collaboration to find out the solutions to future Internet. GENI (Global Environment for Network Innovations) [9], FIND (Future Internet Design) [10], FIRE (Future Internet Research and Experimentation) [11], AKARI [12], and NICT (National Institute of Information and Communication Technology) [13] are a few of the frontiers.

Security needs are updated in the most efficient and possible ways with these limitations but unfortunately security cannot serve as an add-on. Therefore, we need to develop a future Internet with built-in capabilities to survive the security challenges [10, 13-17]. Moreover, security is a process and not a product; therefore, any security system should necessarily be capable when considering detection of new possible threats to be pro-active in order to detect any unusual or malicious activity in the network. Detection of any new threat may help to develop strategies to tame the threat before the threat damages too much to system's overall functioning. Detection can be very useful in developing resistance techniques against the threats or to have alternative management systems to keep the network in a solid functioning state despite of being attacked through some new mechanisms, which may not be already included into system's surveillance and resistance framework.

It will take time and efforts to test the feasibility of research ideas. Hardware and software resources may not be up to date to carry out the required testing. There is a necessity for specific arrangements to test research ideas [18, 19]. The research ideas are good to implement, total transformation to new infrastructure will be a difficult challenge. Another challenge will be the drawing of a universal policy to resolve cybercrimes as different countries follow different set of legal and social rules [20]. A balance of accountability and privacy is a constant question to implement any new security policies in Internet [22-29].

As switching to future Internet is not idealistic in the near future, researchers are also working to improve current Internet [30-35] until researchers reach a feasible point to change to new technologies and mechanisms for future Internet.

Security is a fundamental reason for a clean-slate Internet design [14, 16]. Clean-slate approach advocates building future Internet with new architecture, new control, management, and security features. The other direction focuses on getting as much infusion of security and accountability in current Internet as possible.

GENI, an National Science Foundation (NSF) funded project to develop future Internet, is expected to be dedicated to research and to serve as an extraordinary platform for researchers and supposedly to be secure and advanced. However, if it could be broken (there is always a possibility), it would be very destructive for the overall GENI functionality.

ProtoGENI [36, 37] is a prototype of GENI functionality. ProtoGENI is the Control Framework for "Cluster C" of spiral I of the GENI effort [38]. ProtoGENI is based on Emulab [18] software and the slice-based Facility Architecture (SFA).

In this paper, we report our research funding for security issues in ProtoGENI. Some of the related work along this line is summarized as follows. The papers [33, 34] use Emulab to study 4D future Internet. The papers [39-40] use Emulab to study MapReduce for cloud computing. The papers [41, 42] use Emulab to study building global view with log files. The papers [33-45] study security and attacks for ProtoGENI. The paper [46] provides a survey of authentication and access control for ProtoGENI. The paper [47] provides a tutorial for ProtoGENI experiments. However, none of the above related work is comprehensive. In the paper, we provides a better and a more comprehensive survey, tutorial, and experiments for security issues in ProtoGENI.

The rest of the paper is organized as follows. Section 2 gives some background information. Section 3 studies issues and concepts of GENI security. Section 45 provides our experiment results. We analyze our results in Section 5 and provide a future work in Section 6. Finally we conclude the paper in Section 7.

2. Background

2.1. Security in GENI/ ProtoGENI

Even though it is impossible to prove whether a complex system is secure mathematically or experimentally, there are many security questions that need to be answered [1]. Impacts of network load and differentiated degrees of security on system performance, usability, and functionality need to be studied [48-49]. Responses to known attacks should not interference with normal operations under the experiments [48-49]. Well-behaved nodes need to perform well and bad-behaved notes should not disrupt the normal operations [48-51].

The goal of GENI/ProtoGENI security architecture is to prevent, detect, and manage attacks. GENI also believes that it is almost impossible to prevent all possible kinds of intentional or unintentional problems, no matter how careful the validation is adopted. With all possible threats, GENI believes to have the capability to dispose of any malicious experiment or component that requires adequate monitoring and control management.

In the extreme case, GENI is expected to shut off itself, but not periodically. GENI can be attempted for denial service attacks due to its scale and its visibility [48]. GENI doesn’t claim to provide 100 % technical solution, as it has to use many pre-existing components and vulnerable software to be cost-effective in early phases [49-50].

2.1.1 Security implementation in ProtoGENI

Currently ProtoGENI is applying one Public Key Infrastructure (PKI), which is covering all ProtoGENI registries, slice authorities, aggregates, and principals. Each principal has an Emulab account and SSL (Secure Sockets Layer) certificate that can be generated on user’s Emulab account. This SSL certificate is the identity of the researcher, which may be used to identify user’s credentials. This allows researchers to activate a session with RPC (Remote Procedure Call) servers. Credentials are signed by the appropriate authority (slice). X.509 format certificate is being used to implement ProtoGENI public key infrastructure. Certificates are transferred using X.690 ASN.1 DER (Distinguished Encoding Rules) [49]. The aggregate which receives these credentials may verify it using a set of root certificates.

2.1.2 Threats to ProtoGENI Components

ProtoGENI [56] is capable of providing researchers with its facilities until all components work together efficiently and adequately. ProtoGENI control framework is Emulab based, and operates on enhanced functionality of Emulab and its subsystems to provide a close system to GENI environment. There are security issues related to different threads, which are required to weave an experiment on ProtoGENI.

2.1.2.1 The Secure Shell (SSH) and SSL

SSH is a set of programs that employ public/private key technology to authenticate and encrypt sessions between user accounts on distributed hosts on the Internet. ProtoGENI only allows users to interact through SSH keys. SSH was not designed to protect against password crackers, IP & TCP attacks, and covert channels [52]. SSH-agent remembers the passphrase so the user has no need to repeatedly type the password every time while communicating or sending data to the server. This information can be obtained through PC backup programs. If assessed, SSH keys are usable by the root user, but could not be transported to other machines for indefinite misuse. SSH Keys also can be regenerated if any indication of misuse can be detected. SSL certification may be forged if it is MD5 algorithm based. It is found that the improved algorithms, like SHA-1 and SHA-2, are also vulnerable while SHA-3 is under development [52].

2.1.2.2 Emulab Security Issues

Emulab does not protect against spoofing on the control network. The test networks are fully separated between experiments by virtualization, but virtualization also may have security threats [53]. A port scanner [54] is often adopted by administrators to verify security, and is also by attackers to scan system vulnerabilities.

2.1.2.3 Other Security issues in ProtoGENI

Unix/Linux operating systems are believed to be more reliable on security measures in comparison to Windows operating system and less vulnerable to attacks [55, 56, 57].

In reality, there are an increasing number of threats to Unix operating system like Trojan and other attacks which prevent others from running a program or exhausting system resources [58, 59].

Other than issues discussed above, there is a likelihood of insider attack that can be a result of many possible reasons based on human complex nature [60]. Carelessness, in protecting passwords, passphrases, or machine’s unrestricted access to all may also contribute a lot in security breach [61-62].

Many parts of ProtoGENI software use the existing software. It is very time-consuming and cost-expensive to rebuild each and every part of software being used in Existing vulnerabilities in such software may be exploited to threat overall ProtoGENI security [48].

2.2. Secure and Accountable Internet

Security is an important aspect to motivate building future Internet. Both lower layers and higher layers of the protocol stack suffer security attacks [14]. Add-on security mechanisms are not very successful since assuming either the edge or the core of the network will be responsible all security features. The newly designed future Internet should not be a collection security mechanisms but a total new design of security architecture [15]. These efforts need a platform like GENI to experiment proposed methods in a huge scale. Security architecture needs to handle all layers in a system wide approach.

Researcher are going on to build an accountable future Internet with the minimum overhead of switching to new infrastructure, new protocols and web policies. There are serious efforts to acknowledge the present and possible threats to Internet security and reliability in the future, as well as new concepts that are being developed innovatively and with support of previous work done in the same problem area. WASCo distributed computing platform provide its net space to run the experiments by untrusted, semi-trusted or trusted clients as PlantLab [63, 64] provides, however, legal relationship between clients and servers has not been defined well. XenoServer project [65] is similar to PlanetLab. The OSU flow-tool package can help resolving security incidents [66].

The paper [67] studied accountability of packet loss and delay under Byzantine fault detection [67]. The paper [68] proposed network storage accountability. The paper [31] studies fault localization similar to loss and delay accountability paper. Another work is regarded as the basis to build an alternative audit system to keep track for the loss and delay of packets in network traffic [32]. Similar to Accountable Internet Protocol (AIP) [30], self-certifying addressing forms are done and AIP extends self-certification to the whole network. To implement source accountability, installation of filters at border routers is the easiest way.

A novel accountable logging methodology called flow-net was proposed in [8], and its performance was studied in [28, 69]. Accountability in operating system was proposed and studied in [2, 55]. Accountability for cloud computing was proposed and studied in [39, 40, 70, 80]. Accountability for smart grids was proposed and studied in [71, 72, 73, 74]. Temporal accountability was studied in [75]. Some quantifiable accountability schemes were proposed in [76, 77]. Multiple resolution logging with flow-net for accountability was studied in

SSH (Secure Shell) was developed to keep network traffic protected through encryption, however, studies show that in recent years, brute-force attack against SSH, ftp, and telnet servers is the most common form of attack [81]. Linux is the most popular operating system among today’s Unix type systems. Again Unix/Linux system is considered safer in comparison to Windows OS but as per one study, Linux is the ‘most breached’ OS (65% of 154,846 hacked systems) on the net (web 2004) [81]. Though it is said that an updated OS with latest patches are secure, but brute-force attacks against SSH are possible on fully updated Linux OS. Earlier ProtoGENI [82] resources, Emulab [18] machines used FreeBSD as its default OS (operating system) but now Linux is the default OS, and there is always a possibility to breach the security because of OS vulnerabilities. Even FreeBSD OS has been reported to vulnerable to certain kind of vulnerability, but it is most secured OS at present by reports [83].

An experimental study shows that weak and guessable passwords are among other reasons to attack against SSH [83]. Literature suggests that default port 22 for SSH service and other default settings may provide an easier way to attack [84, 85]. Most of the existing attacking kits have built-in scripts to exploit these default settings and attacks are almost ineffective if these default settings are changed to a different non-standard port. Technically, it is not a strong way to enhance security, as it is just a change of port, but practically it is more work for the attacker as port scanners may or may not respond to this non-standard port so it is always hard to guess which non-standard port is listening on host. Disabling logins via SSH for the root account is another method of adding strength to SSH security.

3. Issues and Concepts of GENI Security

3.1. Initial Requirements of GENI

GENI tries to support prototype network deployment and experiments on it.

3.1.1 Functional requirements

GENI is expected to test multiple ideas and concepts by allowing long-running continuous experiments. One approach to slices is virtualization, which allows multiple virtual processors work like real and separate processors. GENI need to support isolation among slices o conduct experiments so that they are not interfering each other [18].

3.1.2 Support for security

Several requirements are needed for GENI experiments: the stable and secure infrastructure and the robust isolation among slices [18].

3.1.3 Support for experimenters: Ease of Use; Observability; Fail-safe

In order to provide security, GENI needs to run experiments in “bounding box” that limits the activities under control and monitoring; otherwise, the experiments should be shut down or changed into a safe mode [48].

3.2. What is critical in GENI Security?

As per GENI Facility Security (Draft) [48], GENI is a tool used enable new research of large scale experiments, but it is subject to being attacks.

A GENI slice is a fully programmable substrate and can be configured to conduct attacks similar the experience of PlanetLab [48]. There are four distinct factors which make GENI more focused on advanced security features [48].

- GENI will embody more critical and sensitive resources than PlanetLab so misuse of GENI will be devastating.
- GENI is deeply programmable, providing programmability across every layer dealing with outdoor sensors and wireless nodes and all different kind of hardware so one policy may not be work for all.
- As supposed and claimed by GENI goals, it would be at scale large enough not to be managed by some manual handling, as currently being done in PlanetLAb. Therefore, it would be better to take care of that issue immediately, as opposed to waiting for this to happen.
- As the most promising and prestigious project by NSF, security is among top issues to protect itself from converting a most capable attacking facility by malicious users.

3.3. What are goals and principles of GENI Security Architecture?

The goal of GENI security architecture is to prevent, detect, and manage attacks; in this case, GENI need to be both safe and usable.

GENI works on virtual network and allows authorized users to interact on GENI, in turn, it is auditable considering which nodes are responsible for problems as the system has their credentials and monitoring allows verification of network traffic activities.

GENI proposes specification-based intrusion detection and anticipates some declaration from each PI about expected behaviour of experiments to monitor the behaviour of experiment and it can detect any abnormalities. Though it suits GENI’s nature but may be not straightforward to define an expected behaviour of research experiments. GENI proposes specification-based intrusion detection for network behaviour but also suggests other common, well-known detection methods for other activities like Tripwire tool to detect unexpected modifications of files [48].

3.4. GENI’s Approach to Threats:

The GENI’s approach to threats [9]:

- Experiments violating isolation or accidently taken over by an attacker will be separated via negotiation between GENI and hosting organization.
- Errors caused by the experiments need to traced and studied to prevent future problems. Misuses need to traced and studied.
- Security enhancements are needed in terms of authentication and access control to prevent theft of an experimenter’s credentials.
- Keys of experiments of corrupting OS need to be revoked or changed.
- Above discussed threats suggests security requirements in GENI as follows: Explicit trust, least privilege, revocation, auditability, scalability, autonomy, usability, and performance are major requirements in GENI security architecture.

3.5. Challenges in GENI Security

There is a possibility of unanticipated DoS against GENI control plane and GENI is supposed to return to a previous safe mode. GENI control plane expects to adopt the most suitable and most advanced mechanism to avoid DoS attacks during its deployment [48]. Operational security and privacy issues are still in infancy stage and need more discussion, plan, and consideration of challenges to finalize them. GENI has some legacy components and so there is certain level of threat associated [48]. Hosts participating in GENI have strong views about conducting experiments on security and are aware about possible risks to shared resources. There are many segments in GENI that may cause security issues like vulnerable software, secured hosts, secured communication etc. Network protocols, algorithms, operating system vulnerabilities may as well create security problems.

Other than technical issues, limitations due to human errors like guessable passwords and keep them in writing, leaving systems unattended and unlocked may feed attackers.

4. ProtoGENI Security Experiments

4.1 Research methodology

There are several issues which are of great importance to evaluate the functionality of ProtoGENI and GENI. Some basic questions arise by default: do we have infrastructure capable to conduct these research experiments? Are there threats to disturb the normal resource management and availability of resources? Can traffic be affected between slices and slivers? What can disturb a running network experiment? What are the limitations of using resources? Is system working as it is supposed to work? Are there any unusual observations during experiments? Are there threats to modify or affect any running experiment by inside/ outside attacker? What would be the action in case of any component is compromised?

Resource management involves several security issues. First, who are authorized to use the resources? Which involves the question of identification and authorization; second, what are the availability or usage conflicts? Third, is there interference between experiments? Virtualization security and management are critical to GENI.

Recording network behaviour can be a major help in controlling and regulating network traffic but privacy issues need to be considered.

On the basis of questions and issues about ProtoGENI, this research work focuses on experiments to observe and analyze ProtoGENI functionality and identifying the security issues in ProtoGENI. This research work tried to get answers for following three main research questions (RQs) and their associated sub-problems:

RQ1: Is there any threat to ProtoGENI resources?

- Outage of resources
- Non-usability of available resources
- Vulnerability of wildcard specifications in RSpecs

RQ2: Is there any problem in run-time interactions between ProtoGENI components?

- Communication between slices/ slivers
- Interactions between component managers
- Traffic load and malicious traffic

RQ3: Is there any threat because of default settings for SSH and operating system vulnerabilities?

- How default setting can threat the security in ProtoGENI?
- What can be done to improve the overall security of ProtoGENI?

Availability and accessibility of ProtoGENI resources are of much significance as any experiment may be executed only if resources are available as per experiment's requirement for the required time and with consistent performance. Any loose end will put a doubt on the output of experiment and any small interference due to the traffic load or malicious traffic may disturb other experiments. This work focuses on capturing different issues that affect the basic functionality of ProtoGENI, and eventually, may affect the whole GENI system. In the subsequent chapter, different experiments are designed and executed to gather information. Experiments try to explore the possible threats or operational glitches. The documentation helps to repeat the experiments and a base for expanding experiments to evaluate more, and to improve the overall ProtoGENI functionality and security.

4.2. ProtoGENI Resources Acquisition, Utilization, and Releasing through Test Scripts

This work can help a novice learner to understand ProtoGENI functionality and associated problems a bit more with practical outputs and results. It can help to initiate, and to follow a straightforward path where the user can see the application of each test script, the redundancy of operations and rights available to use resources. This work also includes the observations about inconsistency in outputs and not getting the intended operations.

Utah Emulab is a primary component manager, along with many of the other component mangers that exist. There are different kinds of machines available which are helpful to conduct specific experiments in order to reveal the working on that particular type of machine and are also useful to verify the compatibility issues among different type of machines. We tried to explore the usability and applicability of test scripts provided by ProtoGENI.

As we are concerned to identifying the loose ends, and to improve descriptions to follow steps with more ease, we tried to execute the available test scripts to see their effect on experiments and what information may be more helpful to be available through test scripts.

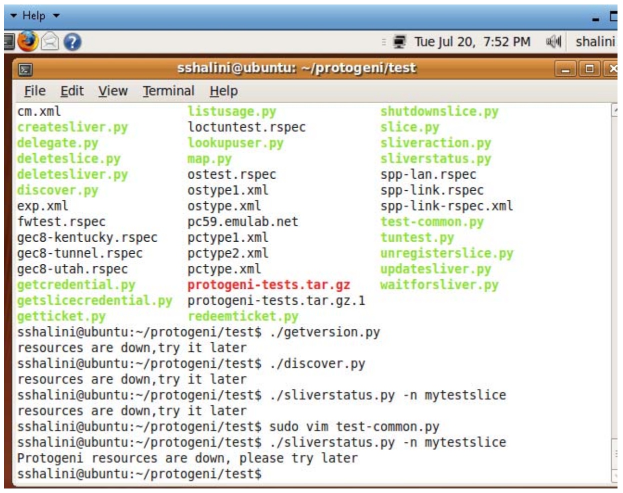
A suite of available command-line tools act as an interim user interface, to provide a means to control ProtoGENI facilities until more sophisticated tools are available. They are also a convenient debugging mechanism to test the emerging ProtoGENI components. All python test scripts help in doing different ProtoGENI operations.

Slice Authority handles some operations that include unprivileged operations, information operations, registration operations, and few special operations. The component manager interacts with user, SA, and clearinghouse through some information operations, ticket operations, manipulation operations, and special operations.

Attack through common test script test-common.py test-common.py script is being used by all other scripts. This test-script can be a prominent attacking target as it can halt the execution of all other test-scripts. test-common.py was modified and a convincible output was generated when user tried to execute any other test script. Following code lines were inserted in the end of the test script file:

- Print "ProtoGENI resources are down, please try later"
- exit()

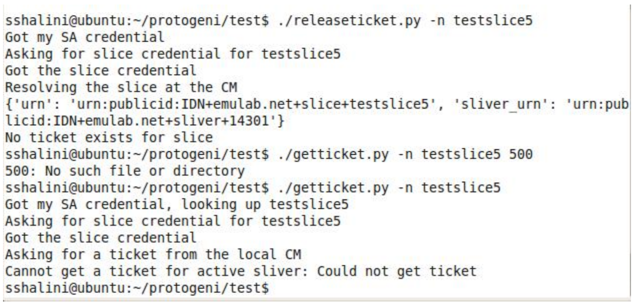
As per Fig. 1, the user may assume that something is in process, so resources are not available and it might restrict any further effort from user to investigate the actual problem.



```
cm.xml                  listusage.py          shutdownslice.py
createsliver.py         loctuntest.rspec      slice.py
delegate.py             lookupuser.py         sliveraction.py
deleteslice.py          map.py                sliverstatus.py
deletesliver.py         ostest.rspec          spp-lan.rspec
discover.py             ostype1.xml           spp-link.rspec
exp.xml                 ostype.xml            spp-link-rspec.xml
fwtest.rspec            pc59.emulab.net       test-common.py
gec8-kentucky.rspec     pctype1.xml           tuntest.py
gec8-tunnel.rspec       pctype2.xml           unregisterslice.py
gec8-utah.rspec         pctype.xml            updatesliver.py
getcredential.py        protogeni-tests.tar.gz  waitforsliver.py
getslicecredential.py   protogeni-tests.tar.gz.1
getticket.py            redeemticket.py
sshalini@ubuntu:~/protogeni/test$ ./getversion.py
resources are down,try it later
sshalini@ubuntu:~/protogeni/test$ ./discover.py
resources are down,try it later
sshalini@ubuntu:~/protogeni/test$ ./sliverstatus.py -n mytestslice
resources are down,try it later
sshalini@ubuntu:~/protogeni/test$ sudo vim test-common.py
sshalini@ubuntu:~/protogeni/test$ ./sliverstatus.py -n mytestslice
Protogeni resources are down, please try later
sshalini@ubuntu:~/protogeni/test$
```

Fig. 1: System is unable to run any script after changes in test-common.py

getticket.py, redeemticket.py, releaseticket.py and getticket.py were executed one after another to see the sequence of processes. After redemption of a ticket the slice got an active sliver and then no ticket operation works on active sliver as shown in Fig. 2.



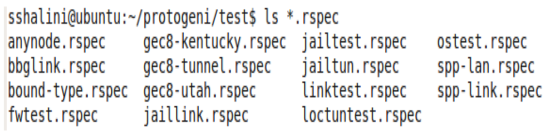
```
sshalini@ubuntu:~/protogeni/test$ ./releaseticket.py -n testslice5
Got my SA credential
Asking for slice credential for testslice5
Got the slice credential
Resolving the slice at the CM
{'urn': 'urn:publicid:IDN+emulab.net+slice+testslice5', 'sliver_urn': 'urn:pub
licid:IDN+emulab.net+sliver+14301'}
No ticket exists for slice
sshalini@ubuntu:~/protogeni/test$ ./getticket.py -n testslice5 500
500: No such file or directory
sshalini@ubuntu:~/protogeni/test$ ./getticket.py -n testslice5
Got my SA credential, looking up testslice5
Asking for slice credential for testslice5
Got the slice credential
Asking for a ticket from the local CM
Cannot get a ticket for active sliver: Could not get ticket
sshalini@ubuntu:~/protogeni/test$
```

Fig. 2: No ticket operation after redemption of ticket as sliver act as active

RSpec .xml file can be borrowed from tutorial of ProtoGENI website [82] and works well to check initial functionality, shown in Fig. 3.

If more resources are required, the same RSpec .xml file can be modified, or a new one can be created once the user is more familiar with RSpec details. OS type of a particular node and a particular kind of PC can be defined in RSpec .xml file.

Different modifications in test scripts were executed to verify the functioning of test scripts available and observed problems that are discussed in great detail of result and analysis section. Experiments helped to distinguish between different ways to acquire resources and preferred the simpler way to acquiring and releasing the ProtoGENI resources to ensure a better accessibility and availability of resources to all ProtoGENI users.



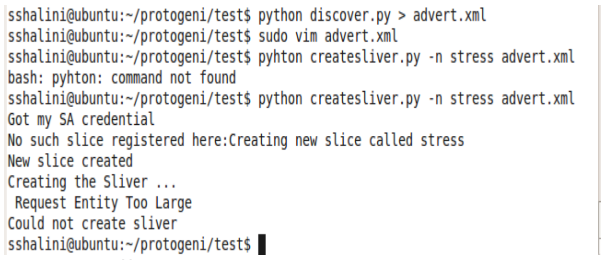
```
sshalini@ubuntu:~/protogeni/test$ ls *.rspec
anynode.rspec     gec8-kentucky.rspec  jailtest.rspec    ostest.rspec
bbglink.rspec     gec8-tunnel.rspec    jailtun.rspec     spp-lan.rspec
bound-type.rspec  gec8-utah.rspec      linktest.rspec    spp-link.rspec
fwtest.rspec      jaillink.rspec       loctuntest.rspec
```

Fig. 3: .rspec files are available in ProtoGENI test scripts suite

4.3. Outage of ProtoGENI Resources

4.3.1 All resources in one sliver, possible?

There was an attempt to acquire all the resources all together by putting advert.xml to create a sliver as it might have a potential threat if all resources may be acquired by one user in a single sliver.



```
sshalini@ubuntu:~/protogeni/test$ python discover.py > advert.xml
sshalini@ubuntu:~/protogeni/test$ sudo vim advert.xml
sshalini@ubuntu:~/protogeni/test$ pyhton createsliver.py -n stress advert.xml
bash: pyhton: command not found
sshalini@ubuntu:~/protogeni/test$ python createsliver.py -n stress advert.xml
Got my SA credential
No such slice registered here:Creating new slice called stress
New slice created
Creating the Sliver ...
 Request Entity Too Large
Could not create sliver
sshalini@ubuntu:~/protogeni/test$ 
```

Fig. 4: Effort to acquire all available resources in one sliver

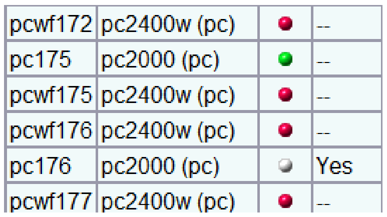
From Fig. 4 we can see that system doesn't allow the creation of a single sliver having all available resources in that one sliver. System is taking care of this issue, and the ProtoGENI authorities can monitor any abnormal demand of resources as well as a sliver may be terminated if required. Most generally used RSpec provides two GENI nodes without any particular node type, ID or OS that results in two available up nodes with default Linux OS image (as on 07-21-2010).

Different RSpecs files were created to see if the system has the capability of providing the requested resources as per specified or not. An experiment can be assembled if the availability of resources is known, available resources can be requested as per requirements, and resources can be allocated or promised to an experimenter [38].

There are several issues about RSpecs with regards to connectivity, topology, and expressiveness of the language to define resource specifications [38, 86]. Rspec can identify every resource explicitly.

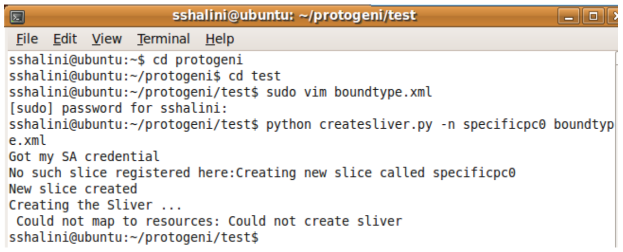
4.3.1.1 Non-Availability of Resources

Subsequently, there is an attempt to reveal how availability of resources can affect different network experiments. A file boundtype.xml was created from bound-type.rspec file in test script package. It initially requested two PC2000 type PCs. As we can see from Fig. 5, there was only one free node of PC2000 type, we could not create the slice. We again modified the RSpec details to request and included PC2400w with PC2000. There were 9 free PC2400w PCs, however, sliver could not be created shown in Fig. 6. We tried to look for characteristics of PC2400w type PC and found its virtnode_capacity was defined as zero [18].



pcwf172	pc2400w (pc)	🔴	--
pc175	pc2000 (pc)	🟢	--
pcwf175	pc2400w (pc)	🔴	--
pcwf176	pc2400w (pc)	🔴	--
pc176	pc2000 (pc)	⚪	Yes
pcwf177	pc2400w (pc)	🔴	--

Fig. 5: Requested PC was busy- Status can be seen at Emulab


sshalini@ubuntu: ~/protogeni/test
File Edit View Terminal Help

```
sshalini@ubuntu:~$ cd protogeni
sshalini@ubuntu:~/protogeni$ cd test
sshalini@ubuntu:~/protogeni/test$ sudo vim boundtype.xml
[sudo] password for sshalini:
sshalini@ubuntu:~/protogeni/test$ python createsliver.py -n specificpc0 boundtyp
e.xml
Got my SA credential
No such slice registered here:Creating new slice called specificpc0
New slice created
Creating the Sliver ...
 Could not map to resources: Could not create sliver
sshalini@ubuntu:~/protogeni/test$
```

Fig. 6: Sliver creation attempt with PC type PC2400w

The RSpec file was then again modified to experiment with other types of nodes. 10 PC600 type nodes were free and the node type virtnode_capacity was defined as 10, so that we requested for two PC600 type PCs. Slice was created successfully.

The experiments in the next section are related to slice/ sliver creation and deletion to see how it affects availability of resources to other experiments. Experiments show that one user can pose threat in execution of other experiments by holding resources in one or another way.

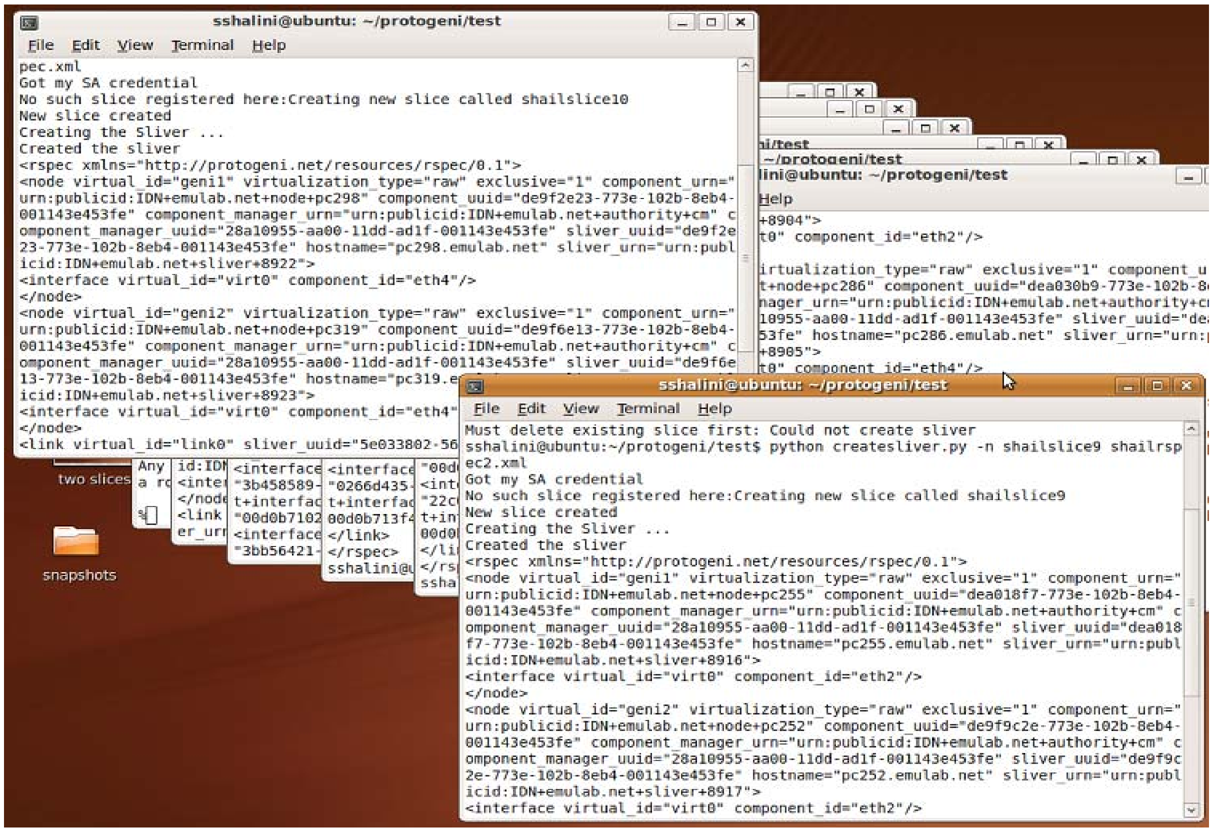
There is a deep concern regarding the threats to resources for all ProtoGENI users. Sometimes a novice experimenter can create many slices to experience different basic experiments and may lose the track of slices created and deleted in process and may hold network resources. This unnoticed and unidentified holding of resources may create shortage of resources or a bottleneck for another topology. Though slices and slivers are short lived for few hours if not being renewed but still some basic experiments, observing the effect of extending sliver time of existence through redeemticket.py, may hold resources for a longer period that can block these resources to be obtained by other ProtoGENI users. These experiments will help to understand the possible threats to availability of resources to other experimenters.

4.3.1.2 Requesting few resources in too many slices/slivers

We observed Emulab account to keep tracking of available, consumed, and freed resources with creation and deletion of slices. Before starting of the experiment, 33 PCs were free on Emulab account, thus, there was a decision to go up to16 Slices at first to see the resources consumption.

The sample RSpec in tutorial [82] was used for all slices, which request only two PCs without stating any specific type. There was an attempt to create one after another in separate terminals with a pattern of slice names like shailslice1, shailslice2 and so forth. Initial 6 slices were created without any problem, but 7th and 8th slices were stuck in creating slices and therefore, aborted both of them, continuing to create the next slices with same name pattern. Again, shailslice11 was not successfully created.

Shailslice13 and shailslice14 could not be created when tried in same terminal with previous slice, but were successfully created when attempted in separate terminals. By the time of creation of shailslice15, there was only one free PC on Emulab records, and system did not create shailslice15 as saying, "could not map to resources, "could not create the slice". Emulab resources were distributed among previous slices and left with one 1 free PC which could not help with creating a sliver with request of two PCs. Fig. 7 shows creation of a series of slices. Figs. 8, 9, and 10 show deletion of slices and resources being freed.

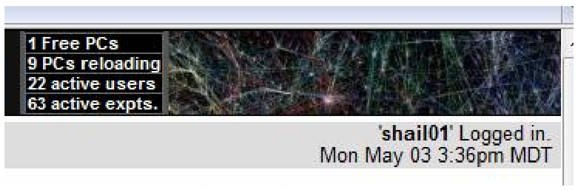


```
pec.xml
Got my SA credential
No such slice registered here:Creating new slice called shailslice10
New slice created
Creating the Sliver ...
Created the sliver
<rspec xmlns="http://protogeni.net/resources/rspec/0.1">
<node virtual_id="geni1" virtualization_type="raw" exclusive="1" component_urn="
urn:publicid:IDN+emulab.net+node+pc298" component_uuid="de9f2e23-773e-102b-8eb4-
001143e453fe" component_manager_urn="urn:publicid:IDN+emulab.net+authority+cm" c
omponent_manager_uuid="28a10955-aa00-11dd-ad1f-001143e453fe" sliver_uuid="de9f2e
23-773e-102b-8eb4-001143e453fe" hostname="pc298.emulab.net" sliver_urn="urn:publ
icid:IDN+emulab.net+sliver+8922">
<interface virtual_id="virt0" component_id="eth4"/>
</node>
<node virtual_id="geni2" virtualization_type="raw" exclusive="1" component_urn="
urn:publicid:IDN+emulab.net+node+pc319" component_uuid="de9f6e13-773e-102b-8eb4-
001143e453fe" component_manager_urn="urn:publicid:IDN+emulab.net+authority+cm" c
omponent_manager_uuid="28a10955-aa00-11dd-ad1f-001143e453fe" sliver_uuid="de9f6e
13-773e-102b-8eb4-001143e453fe" hostname="pc319.e
icid:IDN+emulab.net+sliver+8923">
<interface virtual_id="virt0" component_id="eth4"
</node>
<link virtual_id="link0" sliver_uuid="5e033802-56
```

```
+8904">
t0" component_id="eth2"/>

irtualization_type="raw" exclusive="1" component_u
t+node+pc286" component_uuid="dea030b9-773e-102b-8
nager_urn="urn:publicid:IDN+emulab.net+authority+c
10955-aa00-11dd-ad1f-001143e453fe" sliver_uuid="de
53fe" hostname="pc286.emulab.net" sliver_urn="urn:
+8905">
t0" component_id="eth4"/>
```

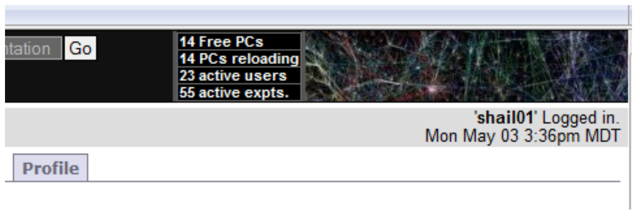
```
Must delete existing slice first: Could not create sliver
sshalini@ubuntu:~/protogeni/test$ python createsliver.py -n shailslice9 shailrsp
ec2.xml
Got my SA credential
No such slice registered here:Creating new slice called shailslice9
New slice created
Creating the Sliver ...
Created the sliver
<rspec xmlns="http://protogeni.net/resources/rspec/0.1">
<node virtual_id="geni1" virtualization_type="raw" exclusive="1" component_urn="
urn:publicid:IDN+emulab.net+node+pc255" component_uuid="dea018f7-773e-102b-8eb4-
001143e453fe" component_manager_urn="urn:publicid:IDN+emulab.net+authority+cm" c
omponent_manager_uuid="28a10955-aa00-11dd-ad1f-001143e453fe" sliver_uuid="dea018
f7-773e-102b-8eb4-001143e453fe" hostname="pc255.emulab.net" sliver_urn="urn:publ
icid:IDN+emulab.net+sliver+8916">
<interface virtual_id="virt0" component_id="eth2"/>
</node>
<node virtual_id="geni2" virtualization_type="raw" exclusive="1" component_urn="
urn:publicid:IDN+emulab.net+node+pc252" component_uuid="de9f9c2e-773e-102b-8eb4-
001143e453fe" component_manager_urn="urn:publicid:IDN+emulab.net+authority+cm" c
omponent_manager_uuid="28a10955-aa00-11dd-ad1f-001143e453fe" sliver_uuid="de9f9c
2e-773e-102b-8eb4-001143e453fe" hostname="pc252.emulab.net" sliver_urn="urn:publ
icid:IDN+emulab.net+sliver+8917">
<interface virtual_id="virt0" component_id="eth2"/>
```

Fig.7: Creation of a series of slices from shailslice1 to shailslice16



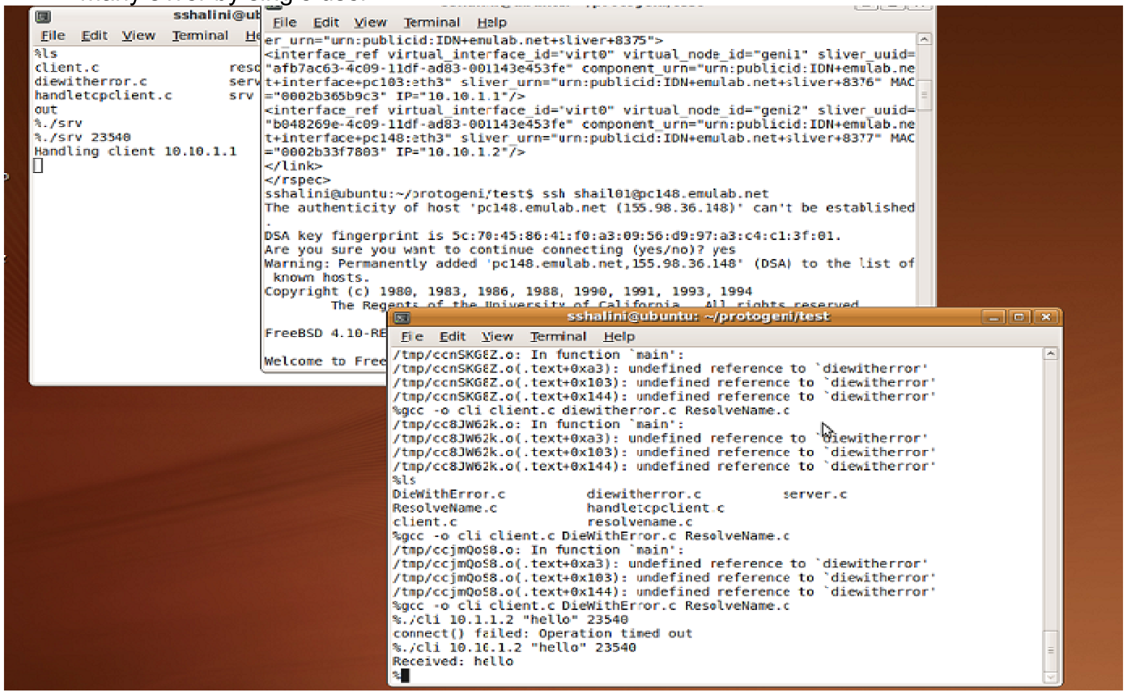
```
1 Free PCs
9 PCs reloading
22 active users
63 active expts.
'shail01' Logged in.
Mon May 03 3:36pm MDT
```

Fig. 8: Outage of ProtoGENI Resources through many sliver by single user



```
14 Free PCs
14 PCs reloading
23 active users
55 active expts.
'shail01' Logged in.
Mon May 03 3:36pm MDT
```

Fig. 9: More resources available with deletion of more slices.



```
%ls
client.c                   reso
diewitherror.c             serv
handletcpclient.c          srv
out
%./srv
%./srv 23540
Handling client 10.10.1.1
```

```
er_urn="urn:publicid:IDN+emulab.net+sliver+8375">
<interface_ref virtual_interface_id="virt0" virtual_node_id="geni1" sliver_uuid=
"afb7ac63-4c09-11df-ad83-001143e453fe" component_urn="urn:publicid:IDN+emulab.ne
t+interface+pc103:eth3" sliver_urn="urn:publicid:IDN+emulab.net+sliver+8376" MAC
="0002b365b9c3" IP="10.10.1.1"/>
<interface_ref virtual_interface_id="virt0" virtual_node_id="geni2" sliver_uuid=
"b048269e-4c09-11df-ad83-001143e453fe" component_urn="urn:publicid:IDN+emulab.ne
t+interface+pc148:eth3" sliver_urn="urn:publicid:IDN+emulab.net+sliver+8377" MAC
="0002b33f7803" IP="10.10.1.2"/>
</link>
</rspec>
sshalini@ubuntu:~/protogeni/test$ ssh shail01@pc148.emulab.net
The authenticity of host 'pc148.emulab.net (155.98.36.148)' can't be established
.
DSA key fingerprint is 5c:70:45:86:41:f0:a3:09:56:d9:97:a3:c4:c1:3f:01.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'pc148.emulab.net,155.98.36.148' (DSA) to the list of
 known hosts.
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994

FreeBSD 4.10-RE

Welcome to Free
```

```
/tmp/ccnSKGEZ.o: In function `main':
/tmp/ccnSKGEZ.o(.text+0xa3): undefined reference to `diewitherror'
/tmp/ccnSKGEZ.o(.text+0x103): undefined reference to `diewitherror'
/tmp/ccnSKGEZ.o(.text+0x144): undefined reference to `diewitherror'
%gcc -o cli client.c diewitherror.c ResolveName.c
/tmp/cc8JW62k.o: In function `main':
/tmp/cc8JW62k.o(.text+0xa3): undefined reference to `diewitherror'
/tmp/cc8JW62k.o(.text+0x103): undefined reference to `diewitherror'
/tmp/cc8JW62k.o(.text+0x144): undefined reference to `diewitherror'
%ls
DieWithError.c          diewitherror.c          server.c
ResolveName.c           handletcpclient.c
client.c                resolvename.c
%gcc -o cli client.c DieWithError.c ResolveName.c
/tmp/ccjmQoS8.o: In function `main':
/tmp/ccjmQoS8.o(.text+0xa3): undefined reference to `diewitherror'
/tmp/ccjmQoS8.o(.text+0x103): undefined reference to `diewitherror'
/tmp/ccjmQoS8.o(.text+0x144): undefined reference to `diewitherror'
%gcc -o cli client.c DieWithError.c ResolveName.c
%./cli 10.1.1.2 "hello" 23540
connect() failed: Operation timed out
%./cli 10.10.1.2 "hello" 23540
Received: hello
%
```

sshalini@ub
sshalini@ubuntu: ~/protogeni/test
File Edit View Terminal Help

Fig. 10: Communication between Client and Server Machines

4.3.1.3 Stress test

For stress test, there was an attempt to request many resources at a time which again may reduce the possible number of experiments on Emulab resources. Modify the basic RSpec file to add more resources in one resource .xml file. Thus, create stressrspec.xml (an expansion of basic RSpec, with more resources in same format with a requirement of 6 PCs and 3 links at a time). Available resources were 33 PCs. Creation of two slices reduced the Emulab resources down to 21 free PCs. More resources were added in one request. A slice was created with all 14 PCs and 7 links and Emulab resources reduced to 7 free PCs. Next attempt was to create a similar slice with 14 PCs and 7 links, it could not be created and stated: “could not map to resources, could not create sliver”.

One may notice that more resources requested at a time in a single slice reduce the chances of creation of other slivers. This poses a threat to availability of resources to experimenters. We also observed the reverse activity of resources being freed with deletion of stress slices.

4.4. Run-Time Interactions between ProtoGENI Components

Run-time network interactions can be a cause of network problems as different experiments interact with different ProtoGENI components. Communication among the ProtoGENI nodes and also between ProtoGENI and outside network is important to explore further experiments. This section focuses on experiments to test whether a ProtoGENI sliver can receive from (or send to) another slice, or outside network.

As known, a user can own resources in a sliver therefore; he or she can have control on those nodes, and may conduct the experiments. Now the interests are to see the communication capabilities between different slivers and between ProtoGENI sliver and its constituent outside network.

This may provide an idea about possible isolation or cooperation between ProtoGENI slivers, as well as between ProtoGENI sliver and outside network. This work may help to identifying the threats which could impact the accessibility of GENI.

The experiments to discover the answers for following have been conducted as follows:

- Whether a sliver can receive from (or send to) another slice?
- Whether a sliver can receive from (or sent to) outside network?
- Is communication possible between nodes from two component managers?
- How communication between wireless nodes in different slivers can pose threats to network traffic?

There was a use of the simple approach to create different slivers and test the communication through simple client-server programs. Subsequently, using the Linux virtual machine (Ubuntu 9.0) as outside network node and verified the communication via client-server programs.

4.4.1 Communication between two ProtoGENI nodes as Client and Server:

For the Client-Server program on remote machines, the server must be initiated first.

At the machine treated as Server:

gcc -o srv server.c handletcpclient.c DieWithError.c

At machine treated as Client:

gcc -o cli client.c DieWithError.c ResolveName.c

After compilation, systems generate executable files to run client and server programs.

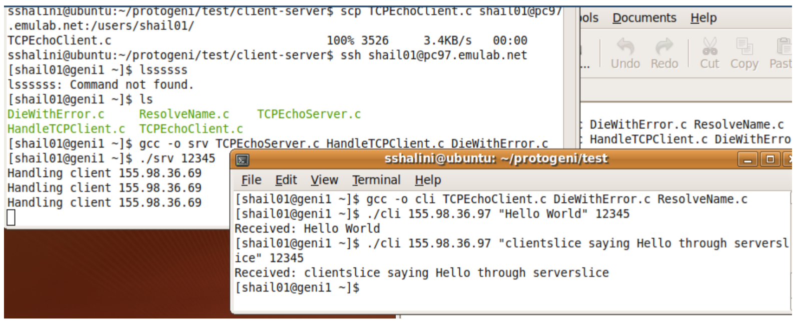
4.4.2. Communication between nodes of different slivers

Two slices were created with no particular rspec file and system allowed one node to each sliver.

PC69 were allotted to clientslice and PC97 was allotted to server slice. There was an upload of simple basic client server C program files on both nodes as we planned to observe both nodes as operating client as well as server. Files were uploaded one by one by following command with the required file name to upload:

sudo scp HandleTCPClient.c shail01@pc69.emulab.net:/users/shail01/

After this, roles were reversed for both slivers to check two-way operation between ProtoGENI slivers. PC97 was reset to act as client and PC69 was reset to act as client. Communication worked fine with reversed roles of sliver nodes, shown in Fig. 11. These operations show that different slivers can send or receive from each other. Fig. 12 shows verification of both slivers as client and server.



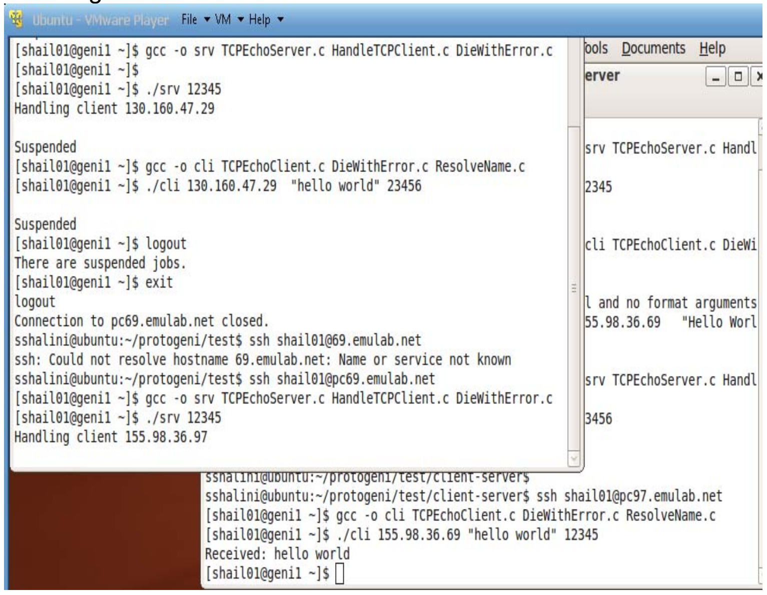
```
sshalini@ubuntu:~/protogeni/test/client-server$ scp TCPEchoClient.c shail01@pc97
.emulab.net:/users/shail01/
TCPEchoClient.c                              100% 3526     3.4KB/s   00:00
sshalini@ubuntu:~/protogeni/test/client-server$ ssh shail01@pc97.emulab.net
[shail01@geni1 ~]$ lssssss
lssssss: Command not found.
[shail01@geni1 ~]$ ls
DieWithError.c     ResolveName.c    TCPEchoServer.c
HandleTCPClient.c  TCPEchoClient.c
[shail01@geni1 ~]$ gcc -o srv TCPEchoServer.c HandleTCPClient.c DieWithError.c
[shail01@geni1 ~]$ ./srv 12345
Handling client 155.98.36.69
Handling client 155.98.36.69
Handling client 155.98.36.69
```

```
DieWithError.c ResolveName.c
HandleTCPClient.c DieWithErro
```

sshalini@ubuntu: ~/protogeni/test

```
[shail01@geni1 ~]$ gcc -o cli TCPEchoClient.c DieWithError.c ResolveName.c
[shail01@geni1 ~]$ ./cli 155.98.36.97 "Hello World" 12345
Received: Hello World
[shail01@geni1 ~]$ ./cli 155.98.36.97 "clientslice saying Hello through serversl
ice" 12345
Received: clientslice saying Hello through serverslice
[shail01@geni1 ~]$
```

Fig. 11: Communication between two different sliver PCs



```
[shail01@geni1 ~]$ gcc -o srv TCPEchoServer.c HandleTCPClient.c DieWithError.c
[shail01@geni1 ~]$
[shail01@geni1 ~]$ ./srv 12345
Handling client 130.160.47.29

Suspended
[shail01@geni1 ~]$ gcc -o cli TCPEchoClient.c DieWithError.c ResolveName.c
[shail01@geni1 ~]$ ./cli 130.160.47.29  "hello world" 23456

Suspended
[shail01@geni1 ~]$ logout
There are suspended jobs.
[shail01@geni1 ~]$ exit
logout
Connection to pc69.emulab.net closed.
sshalini@ubuntu:~/protogeni/test$ ssh shail01@69.emulab.net
ssh: Could not resolve hostname 69.emulab.net: Name or service not known
sshalini@ubuntu:~/protogeni/test$ ssh shail01@pc69.emulab.net
[shail01@geni1 ~]$ gcc -o srv TCPEchoServer.c HandleTCPClient.c DieWithError.c
[shail01@geni1 ~]$ ./srv 12345
Handling client 155.98.36.97
```

```
srv TCPEchoServer.c Handl

2345

cli TCPEchoClient.c DieWi

l and no format arguments
55.98.36.69   "Hello Worl

srv TCPEchoServer.c Handl

3456
```

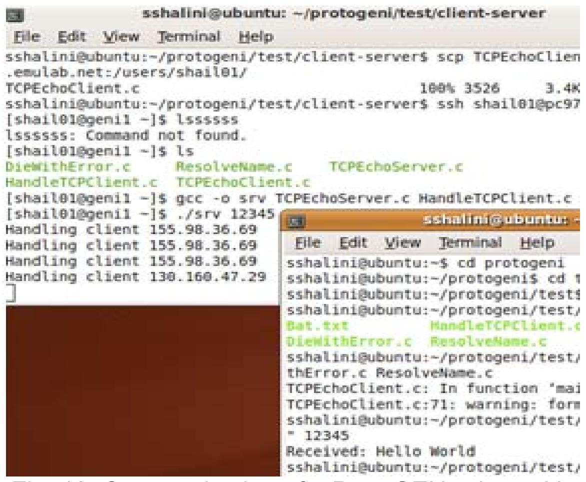
```
sshalini@ubuntu:~/protogeni/test/client-server$
sshalini@ubuntu:~/protogeni/test/client-server$ ssh shail01@pc97.emulab.net
[shail01@geni1 ~]$ gcc -o cli TCPEchoClient.c DieWithError.c ResolveName.c
[shail01@geni1 ~]$ ./cli 155.98.36.69 "hello world" 12345
Received: hello world
[shail01@geni1 ~]$
```

Fig. 12: verification of both slivers as client and server

4.4.3 Communication between a ProtoGENI sliver and outside network node

We continued with same serverslice sliver with single node PC97 and opted for our own PC as outside network node to communicate with ProtoGENI sliver.

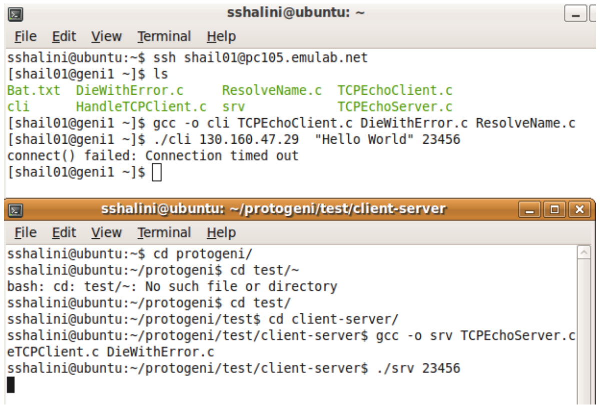
ProtoGENI sliver is operating as a server as well as a lab machine that is used as outside network node. Communication is worthy with this setting, but connection could not be established when the serverslice node was set as a client. To diagnose furthermore, there were created slivers again with shrspec1.xml and PC105 was allotted to experiment. Operational settings for both ProtoGENI node and outside network node were repeated to verify client and server operations, however, it yielded the same results. ProtoGENI node is working fine as server with an outside network client nod, but could not establish the connection when ProtoGENI node was reset as client and outside network node as server. Figs. 13 and 14 show these results.


sshalini@ubuntu: ~/protogeni/test/client-server
File Edit View Terminal Help

```
sshalini@ubuntu:~/protogeni/test/client-server$ scp TCPEchoClien
.emulab.net:/users/shail01/
TCPEchoClient.c                                    100% 3526     3.4K
sshalini@ubuntu:~/protogeni/test/client-server$ ssh shail01@pc97
[shail01@geni1 ~]$ lssssss
lssssss: Command not found.
[shail01@geni1 ~]$ ls
DieWithError.c     ResolveName.c    TCPEchoServer.c
HandleTCPClient.c  TCPEchoClient.c
[shail01@geni1 ~]$ gcc -o srv TCPEchoServer.c HandleTCPClient.c
[shail01@geni1 ~]$ ./srv 12345
Handling client 155.98.36.69
Handling client 155.98.36.69
Handling client 155.98.36.69
Handling client 130.160.47.29
```

```
sshalini@ubuntu:~$ cd protogeni
sshalini@ubuntu:~/protogeni$ cd t
sshalini@ubuntu:~/protogeni/test$
sshalini@ubuntu:~/protogeni/test/
Bat.txt           HandleTCPClient.c
DieWithError.c    ResolveName.c
sshalini@ubuntu:~/protogeni/test/
thError.c ResolveName.c
TCPEchoClient.c: In function 'mai
TCPEchoClient.c:71: warning: form
sshalini@ubuntu:~/protogeni/test/
" 12345
Received: Hello World
sshalini@ubuntu:~/protogeni/test/
```

Fig. 13: Communication of a ProtoGENI sliver with outside network


sshalini@ubuntu: ~
File Edit View Terminal Help

```
sshalini@ubuntu:~$ ssh shail01@pc105.emulab.net
[shail01@geni1 ~]$ ls
Bat.txt  DieWithError.c     ResolveName.c  TCPEchoClient.c
cli      HandleTCPClient.c  srv            TCPEchoServer.c
[shail01@geni1 ~]$ gcc -o cli TCPEchoClient.c DieWithError.c ResolveName.c
[shail01@geni1 ~]$ ./cli 130.160.47.29  "Hello World" 23456
connect() failed: Connection timed out
[shail01@geni1 ~]$ 
```

sshalini@ubuntu: ~/protogeni/test/client-server
File Edit View Terminal Help

```
sshalini@ubuntu:~$ cd protogeni/
sshalini@ubuntu:~/protogeni$ cd test/~
bash: cd: test/~: No such file or directory
sshalini@ubuntu:~/protogeni$ cd test/
sshalini@ubuntu:~/protogeni/test$ cd client-server/
sshalini@ubuntu:~/protogeni/test/client-server$ gcc -o srv TCPEchoServer.c
eTCPClient.c DieWithError.c
sshalini@ubuntu:~/protogeni/test/client-server$ ./srv 23456
```

Fig. 14: Error in communication when ProtoGENI node acted as client

ProtoGENI sliver is working partially with outside network node. More work is required to find out the reason of why it is not working as a client with an outside server node. Although there is some clarification required for non successful communication between a ProtoGENI node as client and outside network node as server, there is anticipation that more experiments would be helpful to explore the reasons.

4.4.4 Communication between ProtoGENI nodes from two Component Managers

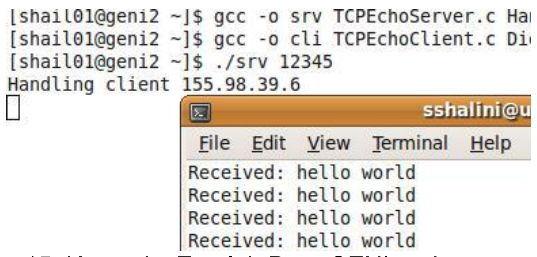
Sliver twocm was created with tuntest.py test script requesting two nodes from two different Component Managers (CM) Utah Emulab and Kentucky Emulab.

Node geni1: Utah Emulab pc206.emulab.net;

Node geni2: Kentucky Emulab, pc27.uky.emulab.net

As before, we uploaded client server program files on remote nodes, and compiled the programs.

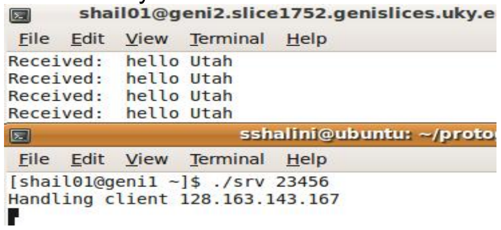
Communication was tried both ways: both nodes acting as server as well as client Results show in Figs. 15 and 16.



```
[shail01@geni2 ~]$ gcc -o srv TCPEchoServer.c Ha
[shail01@geni2 ~]$ gcc -o cli TCPEchoClient.c Di
[shail01@geni2 ~]$ ./srv 12345
Handling client 155.98.39.6
```

```
Received: hello world
Received: hello world
Received: hello world
Received: hello world
```

Fig. 15: Kentucky Emulab ProtoGENI node as server


shail01@geni2.slice1752.genislices.uky.e
File Edit View Terminal Help

```
Received:  hello Utah
Received:  hello Utah
Received:  hello Utah
Received:  hello Utah
```

sshalini@ubuntu: ~/proto
File Edit View Terminal Help

```
[shail01@geni1 ~]$ ./srv 23456
Handling client 128.163.143.167
```

Fig. 16: Utah Emulab ProtoGENI node as server

4.4.5 Communication between Emulab wireless nodes

These experiments are to explore wireless traffic issues. Though these are not through slivers and initiated as Emulab experiments as more work is proposed on wireless nodes through ProtoGENI, this is initial work to interact with wireless nodes and to observe possible communication issues that will affect ProtoGENI experiments as well.

Two experiments were created to see communication between wireless nodes. One experiment had 4 wireless nodes pcwf1, pcwf3, pcwf5, pcwf7 and second experiment was created with pcwf2 and pcwf4, shown in Fig. 17. Configuration of nodes can be verified to communicate via wireless channel through ifconfig command at remote nodes.

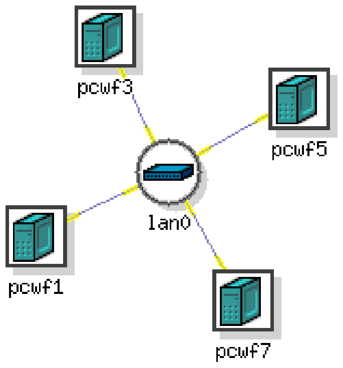
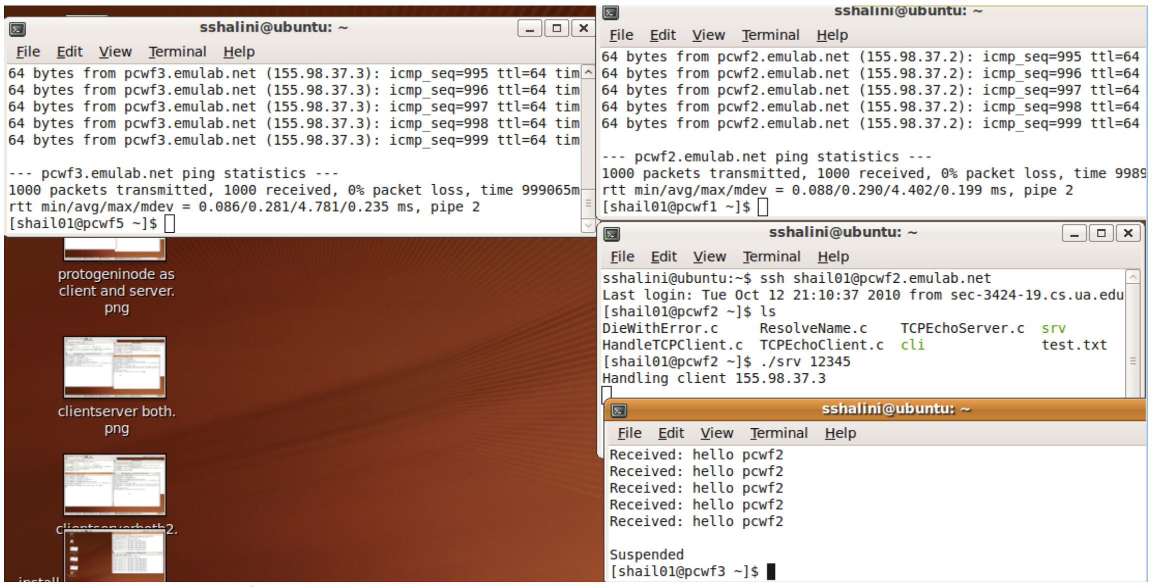


Fig. 17: Emulab topology for wireless nodes

We tried to see the effect of pinging one node to other separately, pinging to every node together and effect of sending data in indefinite loop to see how these all affect traffic outputs and other communication process. Details show that max rtt (round trip time) is inclined with the increasing number of packets and especially when they were pinging all together.

We observed that system applied different pipes to cope with the communication load but it increases the rtt significantly and loss of packets.


sshalini@ubuntu: ~
File Edit View Terminal Help
protogeninode as client and server. png
clientserver both. png
clientserverboth2.
install

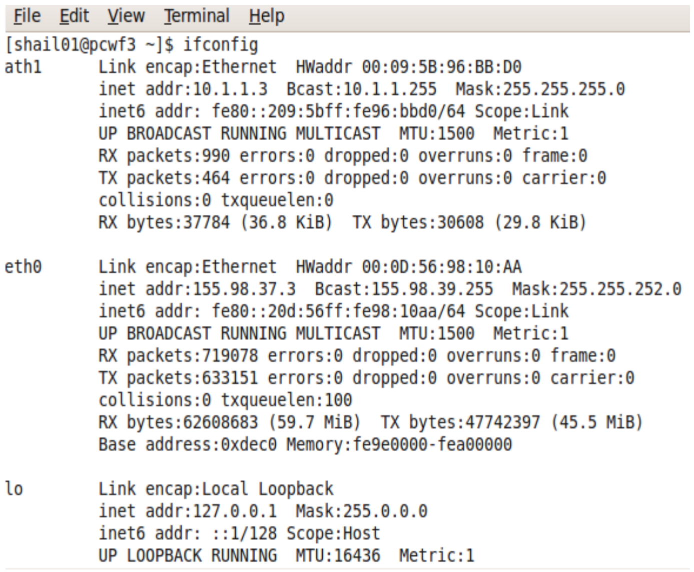
```
64 bytes from pcwf3.emulab.net (155.98.37.3): icmp_seq=995 ttl=64 tim
64 bytes from pcwf3.emulab.net (155.98.37.3): icmp_seq=996 ttl=64 tim
64 bytes from pcwf3.emulab.net (155.98.37.3): icmp_seq=997 ttl=64 tim
64 bytes from pcwf3.emulab.net (155.98.37.3): icmp_seq=998 ttl=64 tim
64 bytes from pcwf3.emulab.net (155.98.37.3): icmp_seq=999 ttl=64 tim

--- pcwf3.emulab.net ping statistics ---
1000 packets transmitted, 1000 received, 0% packet loss, time 999065m
rtt min/avg/max/mdev = 0.086/0.281/4.781/0.235 ms, pipe 2
[shail01@pcwf5 ~]$
```

```
64 bytes from pcwf2.emulab.net (155.98.37.2): icmp_seq=995 ttl=64
64 bytes from pcwf2.emulab.net (155.98.37.2): icmp_seq=996 ttl=64
64 bytes from pcwf2.emulab.net (155.98.37.2): icmp_seq=997 ttl=64
64 bytes from pcwf2.emulab.net (155.98.37.2): icmp_seq=998 ttl=64
64 bytes from pcwf2.emulab.net (155.98.37.2): icmp_seq=999 ttl=64

--- pcwf2.emulab.net ping statistics ---
1000 packets transmitted, 1000 received, 0% packet loss, time 9989
rtt min/avg/max/mdev = 0.088/0.290/4.402/0.199 ms, pipe 2
[shail01@pcwf1 ~]$
```

```
sshalini@ubuntu:~$ ssh shail01@pcwf2.emulab.net
Last login: Tue Oct 12 21:10:37 2010 from sec-3424-19.cs.ua.edu
[shail01@pcwf2 ~]$ ls
DieWithError.c     ResolveName.c    TCPEchoServer.c  srv
HandleTCPClient.c  TCPEchoClient.c  cli              test.txt
[shail01@pcwf2 ~]$ ./srv 12345
Handling client 155.98.37.3
```

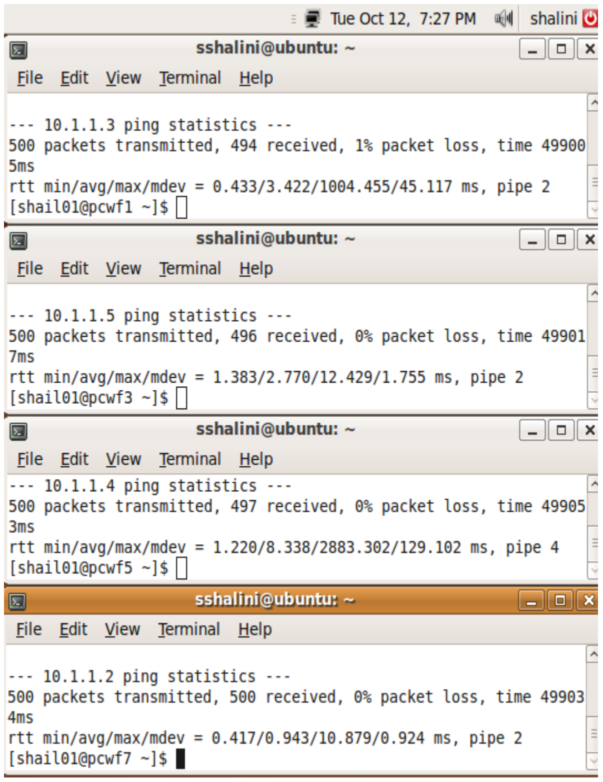
```
Received: hello pcwf2
Received: hello pcwf2
Received: hello pcwf2
Received: hello pcwf2
Received: hello pcwf2

Suspended
[shail01@pcwf3 ~]$
```

Fig. 18: Cross communication between two experiments with wireless nodes



```
[shail01@pcwf3 ~]$ ifconfig
ath1      Link encap:Ethernet  HWaddr 00:09:5B:96:BB:D0
          inet addr:10.1.1.3  Bcast:10.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::209:5bff:fe96:bbd0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:990 errors:0 dropped:0 overruns:0 frame:0
          TX packets:464 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:37784 (36.8 KiB)  TX bytes:30608 (29.8 KiB)

eth0      Link encap:Ethernet  HWaddr 00:0D:56:98:10:AA
          inet addr:155.98.37.3  Bcast:155.98.39.255  Mask:255.255.252.0
          inet6 addr: fe80::20d:56ff:fe98:10aa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:719078 errors:0 dropped:0 overruns:0 frame:0
          TX packets:633151 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:62608683 (59.7 MiB)  TX bytes:47742397 (45.5 MiB)
          Base address:0xdec0 Memory:fe9e0000-fea00000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

Fig. 19: Configuration details at remote node in wireless experiment



```
--- 10.1.1.3 ping statistics ---
500 packets transmitted, 494 received, 1% packet loss, time 49900
5ms
rtt min/avg/max/mdev = 0.433/3.422/1004.455/45.117 ms, pipe 2
[shail01@pcwf1 ~]$
```

```
--- 10.1.1.5 ping statistics ---
500 packets transmitted, 496 received, 0% packet loss, time 49901
7ms
rtt min/avg/max/mdev = 1.383/2.770/12.429/1.755 ms, pipe 2
[shail01@pcwf3 ~]$
```

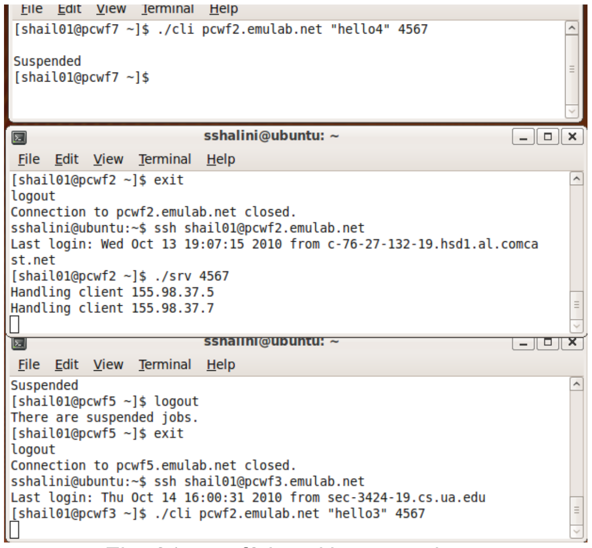
```
--- 10.1.1.4 ping statistics ---
500 packets transmitted, 497 received, 0% packet loss, time 49905
3ms
rtt min/avg/max/mdev = 1.220/8.338/2883.302/129.102 ms, pipe 4
[shail01@pcwf5 ~]$
```

```
--- 10.1.1.2 ping statistics ---
500 packets transmitted, 500 received, 0% packet loss, time 49903
4ms
rtt min/avg/max/mdev = 0.417/0.943/10.879/0.924 ms, pipe 2
[shail01@pcwf7 ~]$
```

Fig. 20: Pinging pcwf1-3-5-7-1, 500 pkts. (All at same time)

Client-server programs were tested while pinging and also tried when two nodes were communicating in indefinite loop and any other node tried to communicate with one of the busy nodes. Cross communication between two experiments is shown as per Fig. 18.

Next, we tried to communicate with a busy node in an indefinite loop of client server communication, shown in Figs. 19, 20, 21.



```
[shail01@pcwf7 ~]$ ./cli pcwf2.emulab.net "hello4" 4567

Suspended
[shail01@pcwf7 ~]$
```

```
[shail01@pcwf2 ~]$ exit
logout
Connection to pcwf2.emulab.net closed.
sshalini@ubuntu:~$ ssh shail01@pcwf2.emulab.net
Last login: Wed Oct 13 19:07:15 2010 from c-76-27-132-19.hsd1.al.comca
st.net
[shail01@pcwf2 ~]$ ./srv 4567
Handling client 155.98.37.5
Handling client 155.98.37.7
```

```
Suspended
[shail01@pcwf5 ~]$ logout
There are suspended jobs.
[shail01@pcwf5 ~]$ exit
logout
Connection to pcwf5.emulab.net closed.
sshalini@ubuntu:~$ ssh shail01@pcwf3.emulab.net
Last login: Thu Oct 14 16:00:31 2010 from sec-3424-19.cs.ua.edu
[shail01@pcwf3 ~]$ ./cli pcwf2.emulab.net "hello3" 4567
```

Fig. 21: pcwf2 handling two clients

Experiments showed some different outputs like any third node that was not able to communicate with the busy node, but in certain cases when we close the client node, server showed handling of another client, which had already been suspended. It was inconsistent as at other times it just denied the connection stating "connection was reset by peer". More experiments will be helpful to diagnose theses issues.

Experiments show that different slivers can communicate and also can communicate to outside network.

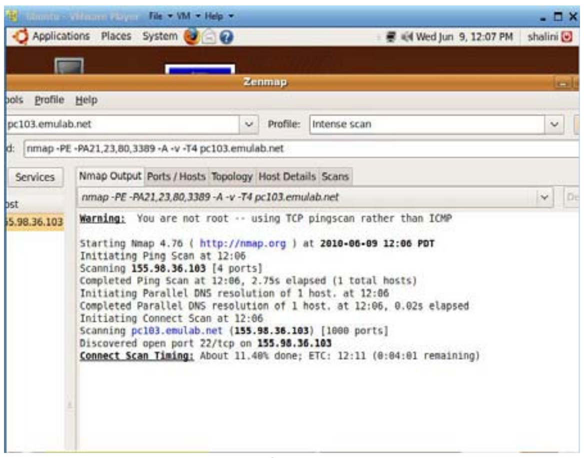
Wireless node communication experiments need more exploration, but it can be seen that even pinging can affect the traffic behavior, and communication can be affected if one user keeps busy another node in an indefinite loop.

4.5. SSH Security Issues in ProtoGENI

SSH is considered as a comparatively secure way to communicate remotely but there are many security issues associated with SSH. The protocol, default port settings are causes of security concerns. Port configuration to a non-standard port, and disabling logins via SSH for the root account may help to increase the required work done in attacking the network. Experiments shows that port scanner like nmap, zenmap (GUI) can scan open ports and associated services. Default settings of SSH client and server can be vulnerable for attacks.

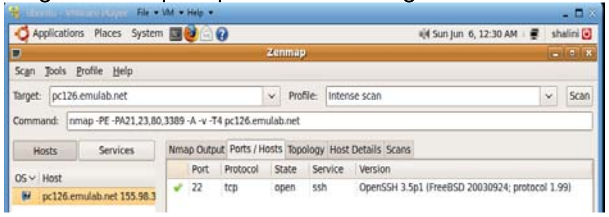
Non-standard port for SSH and some other changes may help in protecting and reducing brute force attacks via SSH. Experiments show to follow certain steps to make a more secure SSH. In this section, the discussion of the methods to activate these defense mechanisms will take place.

Zenmap is GUI for nmap and has different types of scan. We used intense scan to get port details, shown in Figs. 22, 23.


File VM Help
Applications Places System
Wed Jun 9, 12:07 PM shalini
Zenmap
ools Profile Help
pc103.emulab.net
Profile: Intense scan
d: nmap -PE -PA21,23,80,3389 -A -v -T4 pc103.emulab.net
Services
Nmap Output Ports / Hosts Topology Host Details Scans
ost
55.98.36.103

```
nmap -PE -PA21,23,80,3389 -A -v -T4 pc103.emulab.net
Warning:  You are not root -- using TCP pingscan rather than ICMP

Starting Nmap 4.76 ( http://nmap.org ) at 2010-06-09 12:06 PDT
Initiating Ping Scan at 12:06
Scanning 155.98.36.103 [4 ports]
Completed Ping Scan at 12:06, 2.75s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 12:06
Completed Parallel DNS resolution of 1 host. at 12:06, 0.02s elapsed
Initiating Connect Scan at 12:06
Scanning pc103.emulab.net (155.98.36.103) [1000 ports]
Discovered open port 22/tcp on 155.98.36.103
Connect Scan Timing: About 11.40% done; ETC: 12:11 (0:04:01 remaining)
```

Fig. 22: Nmap output after scanning an IP address


File VM Help
Applications Places System
Sun Jun 6, 12:30 AM shalini
Zenmap
Scan Tools Profile Help
Scan
Hosts
Services
Nmap Output Ports / Hosts Topology Host Details Scans
OS Host
pc126.emulab.net 155.98.3

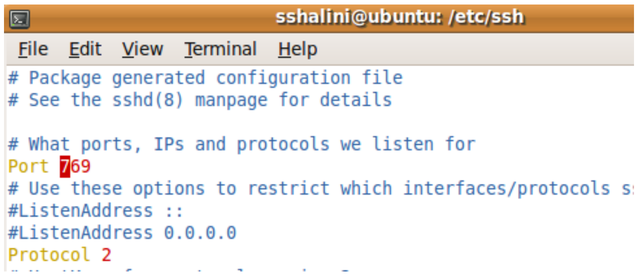
Target: pc126.emulab.net
Profile: Intense scan
Command: nmap -PE -PA21,23,80,3389 -A -v -T4 pc126.emulab.net

	Port	Protocol	State	Service	Version
✓	22	tcp	open	ssh	OpenSSH 3.5p1 (FreeBSD 20030924; protocol 1.99)

Fig. 23: OS details of a machine through zenmap scan

The user can go to /etc/ssh to find the sshd_config file to make changes. Port 22 is set as default for SSH services. This default setting is vulnerable to exploit through automated attacking tools. The port setting in sshd_config file can be changed to a suitable port number. SSH-server will listen on any unused port among 65,535 ports provided by the TCP protocols [83]. Current available port scanners are not capable to detect each and every detail of all 65,535 ports and usually developed to detect most common default settings.

Some other details so using a non-standard port number can be helpful to keep information secured, which will reduce the attacking possibilities. File sshd_config can be modified as per Fig 24. Nmap like scanning tools can scan around 1600 ports by default so other than default port settings it is a bit difficult to trace open non-standard ports. While it is not a preferred method to apply security, but it works most of the time as experiments shows that there were no attacks after changing default SSH port settings from 22 to a non-standard port number.



```
# Package generated configuration file
# See the sshd(8) manpage for details

# What ports, IPs and protocols we listen for
Port 769
# Use these options to restrict which interfaces/protocols s
#ListenAddress ::
#ListenAddress 0.0.0.0
Protocol 2
```

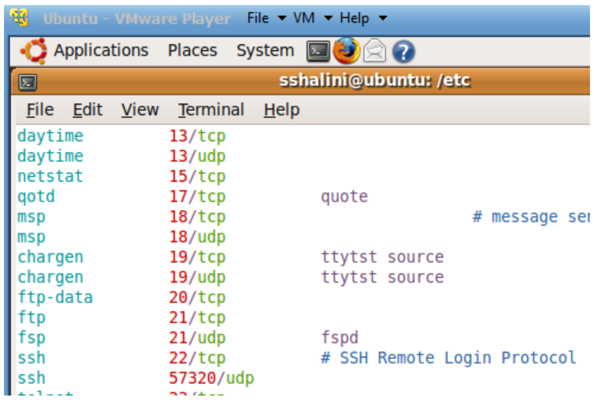
Fig. 24: Modification in sshd_config file for non-standard port for SSH

During the same time period, over 100,000 attacks were reported on systems with default port settings [84]. The only additional work here seems to coordinate with all legitimate users to convey the right port settings for communication. After setting the port 22 to a non standard port number, as shown in Fig 25, settings should also be modified accordingly in /etc/services.

Home:~$ cd /etc

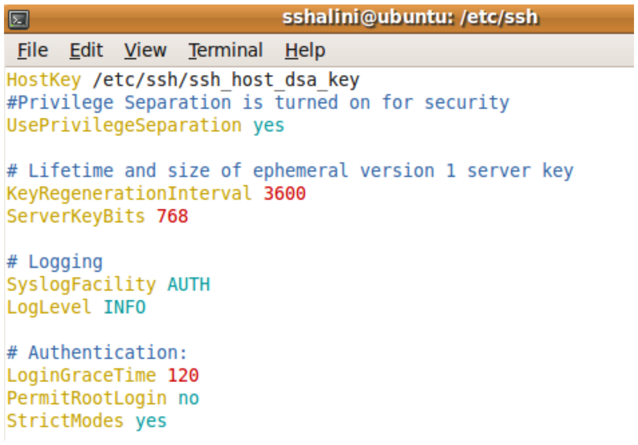
Home:/etc$ sudo vim services

Here, there was a change to the only incoming port settings to new non-standard port for SSH connection because the remote Emulab machines have settings for default settings for SSH as port 22. There was also a modified PermitRootLogin as No in sshd_config file so now open port detection is difficult, and there is also no permission to access SSH remotely that is shown in Fig 26. This exercise is conducted to make SSH more secure and previous research supports and advocates customized settings to make it more difficult for attackers and not to be hacked by automated attacking scripts.



```
daytime         13/tcp
daytime         13/udp
netstat         15/tcp
qotd            17/tcp          quote
msp             18/tcp                          # message se
msp             18/udp
chargen         19/tcp          ttytst source
chargen         19/udp          ttytst source
ftp-data        20/tcp
ftp             21/tcp
fsp             21/udp          fspd
ssh             22/tcp          # SSH Remote Login Protocol
ssh             57320/udp
```

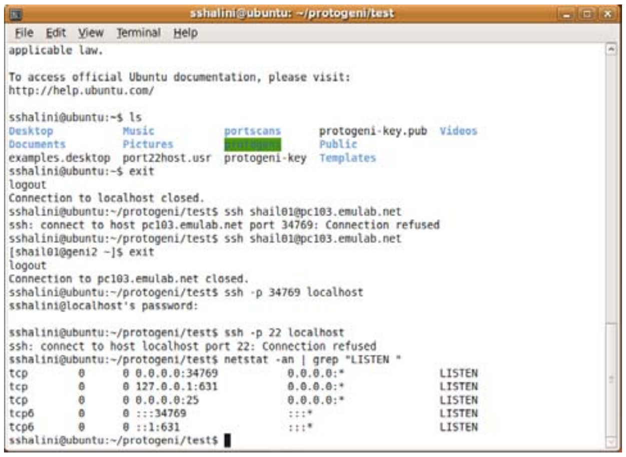
Fig. 25: Changing port no to non-standard port in services



```
HostKey /etc/ssh/ssh_host_dsa_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 768

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes
```

Fig. 26: Restricting Root login via SSH

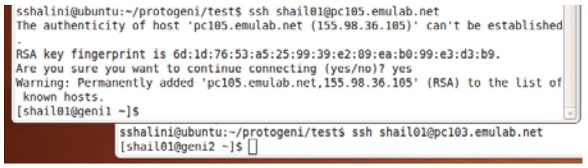
After completing all settings to work with new SSH non-standard port, there was an attempt to login to Emulab machines. Prior to that there also was a verified that system, listening to that new port and could not be reached on default port 22 to access SSH as per Fig 27 and Fig 28.



```
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/

sshalini@ubuntu:~$ ls
Desktop          Music          portscans      protogeni-key.pub  Videos
Documents        Pictures       [illegible]    Public
examples.desktop port22host.usr protogeni-key  Templates
sshalini@ubuntu:~$ exit
logout
Connection to localhost closed.
sshalini@ubuntu:~/protogeni/test$ ssh shail01@pc103.emulab.net
ssh: connect to host pc103.emulab.net port 34769: Connection refused
sshalini@ubuntu:~/protogeni/test$ ssh shail01@pc103.emulab.net
[shail01@geni2 ~]$ exit
logout
Connection to pc103.emulab.net closed.
sshalini@ubuntu:~/protogeni/test$ ssh -p 34769 localhost
sshalini@localhost's password:

sshalini@ubuntu:~/protogeni/test$ ssh -p 22 localhost
ssh: connect to host localhost port 22: Connection refused
sshalini@ubuntu:~/protogeni/test$ netstat -an | grep "LISTEN "
tcp        0      0 0.0.0.0:34769           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN
tcp6       0      0 :::34769                :::*                    LISTEN
tcp6       0      0 ::1:631                 :::*                    LISTEN
sshalini@ubuntu:~/protogeni/test$ 
```

Fig. 27: Verification of new port working as listening port

Now, the verified network operation to connect to Emulab machines in our sliver to check proper working of SSH with non-standard port is in check. System activities may be monitored through logwatch package as shown in Fig 28.



```
sshalini@ubuntu:~/protogeni/test$ ssh shail01@pc105.emulab.net
The authenticity of host 'pc105.emulab.net (155.98.36.105)' can't be established
.
RSA key fingerprint is 6d:1d:76:53:a5:25:99:39:e2:89:ea:b0:99:e3:d3:b9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'pc105.emulab.net,155.98.36.105' (RSA) to the list of
 known hosts.
[shail01@geni1 ~]$

sshalini@ubuntu:~/protogeni/test$ ssh shail01@pc103.emulab.net
[shail01@geni2 ~]$ 
```

Fig. 28: User login to Emulab machines with changed SSH port

5. Analysis on Results and Observations

5.1. Attack through modifying test-common.py script:

test-common.py test script is being used by all other scripts if the attacker has access to the physical machine he can inject simple messages as print outputs without disturbing the actual process of script to fool users. It may stop users to try any further and thus stopping the execution of ProtoGENI experiments by that user.

5.2. Non-Availability of ProtoGENI Resources:

Non-Availability of ProtoGENI Resources:

1) *Outage of Resources*: Experiments show that ProtoGENI can face outage of resources for other experimenters if few experiments keep reserving resources without proper utilization.

The system did not give any warnings, while all resources were being reserved by one user only.

Future work: This outage was for short time and after verification of problem, we released the resources, it may be subject of further exploration that after what time, system can identify this unusual resource reservation by one user.

2) *Failing to map the resources as per RSpec* : Many times, we couldn't create a sliver as requested because it failed to map the requested resources. A particular type of pc was not free; a particular pc was down; the node was free, available, but could not grasp for experiment. Third situation is certainly of concern, as we couldn't get the reason for not getting the resource. Characteristics of a certain pc type like virtnode_capacity also restricted the acquisition of a node in experiment.

One experiment couldn't be activated as one fix node was required with other wireless nodes. It took almost 4 hr. while tried with several free and up nodes been shown on Emulab, but system denied each time stating that fix node is not available. This problem indicates that even after system showing everything o.k. and good through sliverstatus.py, some other problems may halt the access to resources that may hurt the accessibility of resources.

5.3. Non-usability of ProtoGENI Resources:

Non-usability of ProtoGENI Resources:

Resources not usable: In some cases, resources were allotted to sliver, but could not get hold on those nodes mainly because they were reported to be nodes with old OS image or old machines with no consistency in performance. Sliver sharednode was created successfully, and get two nodes virtually shared. Sliverstatus.py showed one node 'ready' and one as 'not ready,' which did not come up even after hours. Therefor, it could not be used any further.

Suggestion: After defined period, system resources should be evaluated for their performance consistency and a list of available resources and other related issues like OS images should be updated accordingly. Related error message or advising notes may also be helpful to choose the right resources to save time and effort, and moreover to remove the confusion clouds of "what went wrong."

Observations:

Slice/Sliver Creation: During several instances, the system got stuck somewhere in displaying the process of creation of sliver and never returned to normal. On screen, it seems that the process is hung and the only solution is to terminate it by closing the terminal. It is determined that even system does not show any progress or completion of slice/sliver creation, it completes it in the background. If status is checked for the same sliver in another terminal, it shows the existing sliver with its current status. This problem creates confusion and this is required to occurr in different terminals. It consumed some time to understand the problem. Sometimes slivers in sequence couldn't be created in one terminal but when tried in a separate terminal, it worked well.

Different variations to create slice and slivers are confusing and difficult to differentiate for the benefits or problems associated with each. Resources may be acquired in a sliver without registering a slice first or also through getticket operation with or without registration of slice. Currently, slice can handle only one sliver identical to slice name. Ticket operations were quite interesting as getticket provides a certain time to redeem the ticket but it expires within minutes. Even a deleted sliver became active when the ticket was redeemed.

No ticket operation works once a slice/sliver is active so to renew the time for extending the experiment, user has to go through different path to renew the slice/sliver. We could not get the execution part for rleaseticket.py or unregisterslice.py.

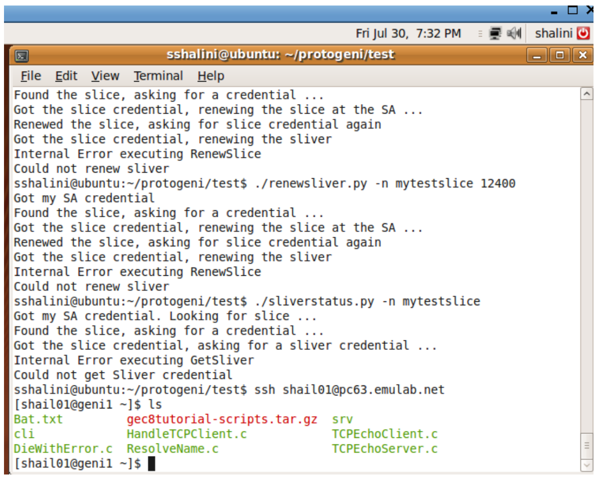
5.4. Deleting the Slice/sliver and release of ProtoGENI Resources:

Resources are released with the deletion of sliver but sliver/slice is kept in system's records until its expiration time at Clearinghouse. Test script showuser.py provides a list of slices for particular users but include all such slices that are not expired, but resources are released. If this includes only active slices holding the resources, it can help user to manage better.

Suggestion: New test script slicestatus.py stating the status of slice as empty/deleted or active

6.5. System bugs: Problem observed in execution of all test scripts

Sometimes bugs interfere and stall the system. During trial of renewing a slice, test script renewsliver.py did not work, shown Fig. 29. No other test scripts could be executed because of a bug. The problem was sent to ProtoGENI people. This bug halted the basic functionality for more than 12 hours.


Fri Jul 30, 7:32 PM
shalini
sshalini@ubuntu: ~/protogeni/test
File Edit View Terminal Help

```
Found the slice, asking for a credential ...
Got the slice credential, renewing the slice at the SA ...
Renewed the slice, asking for slice credential again
Got the slice credential, renewing the sliver
Internal Error executing RenewSlice
Could not renew sliver
sshalini@ubuntu:~/protogeni/test$ ./renewsliver.py -n mytestslice 12400
Got my SA credential
Found the slice, asking for a credential ...
Got the slice credential, renewing the slice at the SA ...
Renewed the slice, asking for slice credential again
Got the slice credential, renewing the sliver
Internal Error executing RenewSlice
Could not renew sliver
sshalini@ubuntu:~/protogeni/test$ ./sliverstatus.py -n mytestslice
Got my SA credential. Looking for slice ...
Found the slice, asking for a credential ...
Got the slice credential, asking for a sliver credential ...
Internal Error executing GetSliver
Could not get Sliver credential
sshalini@ubuntu:~/protogeni/test$ ssh shail01@pc63.emulab.net
[shail01@geni1 ~]$ ls
Bat.txt         gec8tutorial-scripts.tar.gz  srv
cli             HandleTCPClient.c            TCPEchoClient.c
DieWithError.c  ResolveName.c                TCPEchoServer.c
[shail01@geni1 ~]$
```

Fig. 29: System bug created problem in executing test scripts

Here observation was that the acquired node was active and good to login but test scripts were not working.

Other thing is that though bug stopped the new operations on ProtoGENI, users could utilize the resources which were already acquired and up for experiments.

As we are concerned about the threats to ProtoGENI resources, experiments show that though system is basically protected with encryption, authorization, and authentication but system is not consistent in certain functions and prone to bugs that can halt the whole availability and accessibility of ProtoGENI resources. System is also prone to be attacked if host system is compromised and a corrupt test-common.py can affect all other test scripts and so the functionality of ProtoGENI.

Availability and accessibility of resources is very important to do any experiments further to see the network behavior and to identify the ways to breach the system which can be used to improve the overall ProtoGENI functionality.

Emulab resources are allocated to a single user at a time. If nodes are being used by some experimenters, it may hinder the possibility of other experiments to run because of unavailability of resources or because of non-usability of resources for different reasons as described above.

Sometimes a novice experimenter can create many slices to experience different basic experiments and may lost the track of slices created and deleted in process and may hold network resources. This unnoticed holding of resources might create shortage of resources. These resources on hold might be required for more important and preferable experiments to execute to see network behavior by mature experimenters. A beginner may not be capable to understand the severity of problem by holding few resources that could be resulted as bottleneck for another topology. Though slices and slivers are short lived for few hours if not being renewed, but still some basic experiments observing the effect of extending sliver time of existence through redeemticket.py may hold resources for a longer period that can block theses resources to be obtained by other ProtoGENI users. This poses a threat to resources available to experimenters.

5.6. Vulnerability of Wildcard Specification in RSpecs

Emulab has different types of resources, and experimenters can define their required type of resources through specifying the details in RSpecs. Sometimes, it may be hard to find the required specific resources because of other running experiments. It is also complex to know each and every detail about the characteristics of a node type to make sure the adequate choice for experiments. Especially, novice experimenters may not understand complexity of details, therefore, it is a general practice to use wildcard allocation of available resources.

It is simple and repeatable process to get resources of Emulab based on the best availability. In certain cases, the system may assign some special type of PCs which are very few in numbers, and because of holding of resources; they may be unavailable for some time depending on experimenter acquired those PCs. Subsequently, this may cause a bottleneck for other experimenters who may need that special kind of PC in their topology. Therefore, many other available resources would not help, as sliver cannot be created without availability of all asked resources in RSpecs. As shown in Fig. 30, a network topology can suffer by unavailability of resources because of wildcard allocation through RSpecs.

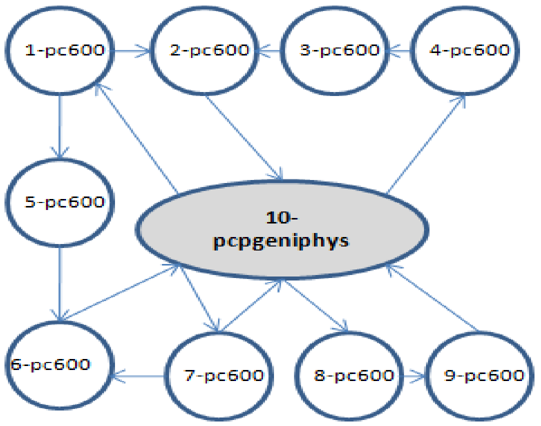


Fig. 30: A network topology which may suffer due to wildcard allocation of resources.

In this example, if 10- pcpgeniphys has allotted to some other experiment through wildcard, the topology will be very restricted to perform the desired operations or interconnections among all nodes of topology.

For example, if 10-pcpgeniphys is not available, 6-pc600 will be almost isolated with all other nodes. 2,3,4 pc600 and 1,5,6 pc600 can have some usability in this topology. With the availability of 10-pcpgeniphys, the network topology may have much more interaction and functionality possible in the experiments.

5.7. Run-time Interactions between ProtoGENI Components:

Communication between nodes of same sliver: it was good in both ways as client as well as server

Communication between nodes of two slivers: good in both ways. Communication between nodes of two Component Managers in two slivers: good in both ways. Communication with outside network and ProtoGENI sliver: good when ProtoGENI sliver node acts as server, but couldn’t establish connection when tried in reverse. Though there is some clarification is required for not successful communication between a ProtoGENI node as client and outside network node as server, there is an expection that more experiments will be helpful to explore the reasons.

Wireless Communication in Emulab: Traffic load can force system resources to utilize its resources at best in concurrency that may lead to significant delays in some processes and loss of packets. It can also affect the communication efforts by other network nodes.

Again there are some unsolved issues and need more experiments. ProtoGENI uses the same resources in same manner but in sliver. Further work is supposed to repeat settings in ProtoGENI sliver settings.

Observation: All wireless nodes with required specification were busy in other experiments and the group had to wait for 6 days then a request was sent to

Emulab people and they provided the required type of nodes by reserving them.

5.8. Attack via stealing user credentials

Passphrase: Once a sliver is created and resources are acquired, the passphrase may be deleted from the host machine through forgetpassphrase.py. The user can login to remote nodes and can do further actions which leaves the host machine a bit more secured as there is no passphrase to steal.

5.9. Attack via SSH

Experiments explore the problems related to SSH (Secure Shall), possible ways to attack via SSH and methods to restrict attackers to attack via SSH like changing the default SSH port no and restricting other users as root user.

More experiments are required to see actual effort and amount for an outsider to collect data and to perform attack. It would be helpful to create a setup as honey-pot and then analyzing outcomes before and after these SSH settings.

The group would like to work on the possibility of defining a non-standard SSH port for all ProtoGENI nodes. Further experiments may explore the better and more secure ways to yield and utilize ProtoGENI resources.

6. Future Work

These experiments deal with basic RSpecs and an effort to see the consumption and release pattern of Emulab resources with creation and deletion of slices and slivers. Future work is proposed to experiment with different combinations of wildcard allocation and specific kind of PCs in one RSpec. More study is necessary to understand detailed description of Emulab node types and their characteristics to incorporate them in the RSpec to create desired network experiments. More elaborate experiments are required to analyze run-time interactions with different network settings. SSH attacking possibilities also may be verified with specific setup as honey-pot for more concrete results on attacking possibilities.

Further studies on security issues and progress of GENI and ProtoGENI are supposed to continue with evolving systems.

7. Conclusions

GENI and ProtoGENI security architecture with operational security issues, and security issues associated with ProtoGENI components are discussed. Security threats to existing ProtoGENI system are identified which can help to improve overall ProtoGENI/GENI security.

The group conducted experiments to exploit possible threats to availability and accessibility of resources in ProtoGENI experiments. Communication between ProtoGENI slivers and slices were observed under different network conditions. The Emulab experiment focused on wireless communication, which may be further explored for more analysis.

Though ProtoGENI showed a certain degree of security through authentication and privilege mechanisms, experiments showed that whole ProtoGENI may face serious threats for outage of resources, non-availability, and non-accessibility of resources. ProtoGENI network traffic is also vulnerable to unwanted traffic loads and attacks through SSH or Host machine's vulnerability. It is also observed that certain operational glitches can halt the whole system for several hours. Further experiments may assist to demonstrate the severity of these issues.

GENI and ProtoGENI are evolving and not in a stable state. The group wishes to receive further updates on GENI security architecture and technologies applied; also, expecting to do more ProtoGENI experiments to explore different security issues in network experiments.

Acknowledgements.
This work is supported in part by The Global Environment for Network Innovations (GENI) (via BBN Technologies from NSF) with Project # 1783, entitled "GENI Experiments for Traffic Capture Capabilities and Security Requirement Analysis", the National Natural Science Foundation of China (NSFC) under grant 61374200, and the U.S. National Science Foundation (NSF) under grants: No. 1427838, CNS-0716211, CCF-0829827, CNS-0737325, and CNS-1059265. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] E. McCary and Y. Xiao, "Smart Grid Attacks and Countermeasures," EAI Endorsed Transactions on Industrial Networks and Intelligent Systems, Vol. 2, No. 2, pp. e4, Feb. 25, 2015, doi: 10.4108/inis.2.2.e4
[2] Y. Xiao, "Accountability for Wireless LANs, Ad Hoc Networks, and Wireless Mesh Networks," IEEE Communication Magazine, Vol. 46, No. 4, Apr. 2008, pp. 116-126.
[3] Y. Zhang, Y. Xiao, K. Ghaboosi, J. Zhang, and H. Deng, "A Survey of Cyber Crimes," (Wiley Journal of) Security and Communication Networks, Vol. 5, No. 4, pp. 422–437, Apr. 2012.
[4] S. Malliga and A. Tamilarasi, "A backpressure technique for filtering spoofed traffic at upstream routers," International Journal of Security and Networks, Vol. 5, No.1 pp. 3 - 14, 2010.
[5] C. Hsieh, J. Chen, Y.-B. Lin, K. Chen, H. Liao, and C. Liang, "NTP-DownloadT: a conformance test tool for secured mobile download services," International Journal of Security and Networks, Vol. 3, No. 4 pp., 240 - 249, 2008.
[6] B. Sun, L. Osborne, Y. Xiao, and S. Guizani, "Intrusion Detection Techniques in Mobile Ad Hoc and Wireless Sensor Networks," IEEE Wireless Communications Magazine, Oct. 2007, pp. 56-63.
[7] B. Sun, K. Wu, Y. Xiao, and R. Wang, "Integration of mobility and intrusion detection for wireless Ad Hoc networks," (Wiley) International Journal of Communication Systems, Vol. 20, No. 6, pp. 695-721, Jun. 2007.

[8] Y. Xiao, "Flow-Net Methodology for Accountability in Wireless Networks," IEEE Network, Vol. 23, No. 5, Sept./Oct. 2009, pp. 30-37.
[9] GENI: Global Environment for Network Innovations. https://www.geni.net/
[10] FIND: Future Internet Design, http://www.nets-find.net/
[11] FIRE: Future Internet Research and Experimentation, http://en.wikipedia.org/wiki/Future_Internet_Research_and_Experimentation
[12] AKARI, New Generation Network Architecture AKARI Conceptual Design (ver1.1) , http://web.archive.org/web/20100414082316/http://akari-project.nict.go.jp/eng/concept-design/AKARI_fulltext_e_translated_version_1_1.pdf
[13] NICT: National Institute of Information and Communication Technology http://www.nict.go.jp/en
[14] D. D. Clark, " Toward the design of a Future Internet", Version 7, Oct 2009
[15] T. Magedanz and S. Wahle " Control framework design for Future Internet" Springer Eelktrotechnik & Informationstechnik(2009), p. 274-279
[16] J. Roberts, "The clean-slate approach to future Internet design: a survey of research initiatives" Ann. Telecommun. 2009, p 271-276
[17] T. Spyropoulos, S. Fdida, and S. Kirkpatrick, "Future Internet: Fundamentals and Measurement" [Report of the COST Arcadia Future Internet Workshop]
[18] https://www.emulab.net/
[19] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "*An Integrated Experimental Environment for Distributed Systems and Networks*", appeared at OSDI 2002, December 2002
[20] H. F. Lipson, "Tracking and Tracing Cyber-Attacks: Technical Challenges and Global Policy Issues, CERT® Coordination Center, November 2002, SPECIAL REPORT
[21] Protecting America's Freedom in the Information Age. Markle Foundation, 2002. http://www.markle.org/downloadable_assets/nstf_full.pdf
[22] D. Brin, 'The Transparent Society', Addison-Wesley, April 1998
[23] D. Weitzner, "Beyond Secrecy: New Privacy Protection Strategies for Open Information Spaces," IEEE Internet Computing, Sept/Oct 2007.
[24] D.J. Weitzner, et al., "Transparent Accountable Data Mining: New x Discretionary, rule-based access for the world wide web, http://www.w3.org/2006/01/tami-privacy-strategies-aaai.pdf
[25] Thuraisingham, editors, Web and Information Security. IRM Press, 2006
[26] http://www.sparta.com/
[27] B. Fu and Y. Xiao, "A Multi-Resolution Accountable Logging and Its Applications," Computer Networks, Vol. 89, No. 4, Oct. 2015, Pages 44–58.
[28] Y. Xiao, K. Meng, and D. Takahashi, "Accountability using Flow-net: Design, Implementation, and Performance Evaluation," (Wiley Journal of) Security and Communication Networks, Vol.5, NO. 1, pp. 29–49, Jan. 2012,
[29] H. Chen, Y. Xiao, X. Hong, F. Hu, J. Xie, "A Survey of Anonymity in Wireless Communication Systems," (Wiley Journal) Security and Communication Networks, Vol. 2 No. 5, Sept./Oct., 2009, pp. 427-444.
[30] D. G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and C. Shenker, Accountable Internet Protocol (AIP), In Proc. ACM SIGCOMM, Aug 2008
[31] M. Huang, A. Bavier, and L. Peterson, "PlanetFlow: Maintaining accountability for network services," ACM SIGOPS Operating Systems Review, v.40 n.1, January 2006
[32] K. Argyraki, P. Maniatis, O. Irzak, A. Subramanian, and S. Shenker, "Loss and Delay Accountability for the Internet", ICNP 2007 Beijing, China.
[33] S. Yue, Y. Xiao, and G. Xie, "Experiments on an Election Algorithm for Decision Element Failures in 4D Future Internet Architecture," Proceedings of The 2009 International Conference on Future Generation Communication and Networking (FGCN 2009), FGCN/ACN 2009, CCIS 56, pp. 250-258, 2009. Springer
[34] S. Yue, Y. Xiao, and G. Xie, "Fault Tolerance Experiments in 4D Future Internet Architecture," Journal of Internet Technology, Vol. 11 No. 4, pp. 543-552, July 2010.
[35] J. Gao, Y. Xiao, S. Rao, and F. Shalini, "Security Tests and Attack Experimentations of ProtoGENI," International Journal of Security and Networks, Vol. 10, No. 3, 2015, pp. 151-169.
[36] http://www.protogeni.net/trac/protogeni
[37] ProtoGENI CF Overview, 022709 GENI-SE-CF-ProtoGENIOver-01.4, February 27, 2009
[38] http://www.protogeni.net/trac/protogeni/wiki
[39] Z. Xiao and Y. Xiao, "Accountable MapReduce in Cloud Computing," in Proceedings of 2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2011, pp. 1082 - 1087.
[40] Z. Xiao and Y. Xiao "Achieving Accountable MapReduce in Cloud Computing," (Elsevier) Future Generation Computer Systems, Vol. 30, No.1, Jan. 2014, pp. 1–13.
[41] S. Yue and Y. Xiao "Building Global View with Log Files in a Distributed/Network System," Proceedings of the IEEE Global Telecommunications Conference 2010 (IEEE GLOBECOM 2010).
[42] Y. Xiao, S. Yue, B. Fu, and S. Ozdemir, "GlobalView: Building Global View with Log Files in a Distributed / Networked System for Accountability," (Wiley Journal of) Security and Communication Networks, Vol. 7, No. 12, pp. 2564–2586, Dec. 2014.
[43] J. Gao, Y. Xiao, S. Rao, and F. Shalini, "Security Tests of ProtoGENI and Attack Experimentations," Proceedings of 2011 International Conference on Security Science and Technology (ICSST 2011), pp. 186-190.
[44] J. Gao and Y. Xiao, "ProtoGENI DoS/DDoS Security Tests and Experiments," Proceedings of First GENI Research and Educational Experiment Workshop (GREE12), in conjunction with GENI GEC 13.
[45] J. Gao, Y. Xiao, S. Rao, and F. Shalini, "Security Tests and Attack Experimentations of ProtoGENI," International Journal of Security and Networks, Vol. 10, No. 3, 2015, pp. 151-169.
[46] Z. Xiao, B. Fu, Y. Xiao, C. L. P. Chen, and W. Liang, "A Review of GENI Authentication and Access Control Mechanisms," International Journal of Security and Networks (IJSN), Vol. 8, No. 1, 2013, pp. 40-60.
[47] F. Shalini, Y. Xiao, and B. Sun, "ProtoGENI Experiments," International Journal of Security and Networks, accepted
[48] GENI Security Architecture, Spiral 1 Draft 0.55, July 31th, 2009
[49] GENI Facility Security, GDD-06-23, Distributed Services working group, Draft work in progress (version 0.5)
[50] GENI Security Architecture, Spiral 2 Draft 0.5, March 15th, 2010
[51] GENI: Towards Operational Security For GENI, Draft, GDD-06-10, July 2006
[52] http://en.wikipedia.org/wiki/Secure_Shell#Security_issues
[53] http://www.networkworld.com/news/2008/062408-sloppy-virtualization.html
[54] http://en.wikipedia.org/wiki/Port_scanner
[55] L. Zeng, H. Chen, and Y. Xiao, "Accountable Administration and Implementation in Operating Systems," Proceeding of The IEEE Global Telecommunications Conference 2011 (IEEE GLOBECOM 2011).
[56] L. Zeng, Y. Xiao, and H. Chen, "Accountable Logging in Operating Systems," Proceedings of The IEEE International Conference on Communications 2015 (IEEE ICC 2015).
[57] L. Zeng, Y. Xiao, H. Chen, "Auditing Overhead, Auditing Adaptation, and Benchmark Evaluation in Linux," (Wiley Journal of) Security and Communication Networks, accepted. DOI: 10.1002/sec.1277.
[58] M. Bishop, "Security Problems with the UNIX Operating System", version2, January 31, 1983[unpublished]
[59] M. Bishop, "Reflections on UNIX Vulnerabilities" , Computer Security Applications Conference, 2009. ACSAC 2009. Annual, Page(s): 161 - 184
[60] M. Bishop, C. Gates, D. Frincke, and F. Greitzer, "AZALIA: an A to Z assessment of the likelihood of insider attack", Technologies for Homeland Security, 2009 IEEE, Page(s): 385 – 392
[61] M. Lei, Y. Xiao, S. V. Vrbsky, and C.-C. Li, "Virtual Password Using Random Linear Functions for On-line Services, ATMs, and

Pervasive Computing," Computer Communications Journal, Elsevier, Vol. 31, No. 18, Dec. 2008, pp. 4367-4375.
[62] Y. Xiao, C.-C. Li, M. Lei, and S. V. Vrbsky, "Differentiated Virtual Passwords, Secret Little Functions, and Codebooks for Protecting Users from Password Theft," IEEE Systems Journal, Vol. 8, No. 2, Jun. 2014, pp. 406-416.
[63] http://www.planet-lab.org/
[64] S. Goyal and J. Carter, "A Lightweight Secure Cyber Foraging Infrastructure for Resource-Constrained Devices ," http://www.cs.utah.edu/~retrac/wmcsa04.pdf
[65] http://www.cl.cam.ac.uk/research/srg/netos/projects/archive/xeno/
[66] M. Fullmer, "The OSU Flow-tools Package and Cisco NetFlow Logs," https://www.usenix.org/legacy/publications/library/proceedings/lisa2000/full_papers/fullmer/fullmer_html/index.html
[67] M. Castro and B. Liskov, "Practical Byzantine Fault-Tolerance and Proactive Recovery," *ACM Trans. Computer Systems*, vol. 20, no. 4, pp. 398-461, 2002.
[68] A. R. Yumerefendi and J. S. Chase, "Strong Accountability for Network Storage," http://www.cs.duke.edu/nicl/pub/papers/cats-fast07.pdf
[69] Y. Xiao, K. Meng, and D. Takahashi, "Implementation and Evaluation of Accountability using Flow-net in Wireless Networks," Proceedings of the IEEE Military Communications Conference 2010 (IEEE MILCOM 2010), pp. 7 - 12 .
[70] Z. Xiao, Y. Xiao, and H. Chen, "An Accountable Framework for Sensing-Oriented Mobile Cloud Computing," Journal of Internet Technology, Vol. 15, No.5, pp. 813-822, 2014.
[71] Z. Xiao, Y. Xiao, and D. Du, "Building Accountable Smart Grids in Neighborhood Area Networks," Proceeding of The IEEE Global Telecommunications Conference 2011 (IEEE GLOBECOM 2011).
[72] J. Liu, Y. Xiao, and J. Gao, "Achieving Accountability in Smart Grids," IEEE Systems Journal, Vol. 8, No. 2, Jun. 2014, pp. 493-508.
[73] Z. Xiao, Y. Xiao, and D. Du, "Non-repudiation in Neighborhood Area Networks for Smart Grid," IEEE Communications Magazine, Vol. 51, No. 1, pp. 18-26, Jan. 2013.
[74] Z. Xiao, Y. Xiao, and D. Du, "Exploring Malicious Meter Inspection in Neighborhood Area Smart Grids," IEEE Transactions on Smart Grid, Vol. 4, No. 1, Mar. 2013, pp. 214-226.
[75] J. Liu, and Y. Xiao, "Temporal Accountability and Anonymity in Medical Sensor Networks," ACM/Springer Mobile Networks and Applications (MONET), Vol. 16, No. 6, pp. 695-712, Dec. 2011.
[76] B. Fu and Y. Xiao, "Q-Accountable: A Overhead-based Quantifiable Accountability in Wireless Networks," Proceedings of IEEE Consumer Communications and Networking Conference (IEEE CCNC 2012), pp. 138-142.
[77] B. Fu and Y. Xiao, "Accountability and Q-Accountable Logging in Wireless Networks", Wireless Personal Communications, Vol. 75, No. 3, Apr. 2014, pp. 1715-1746.
[78] B. Fu and Y. Xiao, "A Multi-Resolution Flow-Net Methodology for Accountable Logging and Its Application in TCP/IP Networks," Proceedings of The IEEE International Conference on Communications 2014 (IEEE ICC 2014).
[79] B. Fu and Y. Xiao, "A Multi-Resolution Accountable Logging and Its Applications," Computer Networks, Vol. 89, No. 4, Oct. 2015, Pages 44–58.
[80] Z. Xiao, Y. Xiao, and J. Wu, "A Quantitative Study of Accountability in Wireless Multi-hop Networks," Proceedings of 2010 39th International Conference on Parallel Processing (ICPP 2010), pp. 198 - 207.
[81] P. Hochmuth, November 11, 2004. LinuxWorld. Linux is 'most breached' OS on the Net, security research firm says. Available at: http://www.linuxworld.com.au/index.php/id188808220;fp;2;fpid;1.
[82] http://www.protogeni.net/trac/protogeni/wiki/Tutorial
[83] J. Owens, J. Matthews, A study of passwords and methods used in brute-force SSH attacks, Available on (last accessed on 06-10-2010): http://people.clarkson.edu/~owensjp/pubs/leet08.pdf
[84] S. Lemon, September 20, 2006. Computer World Security.Bruce Schneier: We are losing the security war. Available at: http://www.computerworld.com/s/article/9003477/Bruce_Schneier_We_are_losing_the_security_war
[85] D. Ramsbrock, R. Berthier, and M. Cukier, 2007. "Profiling Attacker Behavior Following SSH Compromises," in Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp.119-124.
[86] http://www.protogeni.net/trac/protogeni/wiki/RSpecTutorial