Naval Research Laboratory

Washington, DC 20375-5320

NRL/MR/5542--19-9887

Xenon Enterprise Scale Separation VMM Systems

Margery Li

Alexander Velazquez

Center for High Assurance Computer Systems Branch
Information Technology Division

June 5, 2019

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 05-06-2019	**2. REPORT TYPE** NRL Memorandum Report	**3. DATES COVERED** *(From - To)* 1 October 2017 - 30 September 2018
4. TITLE AND SUBTITLE Xenon Enterprise Scale Separation VMM Systems		**5a. CONTRACT NUMBER** N0001419WX00141
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER 0602235N
6. AUTHOR(S) Margery Li and Alexander Velazquez		**5d. PROJECT NUMBER** 55-8089-29-5
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER 8089
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory 4555 Overlook Avenue, SW Washington, DC 20375-5320		**8. PERFORMING ORGANIZATION REPORT NUMBER** NRL/MR/5542--19-9887
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research One Liberty Center 875 North Randolph Street, Suite 1425 Arlington, VA 22203-1995		**10. SPONSOR / MONITOR'S ACRONYM(S)**
		11. SPONSOR / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A: Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Enterprise scale cloud computing for system resource sharing has become increasingly common as virtualization offers quicker system deployment and reduced overhead and costs compared to its physical counterpart. The key challenges for cloud computing technologies are the preservation of strong separation and the fulfillment of security requirements in a virtual environment that is agile and heterogeneous in nature. In this paper, we present Xenon Enterprise to meet these challenges. Xenon Enterprise is a management platform that offers secure virtual workspaces to its users by provisioning hardware resources of hosts running Xenon Virtual Machine Monitor (VMM). To understand the design concept of Xenon Enterprise and the enterprise services it provides, we start with a discussion of the Xenon security model and Xenon security policy components. Next, we provide in-depth description of the Xenon management tool stack for implementing the security policy components and enforcing them in Xenon VMMs. After the policy essentials, we demonstrate how the enterprise services can be utilized to construct a sample security policy on Xenon Enterprise that meets the requirements of strong separation and security enforcement.

15. SUBJECT TERMS

Xenon virtual machine monitor	Virtual machine	Xenon policy xenon
Policy editor xenon enterprise	MSM security module	Hypercalls

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Margery Li
a. REPORT Unclassified Unlimited	**b. ABSTRACT** Unclassified Unlimited	**c. THIS PAGE** Unclassified Unlimited	Unclassified Unlimited	44	**19b. TELEPHONE NUMBER** *(include area code)* (202) 404-4920

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

This page intentionally left blank.

1 Brief Description of Xenon Enterprise Virtualization Platform and Xenon Hosts

In the age of cloud computing, where resource pooling is becoming ubiquitous, Xenon Enterprise offers a hybrid-cloud environment for resource management of multiple hosts running the Xenon Virtual Machine Monitor (VMM).

Xenon VMM is a type-1 hypervisor that provides advanced virtualization security by strongly separating virtual machines (VMs) that run on the same host hardware. Separation in Xenon VMM is achieved with a policy-driven Xenon MSM security module, which enforces simple inter-VM policies by following the separation kernel approach. The module consists of two parts: a separation policy and a communication policy. A separation policy governs isolation of resources (virtual memory, virtual interrupts, or physical hardware) between VMs. A communication policy governs inter-VM communications on the hypercall level and the network level.

A host that runs Xenon VMM is referred to as Xenon host. A standalone Xenon host consists of three integral parts: Xenon Hypervisor, Xenon policy server, and Xenon policy editor. Xenon Hypervisor is based on Xen VMM. However, unlike its upstream counterpart, Xenon Hypervisor and its control plane (dom0) run on a refactored code base that is highly modular and reduced in cyclomatic complexity. The code modularization decouples code dependencies and reduces Xenon's attack surface making Xenon a more secure platform than upstream Xen. Xenon policy server (XPS) is a daemon that runs on dom0. It provides an API for clients to configure Xenon MSM security module using Xenon Security Policy. Xenon policy editor (XPGUI) is the reference client implementation that manages XPS and displays real-time security policy related information with a graphical user interface. It also provides tools for configuring and managing hardware resources and VMs governed by the policy.

Xenon Enterprise (XE) is a virtualization platform that provides enterprise-scale server management of Xenon hosts. However, unlike other modern cloud computing platforms, XE provides a safe and secure environment that reduces security threats among Xenon hosts and their VMs – something that commonly lacks in its counterparts. To mitigate threats and preserve separation, XE implements a composite Xenon Security Policy, which every Xenon host enforces.

The XE virtualization platform can run on any modern Linux-based infrastructure. It consists of a frontend and a backend. On the frontend, XE provides Xenon Enterprise graphical user interface (XEGUI) for accessing and managing the virtualization environment. On the backend, a set of Xenon Enterprise services are available to support policy-related tasks and requests from XEGUI on the frontend. An overview of the XE virtualization platform with Xenon hosts in relation to XE services is presented in FIGURE 1.

The purpose of this technical report is to present the NRL Xenon Enterprise prototype, which manages Xenon hosts to form an enterprise-scale virtualization environment. The remainder of this report is organized as follows. In SECTION 2, we present Xenon security module, which are the building blocks for Xenon Security Policy. Next, we go over Xenon host and guest configuration in the context of Xenon Security Policy. In SECTION 3, we discuss the Xenon management toolstack, which enforces and implements Xenon Security Policy. This includes XPS and XPGUI for implementing and monitoring the policy in place. Finally, in SECTION 4, we discuss the functionalities of Xenon Enterprise platform and XE services by demonstrating the implementation of a simple XE security policy using XEGUI.

Manuscript approved September 10, 2018.

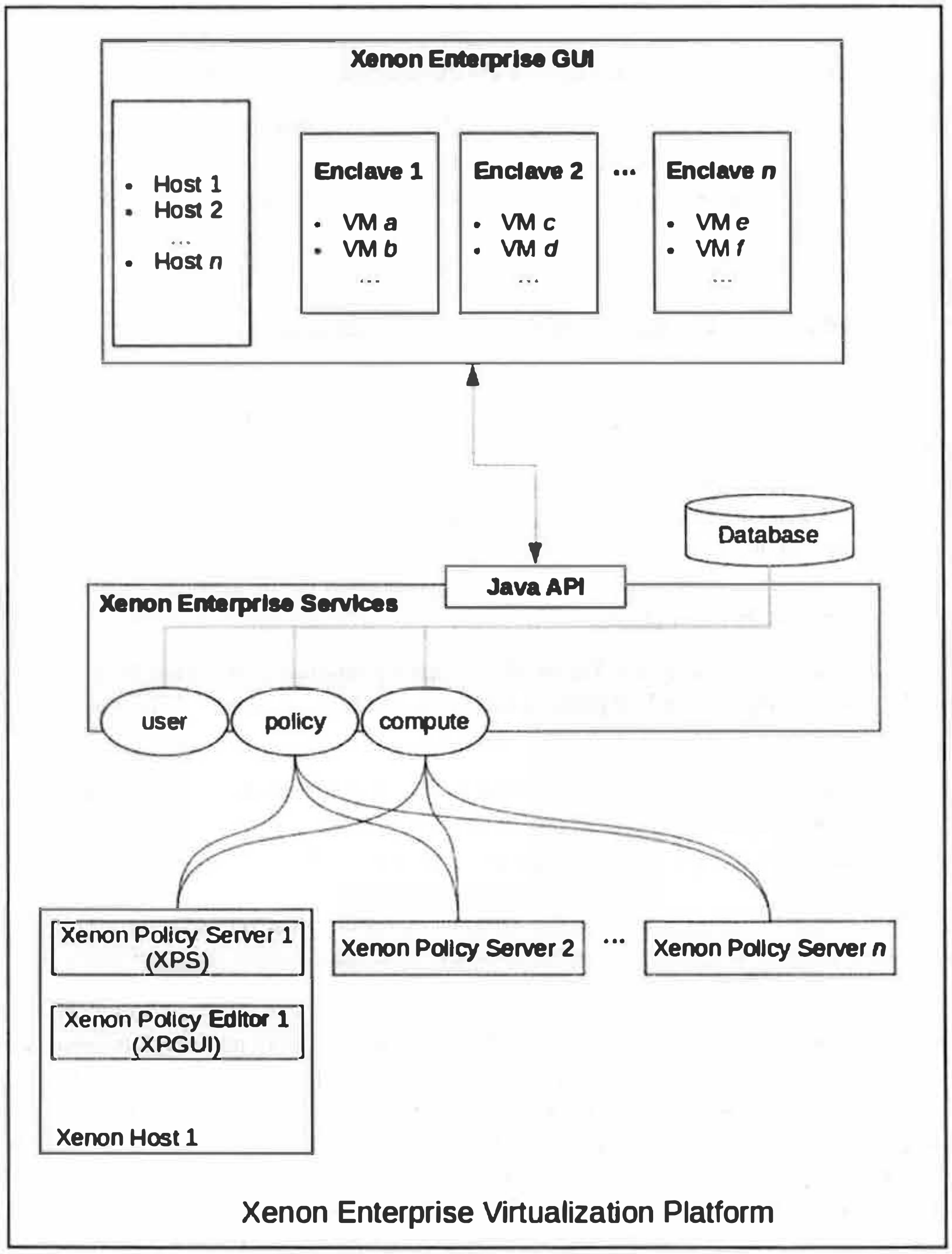


Figure 1 - Xenon Enterprise System Components

2 Xenon VMM and Xenon MSM Security Module

Xenon VMM provides advanced virtualization security by strongly separating virtual machines that run on shared host hardware. Xenon protects against the three significant risks of hardware sharing:

- information could be intentionally exfiltrated from one virtual machine (VM, also referred to as "domain") to another, via the shared hardware,
- one VM could maliciously tamper with the information provided by another VM that shares the same hardware, and
- one VM could shut down or isolate another VM, via the shared hardware.

Commercial virtualization technology provides nominal separation but Xenon is designed to provide significantly stronger separation.

Xenon, which is based on the Xen VMM, reduces the size of the Xen code base by dropping support for selected features that are not needed to run commodity operating systems, e.g. Windows or Linux. Because the internal design and construction of the upstream Xen VMM is highly modular, removing some of the features does not disable any of the remaining features. The Xenon prototype also includes security refactoring. We have changed some of the internal behavior to make Xenon more secure than Xen. For example, Xenon always scrubs free memory pages (overwrites them with zeros) before making them available for re-use by other guests.

Separation in Xenon is enforced by the Xenon MSM security module. The Xenon MSM security module is designed to enforce simple inter-VM policies following the separation kernel approach. The Xenon security model is a simple and intuitive two-part model:

- a *separation policy* component that governs isolation of resources (virtual memory, virtual interrupts, grant tables, or physical hardware) between VMs, and
- a *communication policy* component that governs inter-VM communications.

The Xenon MSM security module also enables administrators to restrict allocation of physical resources, such as PCI slots, to specific VMs, making Xenon ideal for a multi-tenant cloud environment.

Policies enforced by the Xenon security model are easier to verify. The implementation of Xenon MSM security module is an order of magnitude smaller than the Flask security module that comes with Xen, so less work is required to evaluate it. This is not only due to Xenon Security Policy's simplicity, but also because the Xenon MSM security module does not have to conform to the larger Xen community's interface requirements. We have changed the internal structure of Xen by introducing the MSM security module with a set of policy-driven software components. These components are the foundation blocks for building and enforcing Xenon Security Policy. This modular approach of implementing a security apparatus for complex systems makes verifying security checks in Xenon much cleaner and simpler.

2.1 Xenon Security Policy Components

In this section, we briefly go over the essential Xenon security policy components that implement the Xenon MSM security module. The Xenon MSM module is simply a white list interface that provides direct permissions for the objects that should be protected by a separation VMM. Each MSM permission can be viewed as a boolean variable associated with a security-relevant object. If a subject has been assigned the permission, it may use or access the associated object. The objects protected by the MSM module include: hypercalls, hypercall subcommands, hypercall rate, , inter-domain connections, hardware devices and host co-residence.

Under the hood, the white list interface to the Xenon MSM security module is merely a set of hypercall subcommands that can be issued to load and dynamically modified the Xenon Security Policy enforced by XPS running on Xenon hosts. That is, the Xenon Security Policy is an access control list that grants subjects the permission to the objects that need protection by a Xenon host.

To visually display the objects and their permission on the Xenon Security Policy in a way that is intuitive and easy to configure and understand, the hypercall subcommands are abstracted and represented by four Xenon Security Policy components, which we will discuss in details in the subsections that follow.

2.1.1 Hypercall Profiles

A hypercall to a hypervisor is analogous to a syscall to an OS kernel. Xenon guest domains (VMs) use hypercalls to request privileged operations from the hypervisor. A hypercall profile specifies a set of hypercalls that a domain is allowed to make. Hypercall profiles protect the hypervisor from possible hypercall attacks made by compromised domains by reducing the hypervisor's attack surface.

Different types of operating systems may require different hypercall profiles to operate correctly. For example, Linux operating systems such as Ubuntu and CentOS need additional hypercalls compared to a Windows operating system. All unprivileged domains must be assigned a hypercall profile in order to operate correctly.

2.1.2 Tags

The Xenon security model is based on tags. All domains (other than the privileged control plane, dom0) and hardware resources managed by the security policy must be tagged. Enclaves are virtual containers that contain objects of the same tag (more on this later). Each tag is unique system wide. This allows Xenon to provide strong separation among objects of different tags running on the same host.

2.1.3 Conflict Sets

The Xenon MSM module enforces a co-residence protection mechanism referred to as a conflict set A conflict set consists of two or more tags and can specify different levels of communication restriction between objects in different enclaves. A conflict set thus represents a communication policy and affects inter-VM communication between enclaves. For example, a security policy might place a conflict set to disallow domains from two different enclaves to run at the same time. Implementing conflict set requires advanced planning since introducing additional conflict on an active Xenon security policy may require a system reboot.

2.1.4 SecLabels

A SecLabel is a composite object that consists of a hypercall profile, a tag, a guest role, and a max hypercall rate. All unprivileged domains must be assigned a SecLabel; without one, a domain is not allowed to run. The hypercall profile and tag refer to the components described above. The guest role indicates whether the domain is a regular unprivileged user domain or a service domain. The max hypercall rate limits the number of hypercalls a domain can make per millisecond to prevent DoS-type attacks. Exceeding this value is a policy violation and once the maximum number of violations is reached, the domain will be stopped by the hypervisor.

2.2 Xenon Host Configuration

Each Xenon security component on the Xenon Security Policy configures a set of policy rules that the host must accommodate and XPS must enforce. In the section, we go over a set of parameters derived from the Xenon security components that are pertinent to Xenon host configuration.

2.2.1 Guest Type

Description:	Virtualization Spectrum
Supported values:	pvh, pvhvm, hvm, pv

2.2.2 Guest Roles

Description:	Specifies guest domain role, if any
Supported values:	none, network_driver_domain, storage_driver_domain, security_domain

Network and storage driver domains provide hardware resources to unprivileged none-service user domains. Introspection domains monitor both Xenon security policy enforcement and network security for all traffic related to domains running on Xenon Security Policy.

2.2.3 Hypercall Profiles

Description: Specifies which hypercalls a domain is allowed to make. Several pre-defined hypercall profiles ship with Xenon, but other custom profiles can be created and also modified at runtime.

Supported values:
AnyDomU-AllowAll
PV-Combined
PVHVM-Combined
PVH-Combined
PV-LinuxGeneric
PVHVM-LinuxGeneric
PVH-Ubuntu14
PV-Ubuntu14-Security
PV-Ubuntu14-Storage
PVHVM-Ubuntu14-Network
(or any other custom profile)

2.3 Hardware Resources and Domain Configuration File

In the previous section, we discuss the objects on the security policy that need separation protection. In this section, we examine the hardware resources on Xenon hosts that also require permissions to access by the subjects on the policy. The subjects are the guest domains managed by Xenon VMMs.

By default, Xenon boots with a minimal security policy already enforced. The minimal policy contains the control plane, dom0 that are preconfigured with all of its permissions granted. The permissions allow dom0 to add, remove and manage other domains and their permissions on a fine-grained basis at any time after boot. They also give dom0 the ability to allocate system resources to other guests on behave of its Xenon host.

To facilitate dom0 with hardware resource allocation, the XPS daemon running on dom0 routinely performs hardware device discovery during and after Xenon host boot time and presents them on the policy display. We briefly go over commonly available hardware devices, and how dom0 parametrizes and assigns them to the guest domains via domain configuration file.

2.3.1 Hardware Resources Configuration

Every Xenon host provides hardware resources to its guest domains. Typical hardware resources are memory, CPU, network, and disk storage, which can be either internal or external to the host. Xenon Security Policy enforces its separation policy based on tags. That is, every domain (with the exception of dom0) and its hardware resources must be assigned a tag in the security policy in order to run. Resources such as memory and CPU are implicitly tagged when they are specified in a domain configuration file. However, tagging PCI devices and controllers such as network and storage requires additional steps. We will demonstrate the use of PCI devices in SECTION 4.3.

Virtual network switches, provided by network driver domains, provide network connectivity for domains of a given enclave. A virtual network switch can run in one of three different modes: internal, bridged, or NAT. For Xenon hosts running on XE, it is recommended to use a separate bridged network per enclave, for a strong network separation between enclaves of different tags.

Storage driver domains provide storage resources for domains running on the hosts. Unlike network resources, which can be allocated on the fly, storage must be pre-allocated and assigned to domains prior to domain creation. There are two types of storage resources: physical ("phy") and file-based ("file"). A physical storage resource (local disk, partition, logical volume, iSCSI target, etc.) can be assigned to one designated domain in "phy" mode, or a single physical resource can be shared among domains by allocating a disk image file for each domain in "file" mode; in that case, each domain's storage resides on a disk image file stored under a specified directory.

2.3.2 Domain Configuration

Each domain has a configuration file that specifies what hardware resources it utilizes and various configuration options. The configuration file consists of a set of key-value pairs used by the Xenon management toolstack during domain creation. For example, the configuration can specify how much memory and CPU to allocate to the guest or which virtualization spectrum the guest runs on. Most of these values can be dynamically modified on the fly without a system reboot.

3 Xenon Management Toolstack

Xenon management toolstack comes in two parts: Xenon Policy Server (XPS) and Xenon Policy Editor (XPGUI). XPS interfaces with Xenon's MSM security module to enforce the Xenon Security Policy on Xenon hosts and also provides an API that allows clients to manage the policy on the host. XPGUI is a graphical editor that provides access to the security policy and manages XPS instances on demand.

3.1 XPS and XPS Service

The following is a list of XPS features:

- XPS can handle request from multiple local and remote clients.

- XPS handles modification of Xenon's security policy
- XPS detects hardware available on the host, such as: physical CPUs, memory, PCI devices, network interfaces, disks/partitions/volumes, USB devices, input devices
- XPS can set up flexible and secure network configuration to share physical network interfaces among VMs, using a combination of Open vSwitch and iptables
- XPS can provide network configuration via DHCP to VMs
- XPS performs Xenon administrative tasks such as create/pause/shutdown/reboot/destroy VMs, list running domains and their status
- XPS uses the xenstat library to collect resource utilization statistics at regular intervals during runtime

XPS runs as a service on dom0. By default, `xenon-policy.properties` is located at `/etc/xenon-tools/`. This can be altered with the `--properties-file <arg>` command-line option (both `xenon-policy-server` and `xenon-policy-gui` recognize it). All system properties (i.e., built-in Java system properties, and any values passed on the command-line using `-D`) are loaded first, followed by the properties in `xenon-policy.properties` (assuming it exists at the specified location). The script at `/etc/init.d/xenon-policy-server` controls the service, and reads the configuration parameters provided in the properties files.

The following commands stop, start, and then check the status of the XPS service:

```
user@xenon-server:~$ sudo service xenon-policy-server stop
Stopping Xenon Policy Server
user@xenon-server:~$ sudo service xenon-policy-server start
Starting Xenon Policy Server
user@xenon-server:~$ sudo service xenon-policy-server status
Xenon Policy Server is running with pid: [ ... ]
```

3.1.1 XPS Log

With the default configuration, XPS writes log messages to a file at `/var/log/xenon-tools/xenon-policy-server.log`. Every client connection attempt is logged. XPS only accepts connections from local clients and authorized remote clients (details about authorization are described in SECTION 3.1.3).

An unsuccessful connection (e.g., client does not present a certificate and cannot be authenticated) would be logged as follows:

```
2018-09-01 08:00:00 ERROR - catching
javax.net.ssl.SSLPeerUnverifiedException: peer not authenticated
2018-09-01 08:00:00 ERROR - Could not determine peer X500Principal
2018-09-01 08:00:00 ERROR - ReadOnly socket: Closing unauthorized client connection
   from /192.168.1.1
```

An authorized connection from a remote client would be logged as follows:

```
2018-09-01 08:00:01 TRACE - ReadOnly socket: Accepting remote client connection
   from authorized X500Principal
2018-09-01 08:00:01 TRACE - ReadOnly socket: Accepting authorized client connection
   from /192.168.1.1
2018-09-01 08:00:01 INFO  - ReadOnly socket: Client #1 (/192.168.1.1): Ready for
   requests
```

Once the connection is established, the client can make requests. All requests are logged, along with any command that XPS executes on the system in response to the request. For example:

```
2018-09-01 08:00:06 TRACE - ReadOnly socket: Client #1 (/192.168.1.1): Received
   request GetDomainRunStates
2018-09-01 08:00:06 TRACE - Executing command: xl list
2018-09-01 08:00:06 TRACE - ReadOnly socket: Client #1 (/192.168.1.1): Received
   request GetHypervisorInfo
2018-09-01 08:00:07 TRACE - ReadOnly socket: Client #1 (/192.168.1.1): Received
   request GetLogMsgs
2018-09-01 08:00:07 TRACE Executing command: cat /var/log/xen/msm-log-msgs.log
```

When the client closes the connection, a final message is logged, indicating that the socket has been closed:

```
2018-09-01 08:00:15 INFO  - ReadOnly socket: Client #1 (/192.168.1.1): Done
   handling requests, closing socket
```

3.1.2 Rotating XPS Logs

After running for a long time or during periods of heavy usage, the log file may begin to consume considerable disk space. Nevertheless, it is important to maintain the log history for troubleshooting and auditing purposes. To prevent dom0's root filesystem from filling up due to this log, it is recommended to have a separate partition dedicated for logging mounted at `/var/log/xenon-tools/`.

As an additional measure, the `logrotate` utility should be used to automatically compress and archive the log file after a certain amount of time or after the file reaches a certain size. A standard logrotate configuration for XPS, located at `/etc/logrotate.d/xenon-policy-server`, contains:

```
/var/log/xenon-tools/xenon-policy-server.log {
    daily
    size 100M
    rotate 12
    copytruncate
    compress
    missingok
    notifempty
}
/var/log/xenon-tools/xenon-msm-violations.log {
    daily
    size 10M
    rotate 12
    copytruncate
    compress
    missingok
    notifempty
}
```

With this configuration, log files are checked daily and archived whenever their size is greater than or equal to what is specified. Archived log files are compressed in gzip format to minimize their size. The last twelve log files are kept at a time, and the oldest are deleted to make room for new logs. This policy should be adjusted as needed, depending on requirements.

3.1.3 XPS Authentication and Authorization

XPS communicates with XPGUI via SSL sockets. This mechanism provides authentication for both XPS and XPGUI, and ensures that the traffic between them is encrypted. For this to work, XPS and XPGUI must each be configured with a keystore that contains their unique public and private keys. In order to authenticate each other, they present an X.509 certificate containing their public key to the other party and use the other party's public key to encrypt the traffic that they send. Then, each party is able to decrypt the traffic that they receive using their private key.

The following sections describe the process to get all of the required components (e.g., trust stores, personal keystores, and authorization keystore) set up correctly.

3.1.4 Certificate Authority

A certificate authority (CA) is an entity that issues digital certificates. XPS must have a valid X.509 certificate so that it can establish trust and communicate with XPGUI. Likewise, XPGUI must have a valid X.509 certificate. Different deployments of Xenon may have different requirements and procedures that specify how these digital certificates should be obtained.
In a development environment, it may be desirable to create a local CA and use that to issue certificates because it is free and fast. To create a CA on Ubuntu 16.04 using OpenSSL:

Prepare CA directory structure

```
user@ubuntu:~$ mkdir -p ./demoCA/{newcerts,reqs,private,jks,pkcs12}
user@ubuntu:~$ touch ./demoCA/index.txt
user@ubuntu:~$ echo "01" >> ./demoCA/serial
```

Generate CA key pair and X.509 certificate

```
user@ubuntu:~$ openssl req -x509 -days 365 -newkey rsa:2048 -keyout
   ./demoCA/private/cakey.pem -out ./demoCA/cacert.pem
```

The second command will prompt for a password for the new key pair, and then for some information that is included in the certificate to identify the CA.

3.1.5 Trust Store

If the certificates for a Xenon deployment are issued by a well-known CA (e.g., VeriSign, Comodo, Symantec) then the trust stores are most likely set up correctly. Likewise, if DoD-issued certificates are to be used, then the server and clients may already be configured to trust the DoD root certificates. However, if a local CA is being used for development purposes (or, in general, if a system's trust store is not set up to trust the certificates that will be used), the following steps outline the procedure for adding a CA's certificate to a system's trust store. This must be done on both the Xenon host and remote client hosts.

On Ubuntu 16.04 (replace /path/to/cacert.pem with the path to the CA certificate):

```
user@ubuntu:~$ sudo mkdir -p /usr/share/ca-certificates/xenon
user@ubuntu:~$ sudo cp /path/to/cacert.pem /usr/share/ca-certificates/xenon/xenon-
   ca.crt
user@ubuntu:~$ sudo dpkg-reconfigure ca-certificates
```

On Mac OS X (replace /path/to/cacert.pem with the path to the CA certificate; default password for Java trust store is "changeit"):

```
osx:~ user$ cd $(/usr/libexec/java_home)/jre/lib/security/
osx:~ user$ sudo cp cacerts cacerts.orig
osx:~ user$ sudo keytool -importcert -file /path/to/cacert.pem -alias xenon-ca -
   keystore cacerts
```

On Windows 8 Run the Command Prompt as administrator:

Run the following commands, but replace path\to\jre with the path to the Java JRE or JDK directory, and replace path\to\cacert.pem with the path to the CA certificate (default password for Java trust store is "changeit"):

```
C:\WINDOWS\system32> cd "C:\path\to\jre\lib\security"
C:\path\to\jre\lib\security> copy cacerts cacerts.orig
C:\path\to\jre\lib\security> ..\..\bin\keytool.exe -importcert -file
   "C:\path\to\cacert.pem" -alias xenon-ca -keystore cacerts
```

3.1.6 Public Key, Private Key, X.509 Certificate, and Personal Keystore

The development CA created in SECTION 3.1.4 can be used to issue X.509 certificates for the Xenon server and clients. First, generate the public and private key pair along with a Certificate Signing Request (CSR):

```
user@ubuntu:~$ openssl req -newkey rsa:1024 -keyout ./demoCA/private/xps-key.pem -
   out ./demoCA/reqs/xps-req.pem
```

Then, use the CA to generate a signed X.509 certificate from the CSR:

```
user@ubuntu:~$ openssl ca -in ./demoCA/reqs/xps-req.pem -days 3650 -out
   ./demoCA/newcerts/xps-cert.pem -notext
```

Use the same procedure to generate a key pair and signed certificate for each client.
When launching XPS or a client, the Java Virtual Machine (JVM) must be told where to locate the keystore that contains the certificate and key pair that will be used for SSL connections. To generate a PKCS#12 container from an X.509 certificate and private key, run the following command:

```
user@ubuntu:~$ openssl pkcs12 -export -in ./demoCA/newcerts/xps-cert.pem -inkey
   ./demoCA/private/xps-key.pem -out ./demoCA/pkcs12/xps-certkey.p12
```

Note that this could also be in Java keystore format (.jks extension). Both PKCS#12 containers and Java keystores are able to store X.509 certificates (containing the public key) along with the private key. The JVM arguments to point to the keystore are (replace /path/to/keystore.p12 with the path to the keystore on the system):

```
-Djavax.net.ssl.keyStore=/path/to/keystore.p12 -Djavax.net.ssl.keyStoreType=pkcs12
   -Djavax.net.ssl.keyStorePassword=password
```

For an XPS instance, these arguments are passed via the JAVA_OPTS parameter in the configuration file /etc/xenon-tools/xenon-policy-server.properties.

3.1.7 Authorization Store

The final piece of the puzzle is telling XPS which clients are authorized to establish connections and make requests. Connecting XPGUI locally (from the same host as XPS) is always authorized; as long as XPGUI is configured with a valid X.509 certificate, XPS will accept the connection. Connecting XPGUI remotely (e.g., from a laptop), however, additionally requires XPGUI's certificate to be added to XPS's authorization store.

To create an (initially empty) authorization store for XPS, run the following commands:

```
user@ubuntu:~$ keytool -genkey -alias foo -keystore authstore.jks
user@ubuntu:~$ keytool -delete -alias foo -keystore authstore.jks
```

To tell XPS to load this authorization store, set the --authstore-file option in the GENERAL_OPTS parameter in the configuration file /etc/default/xenon-policy-server. To allow authorized clients to add or remove authorizations from the server's authorization store, also set the --authstore-pass option (this may be useful during initial setup when the authorization store is empty: a local client can connect and add authorization for remote client certificates).
To manually add a client certificate to the authorization store, run the following command (replace /path/to/client-cert.pem with the client certificate, client-alias with a unique identifier for that certificate, and /path/to/authstore.jks with the path to the authorization store):

```
user@ubuntu:~$ keytool -importcert -file /path/to/client-cert.pem -alias client-
     alias -keystore /path/to/authstore.jks
```

To manually remove a client certificate from the authorization store, run the following command (replace client-alias with the unique identifier for the certificate to be removed and /path/to/authstore.jks with the path to the authorization store):

```
user@ubuntu:~$ keytool -delete -alias client-alias -keystore /path/to/authstore.jks
```

3.2 XPGUI

The following is a list of XPGUI features:

- XPGUI can be used to manage a local Xenon host and/or one or more remote Xenon hosts
- XPGUI has an intuitive graphical enclave designer, which allows hardware and VMs to be clearly separated
- By default, rules are put in place to prevent VMs in different enclaves from communicating with each other, either through the hypervisor or through the network IP layer
- XPGUI has an integrated VNC client, which can be used to connect a graphical display to VMs which have VNC activated in their configuration
- XPGUI ships with recommended Xen XL configuration templates for each Xen guest type (PVH, PV, PVHVM, HVM), which means that guests will work correctly out-of-the-box, and are by default set up with the most secure configuration options required to get a particular guest type up and running
- XPGUI has an alert system in place to report any policy violations
- XPGUI provides tools to address legitimate policy violations
- XPGUI can request resource utilization statistics from XPS and displays charts for current and historical performance

- XPGUI allows a Xenon security policy to be exported as a signed XML document, which enables quick dissemination to other hosts
- XPGUI can export the current policy as a template, which generalizes mappings to specific hardware
- XPGUI can import an existing policy template by re-assigning hardware mapping as appropriate on the target hosts

To launch XPGUI, there is an application shortcut at `/usr/share/applications/xenon-policy-gui.desktop` which calls a script at `/usr/local/sbin/xenon-policy-gui`. This script reads configuration parameters from `/etc/default/xenon-policy-gui`. A standard configuration file contains:

```
XENON_JAVA_OPTS
XENON_JAR_FILE
XENON_PROPS_FILE
[...]
```

The JAR_FILE parameter points to the XPGUI binary that ships with Xenon, which is the reference client implementation. When running `xenon-policy-gui`, an additional properties file is loaded from the location specified by the `xenon.policy.gui.path.propsFile` property. This property is set in `xenon-policy.properties`, and by default points to `/etc/xenon-tools/xenon-policy-gui.properties`.

On Ubuntu Desktop, the GUI can be launched by running this script from the command line or via the application shortcut by searching for Xenon Policy in the Applications menu. If dom0 is running Ubuntu Server, however, it is suggested to launch XPGUI remotely, or to invoke it from a host with a graphical desktop via an SSH connection with X11-forwading enabled. For example:

```
user@ubuntu-desktop:~$ ssh -X user@ubuntu-server # where 'ubuntu-server' is Dom0
user@ubuntu-server:~$ /usr/sbin/xenon-policy-gui &
```

XPGUI initially opens the *Xenon Policy GUI* window. In the left pane, there is a container for *Xenon servers*. Right-click on localhost to connect to server. The text changes to green once the connection is established.

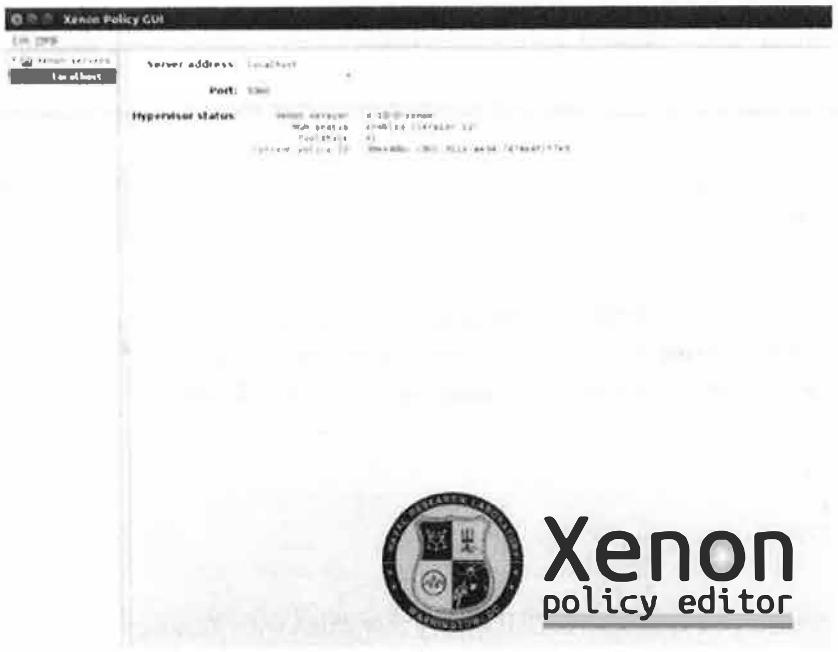
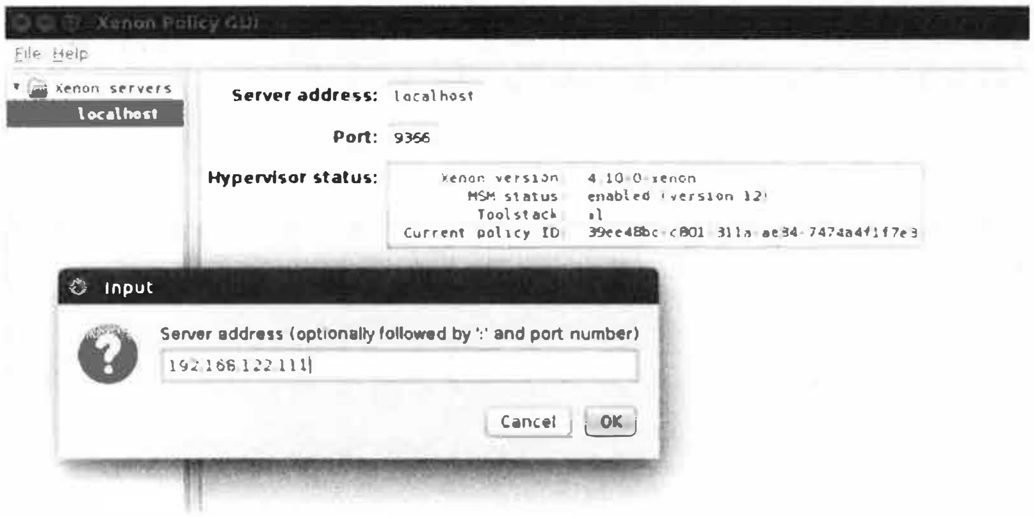


Figure 2 - Xenon Policy GUI

To add a remote server, right-click on *Xenon servers*, select *Add a server...*, and type the server's host name or IP address [FIGURE 3].



Server address: localhost

Port: 9366

Hypervisor status:

```
    Xenon version   4.10.0-xenon
       MSM status   enabled (version 12)
        Toolstack   xl
Current policy ID   39ee4Bbc-c801-311a-ae34-7474a4f1f7e3
```

Server address (optionally followed by ':' and port number)

192.168.122.111

Figure 3 - Add a Server

It is possible to assign an arbitrary color to a server by right-clicking and selecting *Assign color > This server....* As a convenience, there are other menu items in the "Assign color" submenu that allow comparing the policy on a given server to all other connected servers and assigning the color at once.

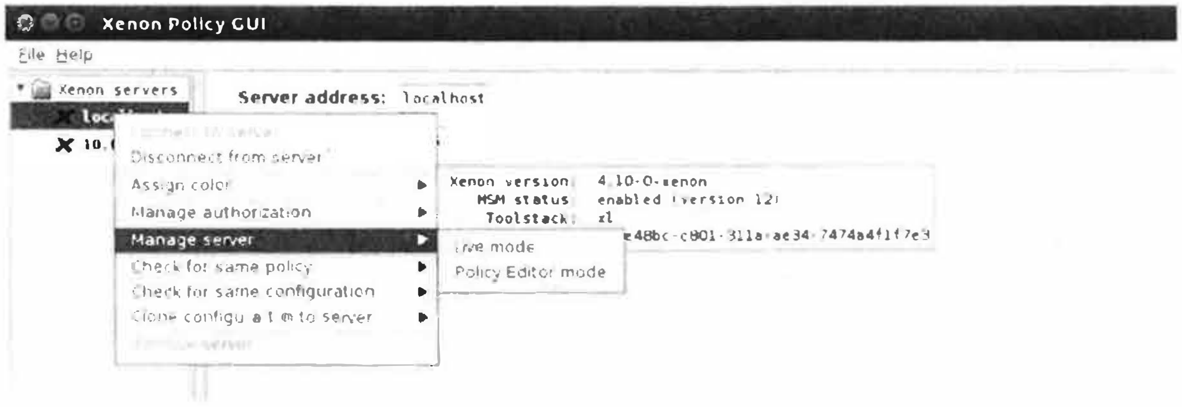
To ease the burden of authoring similar or identical Xenon security policies on many different servers, it is possible to clone a policy from one server to another with a right-click on the source server and select the destination server from the *Clone configuration to server* submenu. Connecting to a server running on the local host only requires the client to be configured with a personal keystore containing their X.509

certificate and private key; the server will accept the connection, even if its authorization store is empty. However, for remote access, a client will need its X.509 certificate to be added to the server's authorization store. The procedure for doing this manually on the server is described in SECTION 3.1.7. However, any authorized user can modify the authorization store via the XPGUI window by right-clicking on a connected server and using the *Manage authorization* submenu.

XPGUI has two modes and they can be identified visually by the color band on the bottom of the tool. Policy editor mode has a red color band and Live mode has a green color band. Each mode has a different set of menu tabs on the main display. Policy editor mode provides editing tools for configuring components in the Xenon security policy. Live mode offers a snapshot of the active Xenon security policy and monitors any security violations. It also manages domain activities such as starting them up or shutting them down.

3.2.1 Policy Editor Mode

XPGUI's editor mode can be launched by right-clicking on any connected server in the XPGUI window and selecting *Policy Editor mode* from the *Manage server* menu [FIGURE 4].



```
Xenon version  4.10-0-xenon
   MSM status  enabled (version 12)
    Toolstack  xl
```

Figure 4 - Manage Server in Dual Mode

3.2.1.1 Policy Menu Tab

The policy editor main display is located under the *policy* menu tab [FIGURE 5]. It contains three panels: *Unassigned Hardware*, *Domains*, *SecLabels, & Enclaves*, and *Enclaves*. Located on the upper left hand corner, the *Unassigned Hardware* panel provides a collapsible tree of host's hardware resources that are available. On the bottom left hand corner, the *Domains, SecLabels , & Enclaves* panel displays a collapsible tree of all the objects currently enforced by the active security policy. This includes network switches, storage allocations, and domains. The *Enclaves* panel displays similar information but in a structural view showing domains in their enclaves and the resources they use. Enclaves in the context of security policy have no policy-related significance except for the tags that are assigned to them when they are created. The main function of enclaves is to group domains and hardware resources with same tag inside one virtual container so that enforcing a strong separation is straightforward and easy to verify.

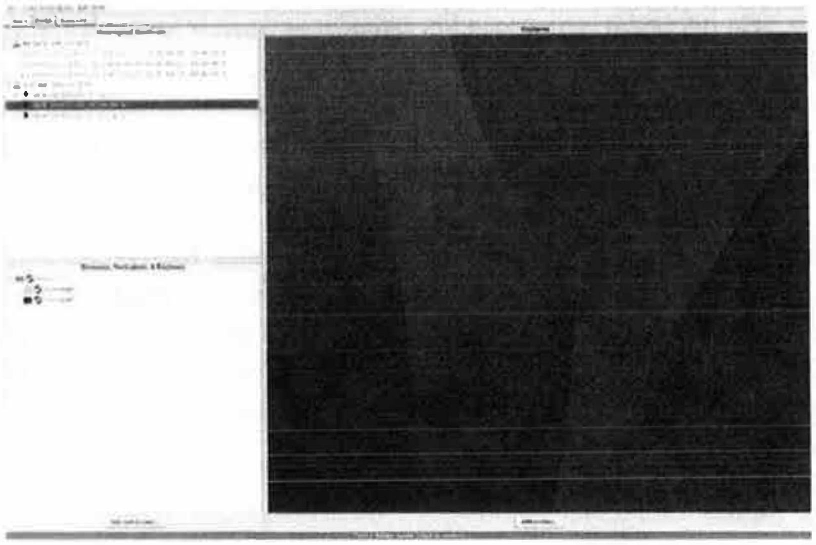


Figure 5 - Policy Editor Main Display

3.2.1.2 Advanced Menu Tab

The *Advanced* menu tab offers a set of submenu tabs that facilitate users with policy related tasks. The tabs are *Hypercall Profiles*, *Tags*, *Conflicts*, *SecLabels*, *Hardware*, *Network Switches and Storage Locations*.

By default, XPGUI comes preloaded with some hypercall profiles that can be assigned to domains running common operating systems like Ubuntu, Fedora/RHEL/CentOS and Windows. The *Hypercall Profiles* tab allows users to modify the existing profiles, import saved profiles, and remove or create new ones [FIGURE 6]. All unprivileged domains must be assigned a hypercall profile.

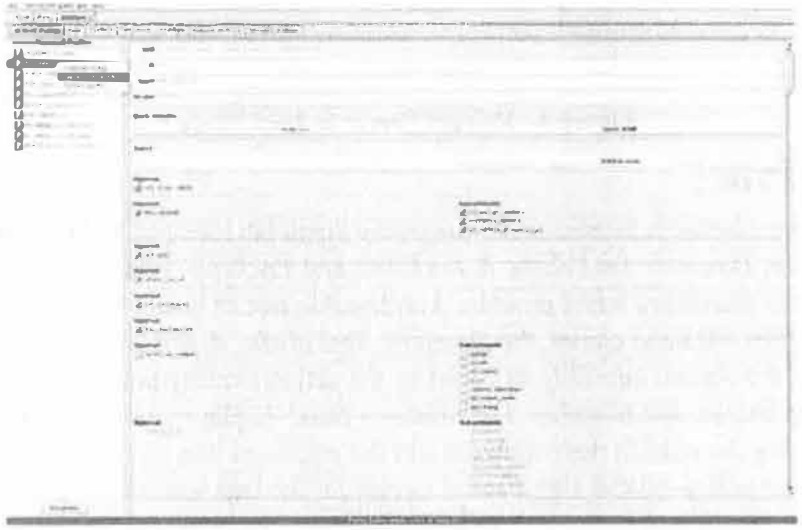


Figure 6 - Hypercall Profiles under Advanced Tab

Tags are presented differently depending on the context. On XPGUI, they are visually displayed in the color of their enclave. On the Xenon security policy, they are represented by their UUID (a unique 128-bit identifier) [FIGURE 7].

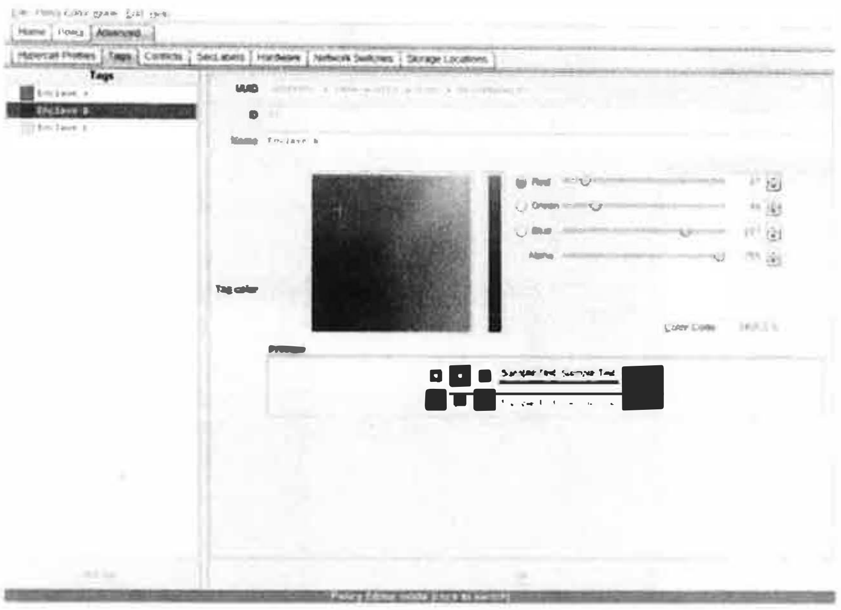


Figure 7 - Tags under Advanced Tab

Likewise, each conflict is assigned a unique UUID within the Xenon Security Policy. By default, domains from different enclaves are allowed to run simultaneously on a single Xenon host, unless a new conflict set is introduced that restricts it [FIGURE 8].

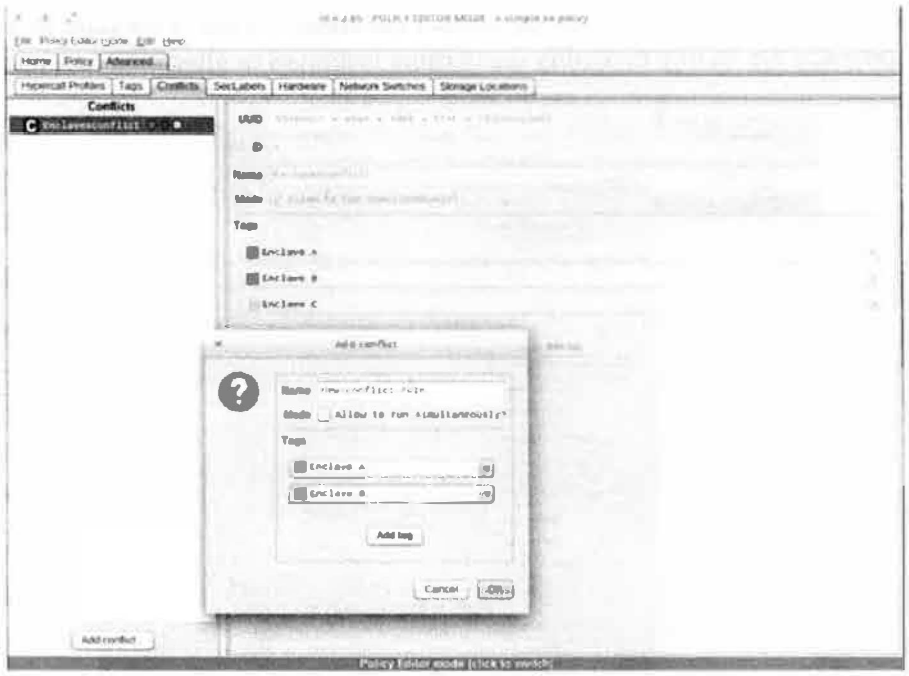


Figure 8 - Create a Conflict Set

All security labels in the current policy can be displayed at once by selecting *SecLabels* under the *Advanced* tab. *SecLabels* provides the facility to add new security labels, and it allows modification of hypercall rate and hypercall profile of the existing ones on the fly [FIGURE 9].

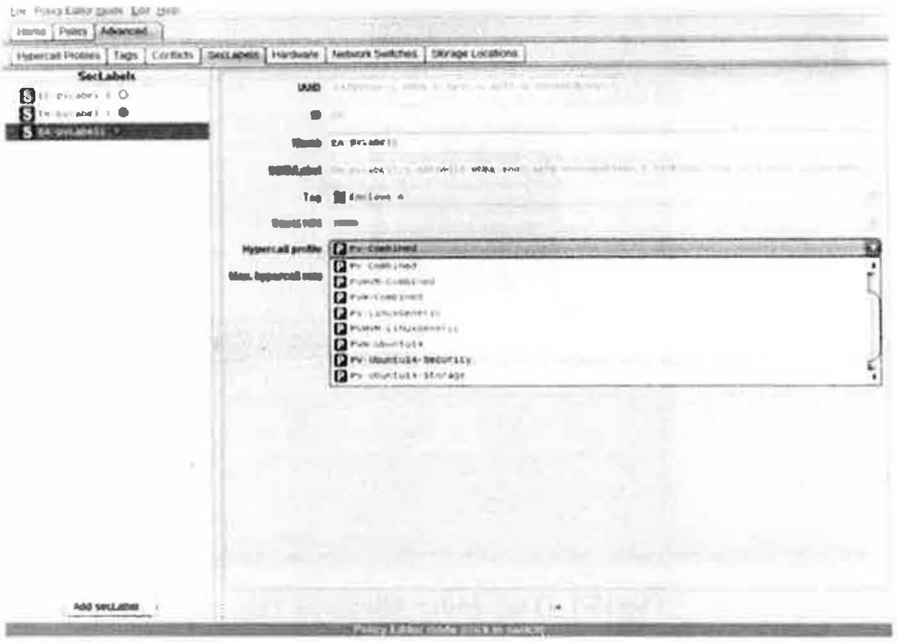


Figure 9 - SecLabels under Advanced Tab

Network Switches and *Storage Locations* tabs share a similar graphical view, which consists of two panels. The left panel displays the hardware resources available to the domains running on the host. The right panel comes with two additional menu tabs. The first tab provides additional information of the highlighted resource on the left [FIGURE 10]. The second tab displays a list of domains that use the resource. It also provides the facility to modify the existing resources or allocating new ones [FIGURE 11].

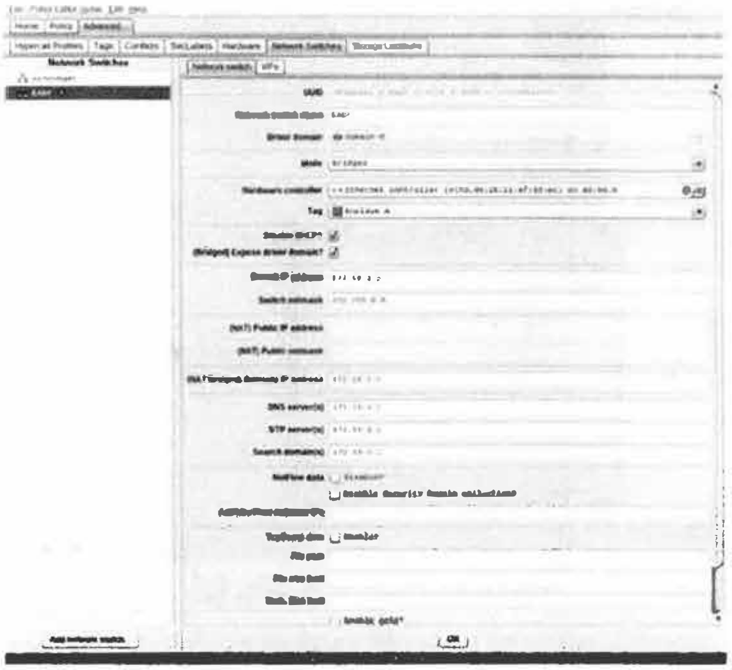


Figure 10 - Network Switch Panel under Advanced Tab

Figure 11 - Domains Serviced by Network Switch, EAbr

3.2.2 Live Mode

Live mode gives a read-only view of the currently running policy. It also provides access to the domains, allows management of their startups and shutdowns, and allows monitoring of policy violations and utilization statistics. Live mode has access to two *read-only* policy menu tabs, *Policy* and *Advanced* that we have discussed previously. In the following sub-sections, we will go over *Policy Violations Log*, *Current Performance* and *Performance History* tabs.

3.2.2.1 Policy Violation Log

XPS has a process in place for monitoring the system for any policy violations as they occur. When a security violation occurs, XPS triggers a warning sign on the *Policy Violations Log* tab to inform the user something out of ordinary has occurred [FIGURE 12]. Violations can be, for example, a domain making a hypercall that is not allowed in its hypercall profile, or it exceeding its predefined max hypercall rate.

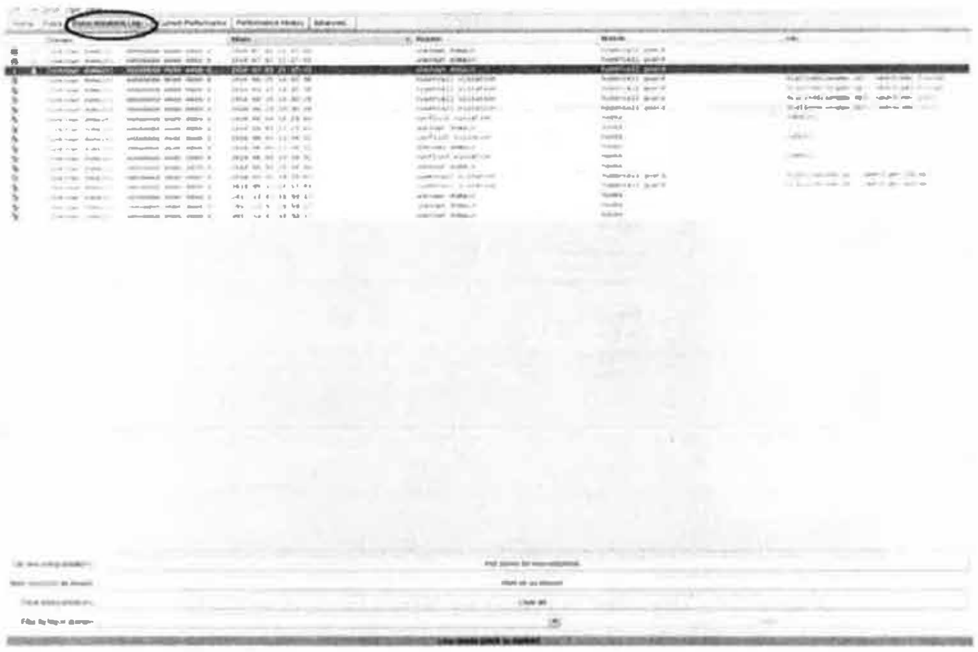


Figure 12 - Policy Violation Log

3.2.2.2 Addressing Policy Violations

The Xenon MSM Security Module will automatically stop any domains that exceed the maximum number of violations, until the violations are addressed. If the violations are caused by legitimate missing hypercalls on the profile, it is possible to update the hypercall profile using the *Hypercall Profiles* tab in the *Policy Editor* mode [FIGURE 6]. The profile will be updated dynamically and takes effect once you make the policy active using *Make this policy & host configuration active....* option under the Policy Editor tool bar.

3.2.2.3 Performance Monitoring

There are two monitoring tabs that provide the user with snapshots of current [FIGURE 13] and historical [FIGURE 14] performance data of the hypervisor and domains that are running. The data includes memory and VCPU utilization that can be filtered by the user.

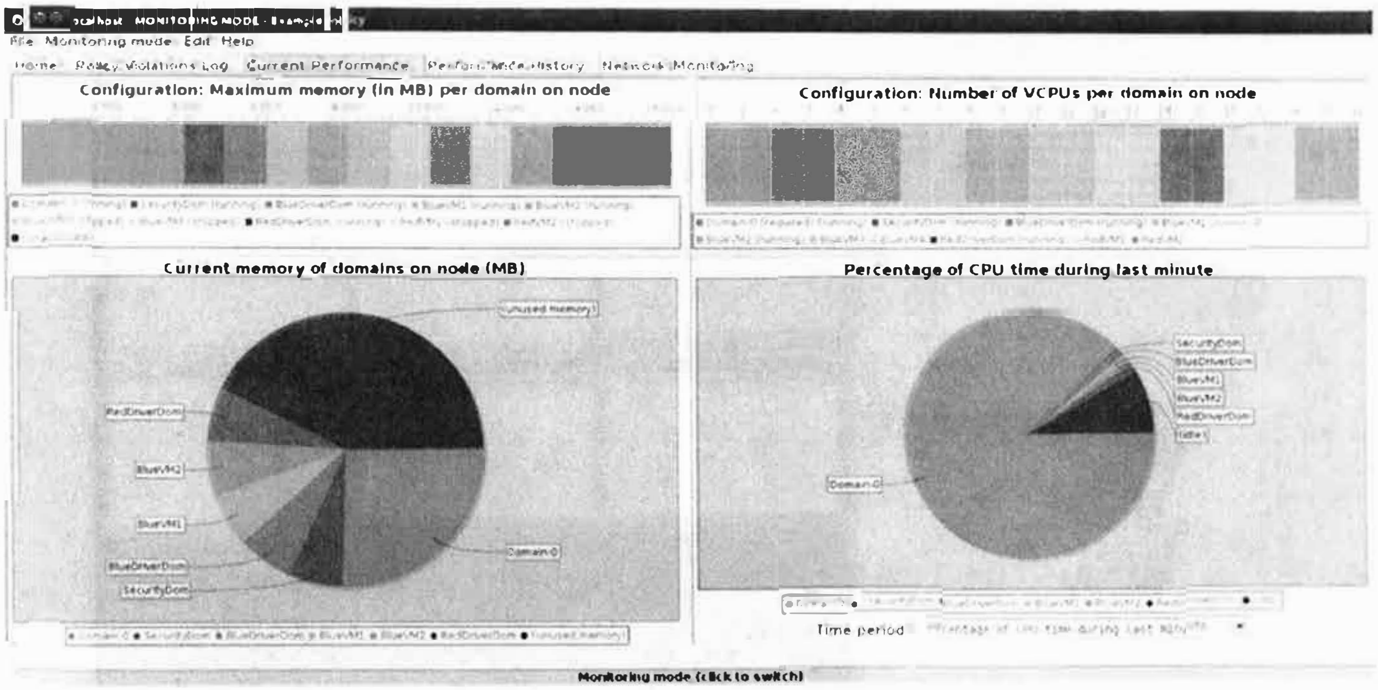

localhost - MONITORING MODE - ExamplePolicy
File Monitoring mode Edit Help
Home Policy Violations Log Current Performance Performance History Network Monitoring
Configuration: Maximum memory (in MB) per domain on node
Configuration: Number of VCPUs per domain on node
Current memory of domains on node (MB)
Percentage of CPU time during last minute
Unused memory
RedDriverDom
BlueVM2
BlueVM1
BlueDriverDom
SecurityDom
Domain-0
Idle
Time period
Monitoring mode (click to switch)

Figure 13 - Current Performance Tab

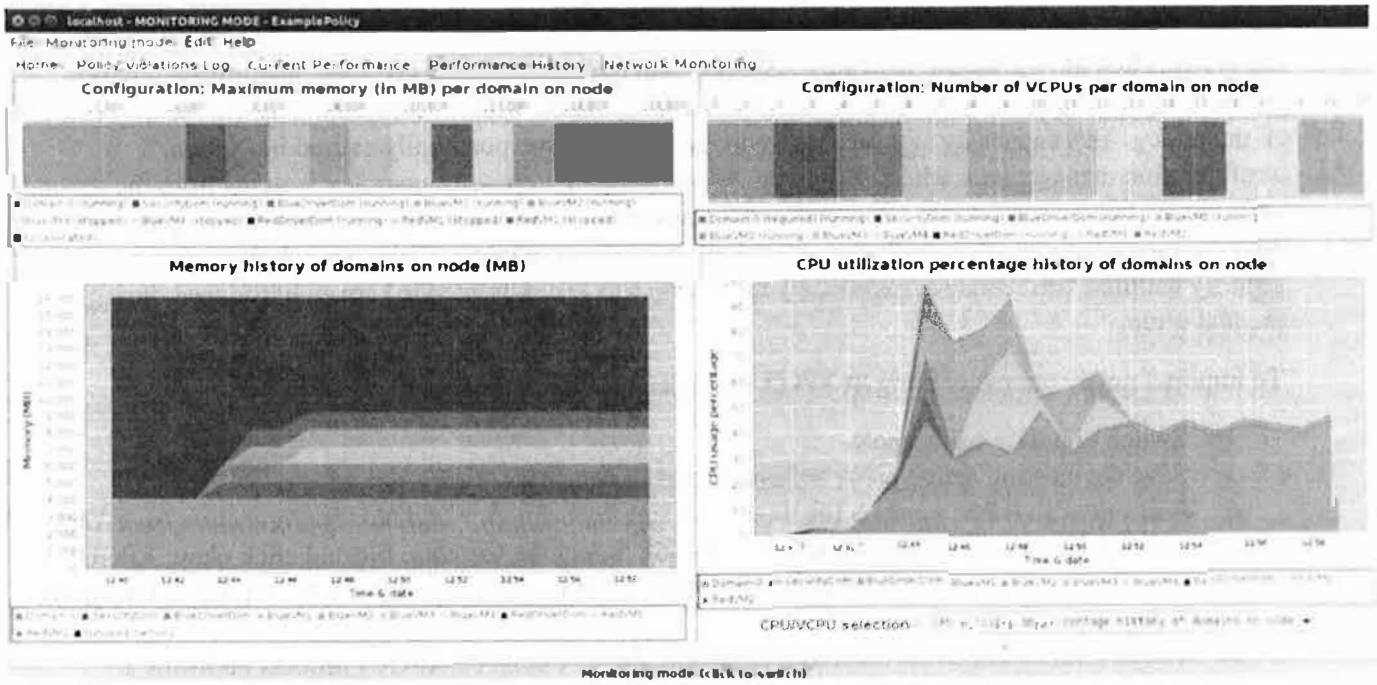

localhost - MONITORING MODE - ExamplePolicy
File Monitoring mode Edit Help
Home Policy Violations Log Current Performance Performance History Network Monitoring
Configuration: Maximum memory (in MB) per domain on node
Configuration: Number of VCPUs per domain on node
Memory history of domains on node (MB)
CPU utilization percentage history of domains on node
Memory (MB)
CPU usage percentage
Time & date
CPU/VCPU selection
Monitoring mode (click to switch)

Figure 14 - Historic Performance Tab

3.2.3 XPGUI Policy Template Import

As seen in FIGURE 3, XPGUI allows management of multiple Xenon hosts. XPGUI has the capability to compare the security policies currently being enforced on two different Xenon hosts. If necessary, XPGUI provides tools for cloning one host configuration (including the security policy) to another host of similar hardware configuration or exporting the configuration of a host as a signed XML document so that other Xenon hosts can import it [FIGURE 15].

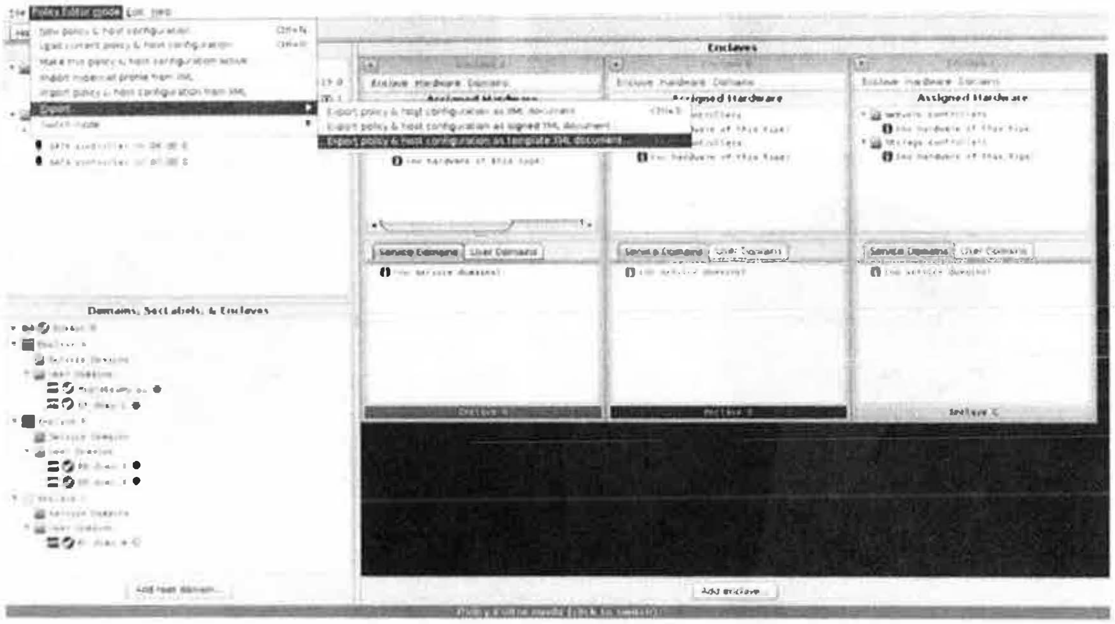

Policy Editor mode
Export
Export policy & host configuration as XML document
Export policy & host configuration as signed XML document
Export policy & host configuration as template XML document...
Enclaves
Assigned Hardware
Service Domains
User Domains
Domains, SecLabels, & Enclaves
Add enclave...

Figure 15 - Policy Export

For systems that have different hardware resources and requirements, XPGUI can disseminate security policy templates to hosts with different hardware configuration without compromising the security rules of the policy. This capability significantly reduces the downtime potentially caused by Xenon configuration management, where setting up Xenon Security Policy for complex systems from the ground up typically takes considerable effort and careful planning. A Xenon policy template can be generated from high-assurance systems with Xenon security mechanism in place. Using a predefined Xenon security template can reduce system deployment time and avoid unintended errors introduced during manual setup.

To import a policy template, bring up XPGUI and perform the following:

- ➢ Switch to policy editor mode.
- ➢ Save the existing active policy before import, if necessary.
- ➢ Select *Policy Editor mode* on the menu bar and *Import policy and host configuration from XML* to bring up the system file dialog [FIGURE 16]. Select the template file and click okay. XPGUI now runs on import mode and an informational dialog is displayed [FIGURE 17]. Click OK on the dialog to proceed.
- ➢ Assign missing hardware resources to the enclaves. A built-in XPGUI process performs a consistency check after each import. Any missing hardware assignment is indicated with a yellow icon and an exclamation mark in it. When the resource is fulfilled, the icon turns green.
- ➢ Click *import* to import the instantiated policy. If import is successful, XPGUI switches back to policy editor mode.
- ➢ Make additional changes on the policy as needed. For example, delete or create new components, reconfigure network and allocate storage for VMs, etc. See SECTION 3.2.1 for more details.
- ➢ To activate policy, select *Policy Editor mode* on the menu bar and *Make this policy & host configuration active...*

➤ Switch to live mode to manage the policy

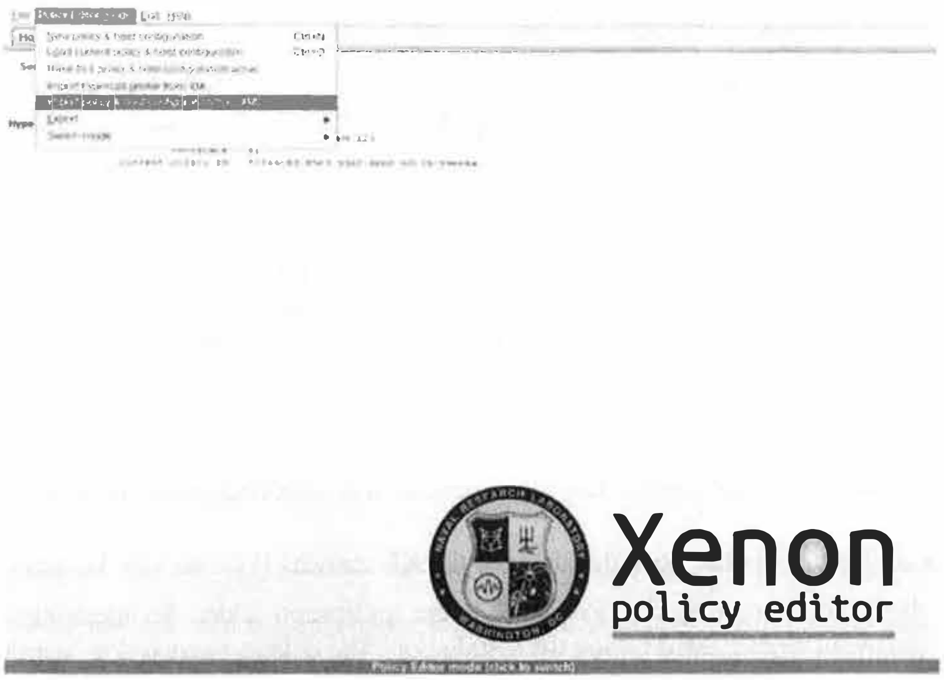


Figure 16 - Policy Import

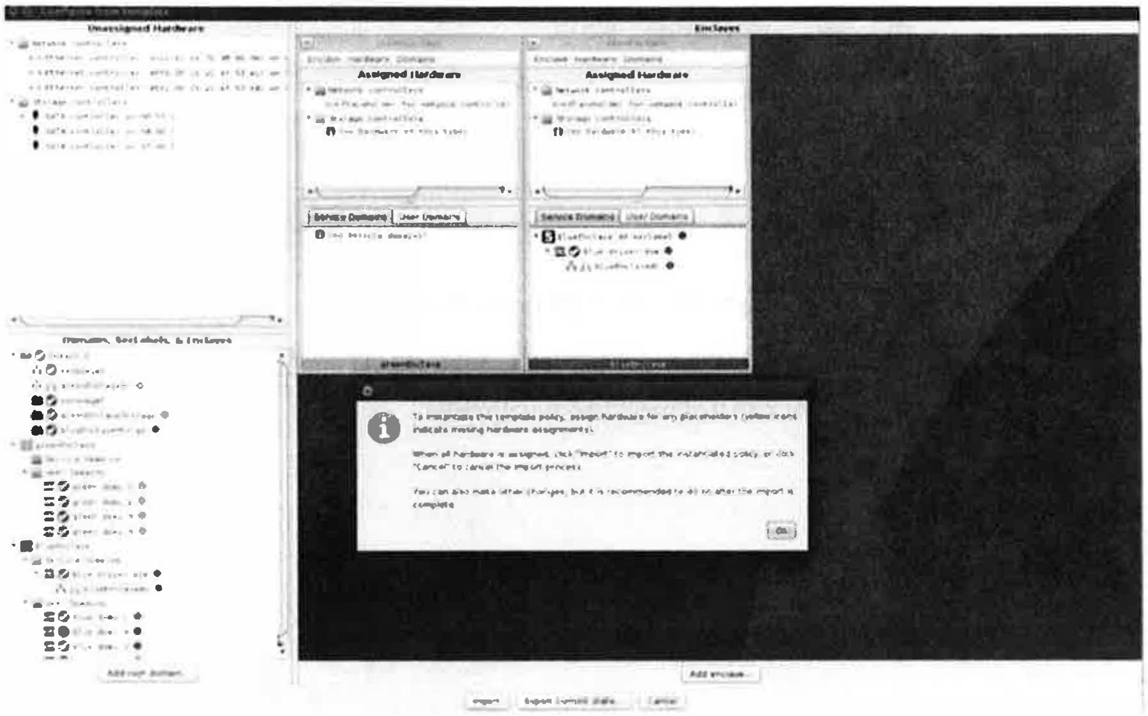


Figure 17 - Policy Editor in Import Mode

4 Xenon Enterprise

Xenon Enterprise (XE) is an enterprise-scale virtualization platform that manages Xenon hosts using XE security policy. XE security policy is an extended version of Xenon security policy that manages resources across multiple Xenon hosts under one federated policy. XE comes with a set of services that allow users to perform policy related tasks and manage the enterprise. These services are accessible using Xenon Enterprise GUI (XEGUI) management tool.

In this section, we first go over the functionality provided by XE services, follow by implementing a simple XE security policy using XEGUI. We conclude the section with discussions on managing host-specific VM resources and VM migration.

4.1 XE Services

The XE services are a set of backend services, exposed via a RESTful API, that handle requests from each other as well as from frontend clients. The reference frontend client implementation is the XEGUI management tool. The three available services are the User Service, the Compute Service and the Policy Service.

4.1.1 User Service

The User Service provides authentication, role management, and authorization of XE users.

XEGUI collects credentials in order to authenticate to the XE services [FIGURE 18]. To access the XEGUI portal, click on the Xenon Enterprise GUI icon (■) in the application folder. An administrative user account and its password were created during the installation. The default password is "adminpass".

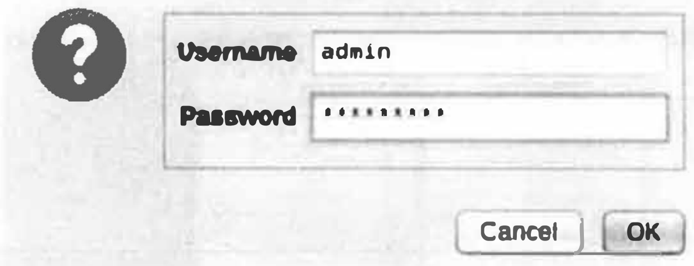

Username: admin
Password: ••••••••••

Figure 18 - XEGUI Login Portal

Upon successful login, click on *Users* on the menu bar (or use the Alt+U mnemonic) to access its toolset [FIGURE 19]. Create additional users and manage roles if necessary.

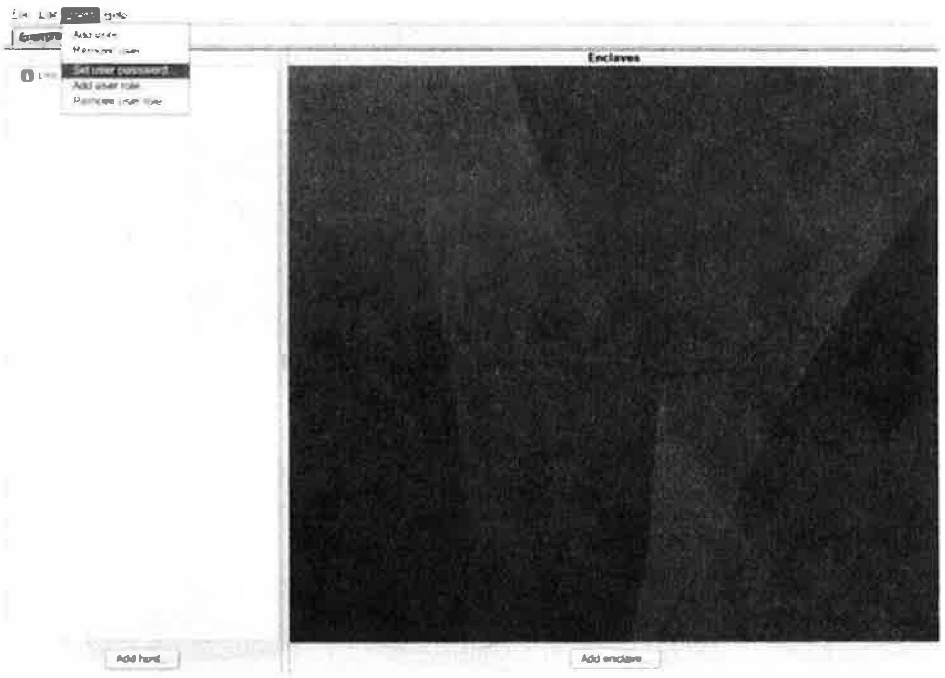


Figure 19 - User Service

4.1.2 Compute Service

The Compute Service provides compute node management such as adding a Xenon host to the enterprise. It also offers a set of tools for managing VM instances running on XE. The tools include adding and removing Xenon hosts from the enterprise; creating, pausing and destroying VMs; and migrating VMs from one Xenon host to another.

4.1.3 Policy Service

The Policy Service provides functionality related to manipulating the XE security policy. This includes configuring and modifying the policy while propagating any policy changes to the Xenon hosts. In the background, the service ensures that the changes are compatible and persistent across the enterprise using a built-in consistency checker.

4.2 XE Security Policy

In this section, we will construct a simple XE security policy from the ground up using XEGUI and XE services installed on a local server. This server provides a PostgreSQL database, which persists the XE security components, policy related information, and other data needed by the XE services.

The XE security policy will define a configuration with a set of VMs running on three Xenon hosts. Two assumptions are made for each host:

- ➢ The participating Xenon host has acceptable network connectivity and XPS has been configured with certificates such that it can be reached by XE and added to the enterprise's pool of hosts
- ➢ The host has no security policy enforcement in place prior to joining XE

The following diagram displays the XE configuration. On the frontend, XEGUI displays the three remote Xenon hosts and manages three enclaves and their VMs running on XE. On the backend, XE services and a database are available to perform policy related tasks requested from the frontend.

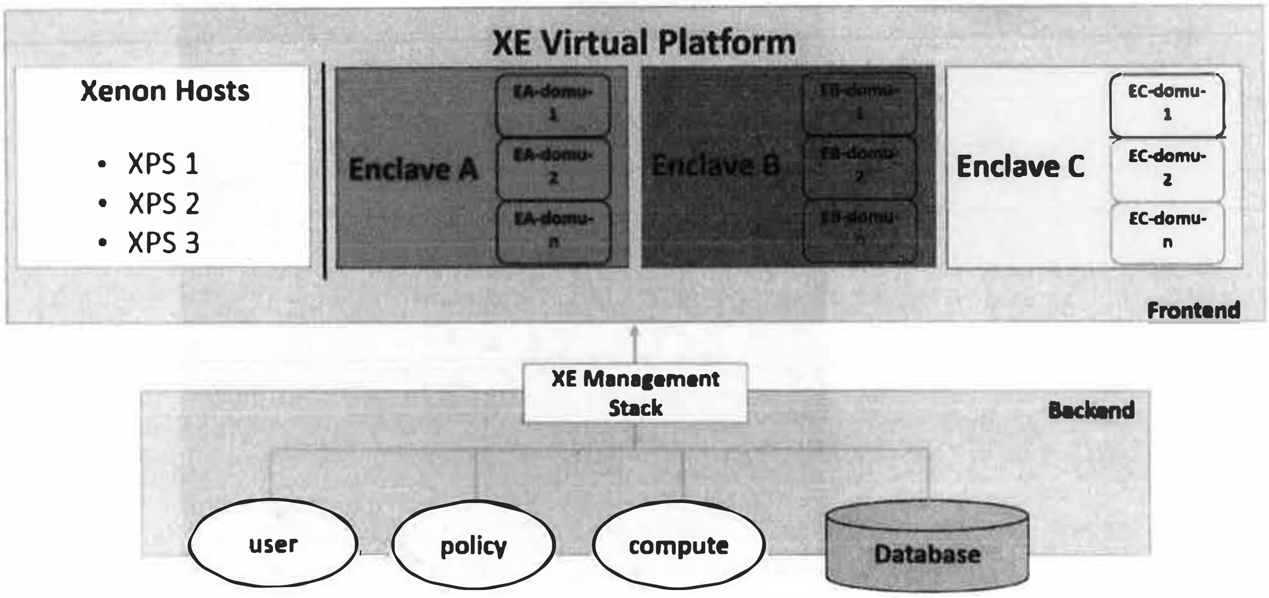


Figure 20 - Accessing XE on Local Desktop

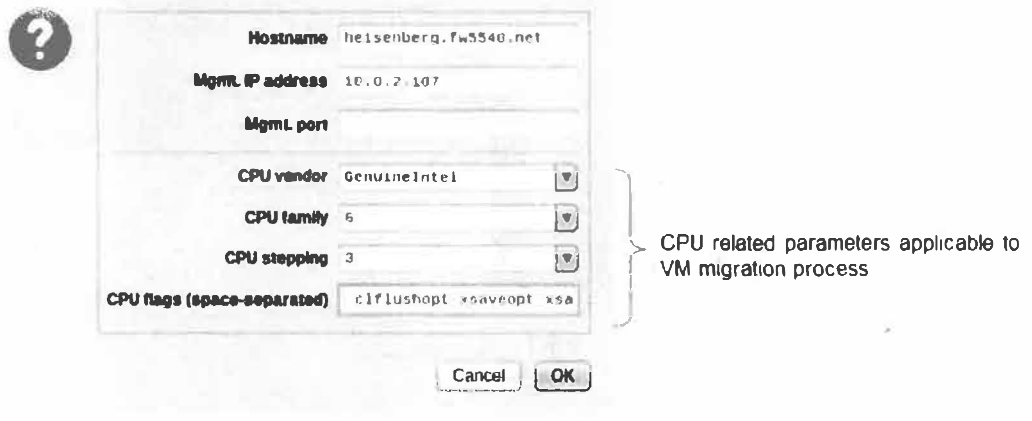
4.2.1 Implementation Rules

The following summarizes a set of guidelines for implementing XE security policies.

- ➢ XE security policy is based on security tags. Enclaves are virtual containers that contain objects of the same tag.
- ➢ A tag can be assigned to one or more Xenon hosts. Thus, an enclave can span multiple hosts.
- ➢ Every VM instance belongs to one designated enclave.
- ➢ Every VM instance must be assigned a SecLabel from the enclave it belongs to.
- ➢ Although enclaves can span multiple Xenon hosts, VMs can only reside and run on one host at the time.
- ➢ The current release of XE does not support services domains running on Xenon hosts.
- ➢ Allocation of hardware resources to VMs (passthrough) is host-specific and must be handled by using XPGUI on the target host.
- ➢ VMs can migrate to a target Xenon host of the same enclave and with compatible CPU flags of the source host.

4.2.2 Add Hosts to Policy

Log on to XEGUI with a predefined user account and password. Add each Xenon host to the policy using the *Add host...* button on the lower left hand corner inside the *Hosts* panel. Fill in *Hostname* and *Mgmt. IP address* on the widget. For hosts that are participating in VM migration, CPU related parameters located on the bottom half of the widget [FIGURE 21] are required. Click OK to save. Once the host has succeeded in joining XE, it will appear inside the *Hosts* panel.



Hostname	heisenberg.fw5540.net
Mgmt. IP address	10.0.2.107
Mgmt. port	
CPU vendor	GenuineIntel
CPU family	6
CPU stepping	3
CPU flags (space-separated)	clflushopt xsaveopt xsa

Figure 21 - Adding Host to Policy

At this point, XE can access each registered host and view its current state and hardware related information. For example, right-click on the host we just added in the *Hosts* panel to bring up a context menu. Choose *Manage Server* → *Editor mode* and enter. A separate panel appears displaying general information about the host. Click on the *Policy* tab on the menu bar to view host's local hardware resources (see FIGURE 5 in SECTION 3.2.1.1). If an existing security policy is in place, save it by exporting it or creating a new policy. These options can be found under *Policy Editor mode* on the menu bar. Save changes by making the policy active before returning to XEGUI.

Repeat the steps above to register two additional Xenon hosts to the policy.

4.2.3 Create Enclaves

Every XE security policy is started by creating a set of enclaves. In XEGUI, add an enclave to the security policy using the *Add enclave*... button on the bottom of the main panel. A widget comes up titled *Add tag* [FIGURE 22]. Every enclave created on the XE platform corresponds to a tag. The XE policy service automatically generates a unique 128-bit UUID that represents the tag within the XE security policy. Provide a *Name* for the enclave and select a *Tag color* by adjusting the RGB values on the widget. The selected color provides a visual indicator of the tag throughout XEGUI.Repeat this step to create two additional enclaves.

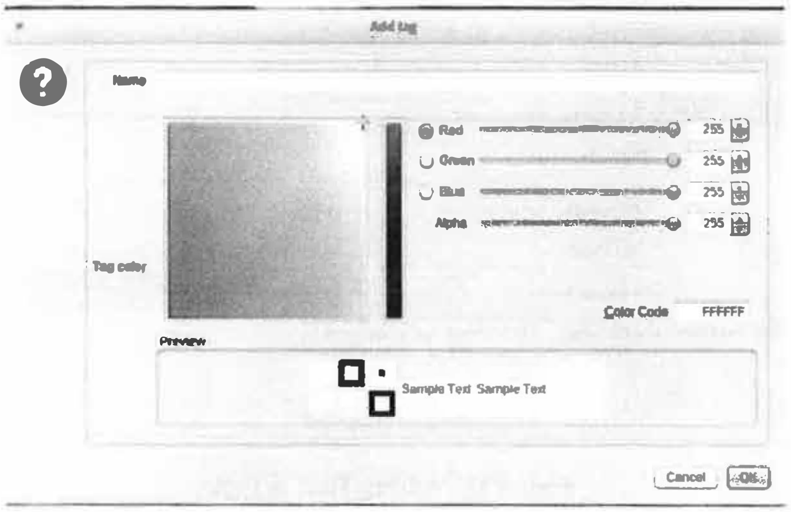


Figure 22 - Creating Enclaves in XEGUI

4.2.4 Assign Enclave to Hosts

Every enclave in the security policy should be assigned to at least one host. For this simple XE security policy, we assign Enclave A to two hosts (to later demonstrate migration of a VM between two hosts), and Enclave B and C to all hosts. One at a time, right-click on each Xenon host to bring up its context menu. Select *Add tag* and choose an enclave name to add. Click *OK* to save. Once the assignment is complete, each host displays its assigned enclaves in a collapsible tree inside the *Hosts* panel. Notice that every enclave on the policy spans more than one Xenon host [FIGURE 23].

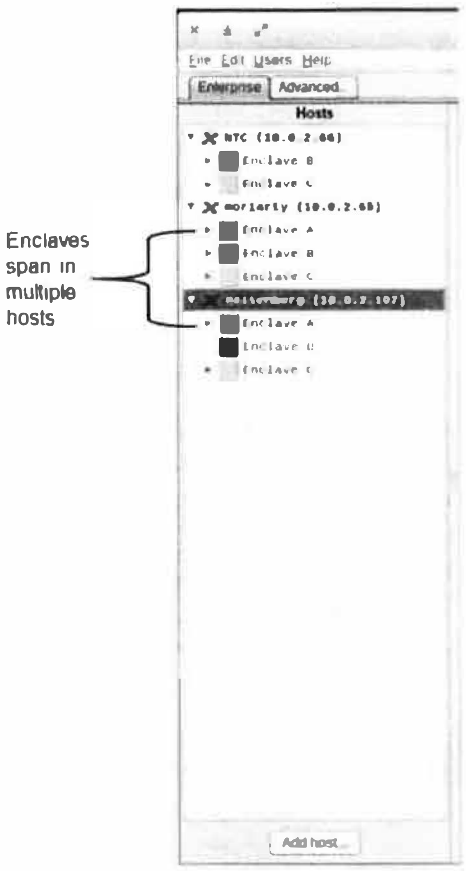


Figure 23 – Hosts Panel Displaying the Host-Enclave Tree View

4.2.5 Create SecLabel and VMs

Next, create VMs using *Domains* on the toolbar inside the enclave panel. As one of the implementation rules in SECTION 4.2.1, every VM must have a SecLabel assigned by the enclave it belongs. Since the policy has yet created any SecLabels, the policy service stops the transaction and issues a warning [FIGURE 24]. This is expected, as the XE policy service performs security and consistency checks before each enterprise transaction takes place.

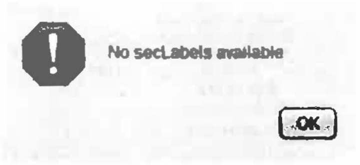
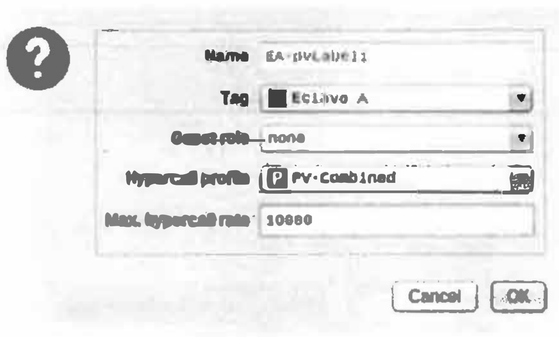

No secLabels available

Figure 24 - Policy Service Warning Dialog

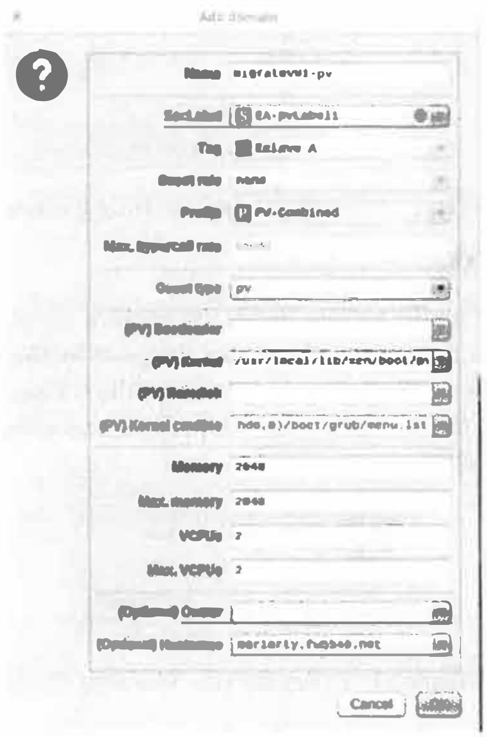
To create a SecLabel inside Enclave A for example, click on *SecLabels* on the enclave toolbar and choose *Add secLabel* → *Create new*.... to bring up the *Add secLabel* widget. Provide *Name*, and *Max. hypercall rate* value (which cannot exceed 65535), but leave *Guest role* as the default "none" value. Use the pulldown menu to choose a hypercall profile that matches the VM type (see SECTION 2.2.2 for domain configuration options). Click *OK* to save [FIGURE 25]. The SecLabel is now available for user domains in Enclave A. Create two additional SecLabels, one for each of remaining enclaves.



Name	EA-pvLabel1
Tag	Enclave A
Guest role	none
Hypercall profile	PV-Combined
Max. hypercall rate	10000

Figure 25 - Add SecLabel Widget

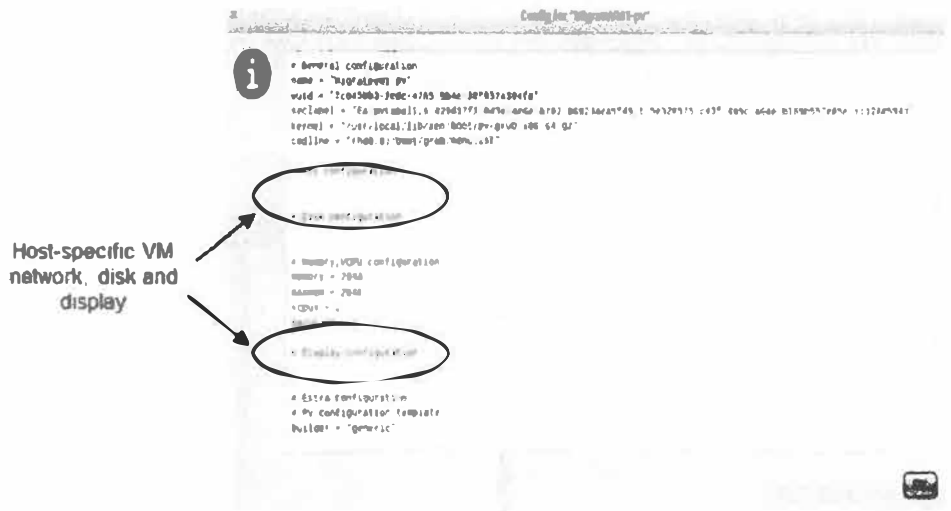
Proceed to create a VM. Click on Domains on the toolbar and choose *Add domain* to access the *Add domain* widget [FIGURE 26]. Provide a domain *Name* and a *Guest type* that matches the SecLabel's hypercall profile selected previously. Other mandatory user input include *Kernel* and *Bootloader* (applicable to PV guests only), *Memory*, *VCPUs*, etc. One notable configuration option is the optional *Hostname* dropdown. If the enclave where the VM belongs spans more than one hosts and the VM's *Hostname* is unspecified, XEGUI creates the VM on the first Xenon host listed in the *Hosts* panel that has been assigned this enclave [FIGURE 23].



Name	migratevm1-pv
SecLabel	EA-pvLabel1
Tag	Enclave A
Guest role	none
Profile	PV-Combined
Max. hypercall rate	[illegible]
Guest type	pv
(PV) Bootloader	
(PV) Kernel	/usr/local/lib/xen/boot/pv
(PV) Ramdisk	
(PV) Kernel cmdline	hd0,0)/boot/grub/menu.lst
Memory	2048
Max. memory	2048
VCPUs	2
Max. VCPUs	2
(Optional) Owner	
(Optional) Hostname	moriarty.fubsec.net

Figure 26 - Add Domain Widget

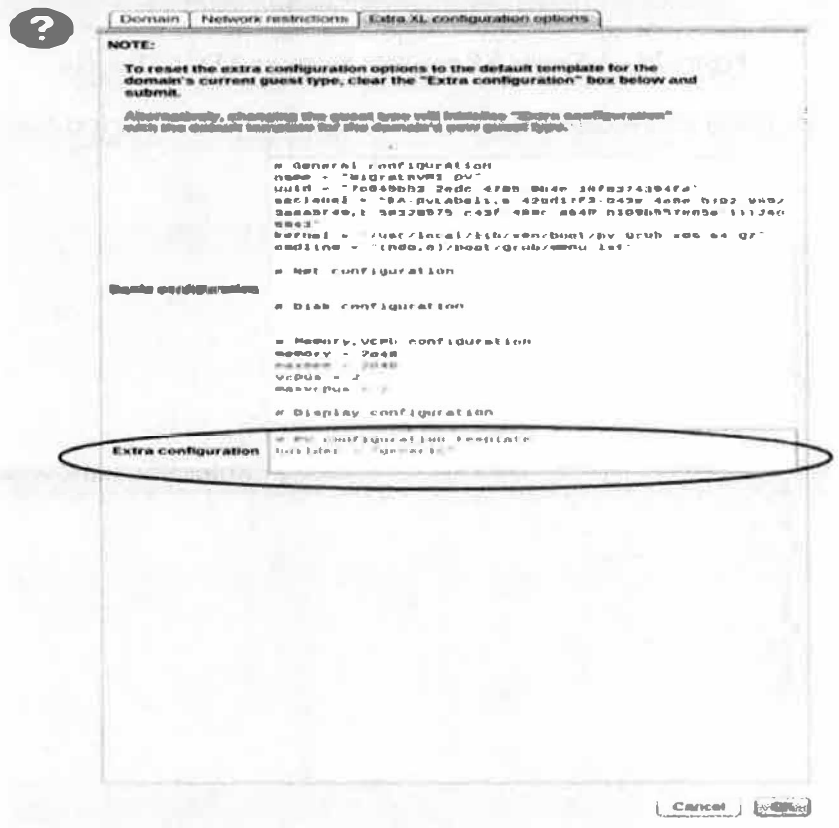
At this point, a VM config file for the VM has been generated. To view it, right-click on the VM to bring up the context menu and choose *View domain config*. Notice host-specific hardware resources for virtualized network, disk storage and display configuration are not configured [FIGURE 27].



```
# General configuration
name = "MigratevM1 pv"
uuid = "[illegible]"
seclabel = "[illegible]"
kernel = "/usr/local/lib/xen/boot/pv-grub x86 64 gz"
cmdline = "(hd0,0)/boot/grub/menu.lst"

# Net configuration

# Disk configuration

# Memory,VCPU configuration
memory = 2048
maxmem = 2048
vcpus = [illegible]
[illegible]

# Display configuration

# Extra configuration
# PV configuration template
builder = "generic"
```

Figure 27 - VM Configuration File on XEGUI

We will discuss host-dependent hardware allocation in SECTION 4.3. Additional configuration specific to the VM is optional using the *Network restrictions* and *Extra XL configuration options* under VM's *Configure domains*... [FIGURE 28].



NOTE:

To reset the extra configuration options to the default template for the domain's current guest type, clear the "Extra configuration" box below and submit.

Alternatively, changing the guest type will initialize "Extra configuration" with the default template for the domain's new guest type.

Basic configuration

```
# General configuration
name = "MigratevM1 pv"
uuid = "[illegible]"
seclabel = "[illegible]"
kernel = "/usr/local/lib/xen/boot/pv grub x86 64 gz"
cmdline = "(hd0,0)/boot/grub/menu lst"

# Net configuration

# Disk configuration

# Memory,VCPU configuration
memory = 2048
maxmem = 2048
vcpus = 2
maxvcpus = 2

# Display configuration
```

Extra configuration

```
# PV configuration template
builder = "generic"
```

Figure 28 - Domain Configuration Options in VM Management Tool

For the simple XE policy, we created additional VMs for all three enclaves and placed them in different hosts throughout the enterprise. For example, *Enclave C* has four VM instances that run on all three hosts on XE. FIGURE 29 displays the policy we constructed on XEGUI.

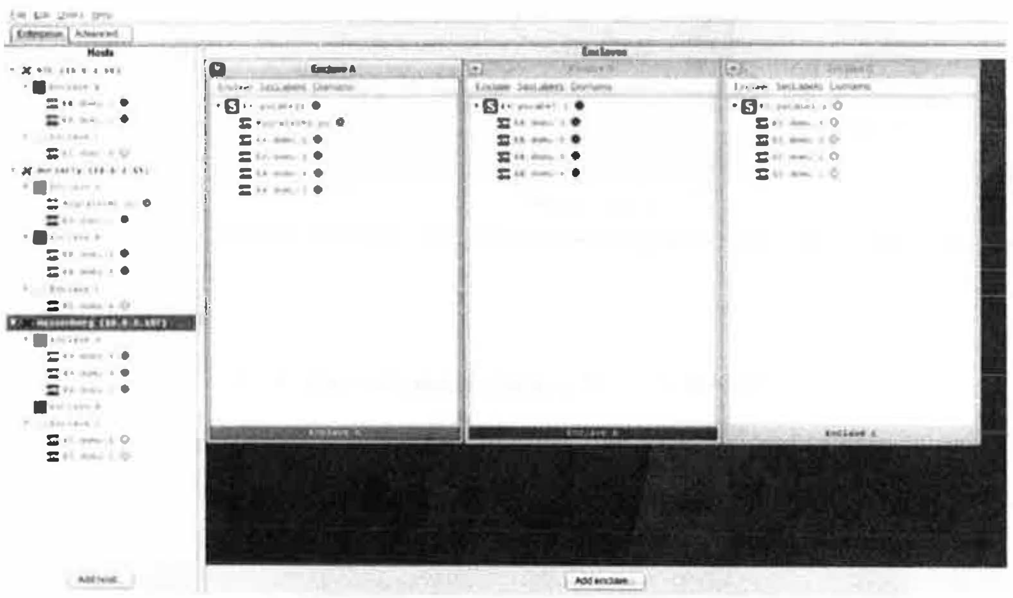


Figure 29 - A Simple XE Security Policy on XEGUI Display

We can also view the policy as a collapsible tree [FIGURE 30] or in a XML or text format.

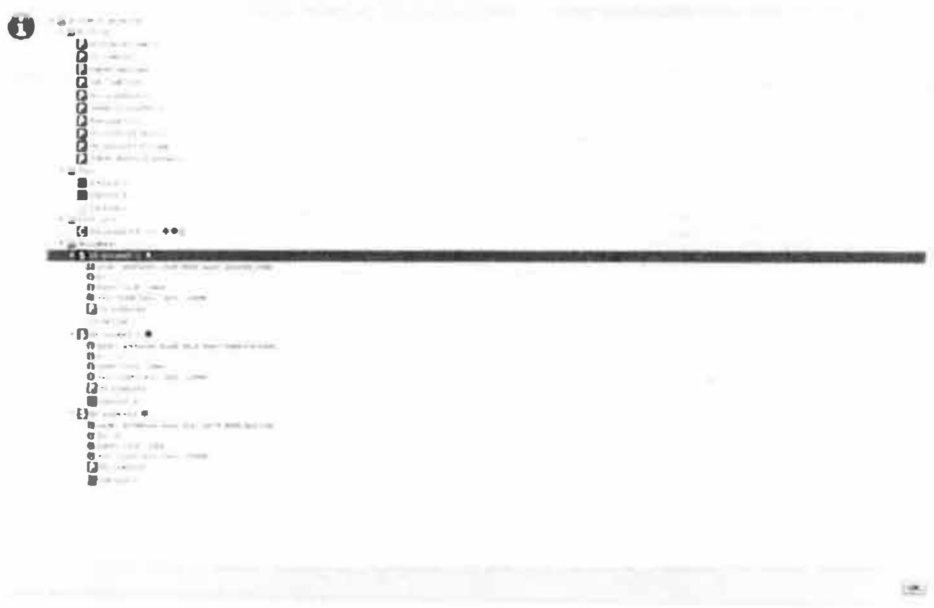


Figure 30 - A Simple XE Security Policy in Tree View

The XE policy service has a built-in process that checks for policy consistency. The process also monitors changes made to the policy in a fixed time interval and propagates them to each participating host. Each XPS running on the remote Xenon host enforces the portion of the simple XE security policy pertinent to its localhost. For example, Host 3 manages three of five VMs in Enclave A, two in Enclave C and none in Enclave B [FIGURE 31].

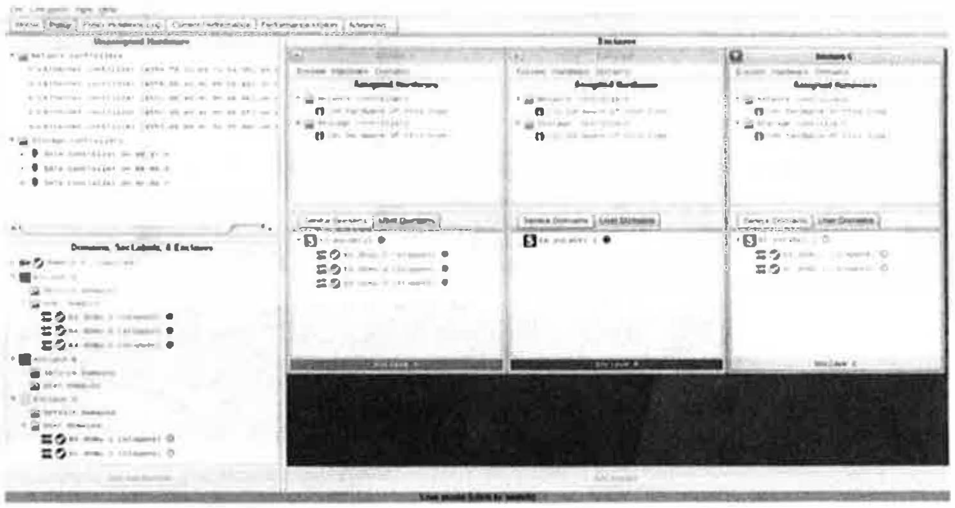


Figure 31 - XE Security Policy Enforcement on the Local Level

4.3 Host-Specific VM Configuration

This section discusses host-specific resource management for VMs running in an enclave that a Xenon host on XE manages. The resources in question are network, disk and display configurations.

As seen in Figure 27, VMs created and managed by the XEGUI does not include services for managing host-specific hardware configuration. This means that each Xenon host must allocate resources for the VMs before they can run. Moreover, these host-specific VM resource allocations will be maintained and preserved on the local level without XE intervention.

In the following two subsections, we present examples of network and storage allocations for the VM instances on the XE security policy we have constructed thus far. Begin by accessing a Xenon host through XEGUI. Right-click on the host and select *Manage server* and then *Editor mode* to connect an instance of XPGUI to this host. Inspect available system resources inside the *Unassigned Hardware* panel on the upper left hand corner. For example, the host in Figure 31 has five unassigned Network controllers and three Storage controllers.

4.3.1 Virtual Network Setup

Standalone Xenon hosts offer virtual network configuration in three different types: bridged, routed, and NAT. However, we strongly advise using *bridged-only* networking for Xenon hosts managed as part of XE. Furthermore, the bridged network shall work with a dedicated DHCP server for routing VM network traffic. This approach provides additional finer-grain network filtering capability that complements the communication policy implemented in the Xenon MSM security module.

Continuing with XPGUI in the policy editor mode, drag and drop an unassigned network controller to Enclave A. Press *OK* when a confirmation dialog comes up. Notice the controller is now assigned the same color tag (red) as the enclave to which it has been added. Next, we configure a bridged network for Enclave A on this host. Right-click on Domain-0 in the *Domains, SecLabels, & Enclaves* panel and choose *Add network switch...* to access the *Add network switch* widget [FIGURE 32].

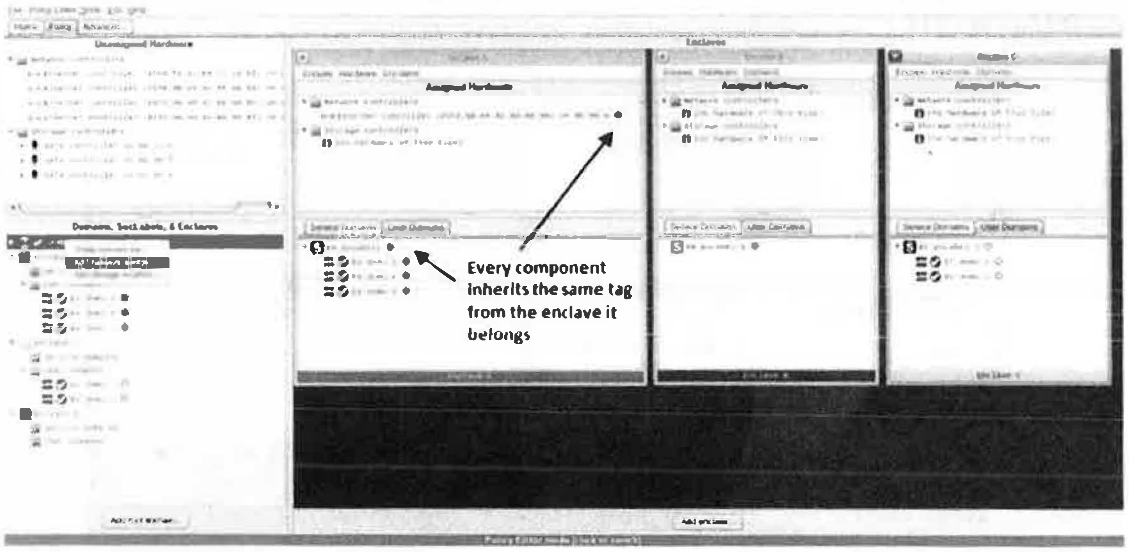

Unmanaged Hardware
Domains, SecLabels, & Enclaves
Enclaves
Assigned Hardware
Every component inherits the same tag from the enclave it belongs

Figure 32 - Adding a Local Network Switch

For a bridged network configuration, we have the following:

```
Network switch name: EAbr
Mode: bridged
Hardware controller: choose controller with same color tag
Tag: choose the same color tag
Disable DHCP? : checked (disable local DHCP and enable the use of the remote DHCP)
(Bridged) Expose driver domain? : checked
Switch IP address: assign random value but in the subnet of the remote DHCP
Switch netmask: 255.x.x.x.
(Nat/Bridged) Gateway IP address: remote DHCP server
DNS server(s): remote DHCP server
Search domain(s): remote DHCP server
```

To view an existing network configuration, right-click on the network switch and select *Configure network switch...* from the context menu [FIGURE 33]. The menu also provides information on the firewall and network routing rules under *Show rules & configuration* menu option.

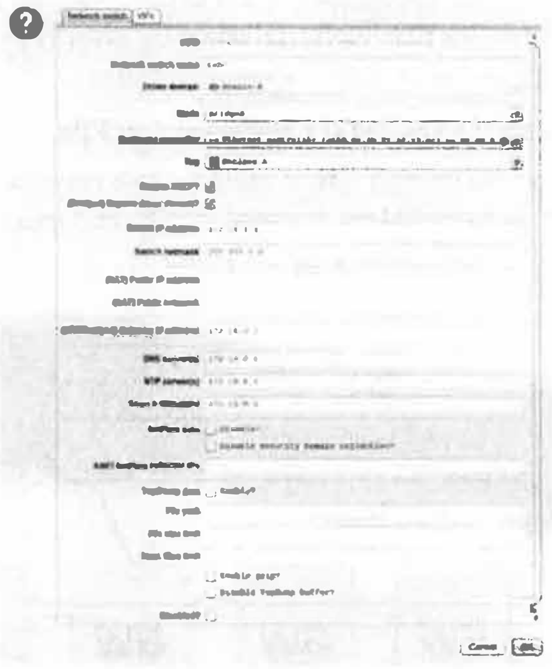


Figure 33 - Network Switch Configuration Panel

Next, with the network switch in place, proceed to allocate a network interface for a VM. Right-click on EAbr in the *Domains, SecLabels,& Enclaves* panel and *choose Configure network switch...* Select the *VIFs* tab and click on *Add VIF...* on the bottom of the panel to access the *Select a domain* dialog. Highlight the VM and press *OK* to continue. Once the *VIFs* dialog appears, enable the *Enable VIF* checkbox and the system will automatically generate a MAC address for the VM. Assign an IP to the VM and press *OK* to exit [FIGURE 34].

Figure 34 - Allocate Network Interface for VM

Another approach to create a VM network interface is to use the *Configure domain* option from the VM's context menu. To save and activate the new configuration on the security policy, select *Policy Editor mode* on the toolbar -> *Make this policy & host configuration active*.

4.3.2 Update Remote DHCP (DNSMASQ) Configuration File

As discussed in previous section, we strongly advise using bridged network with dedicated DHCP server to support VMs running in the enclaves that are managed by XE. This ensures a strong communication policy enforcement among VMs of different tags on the policy.

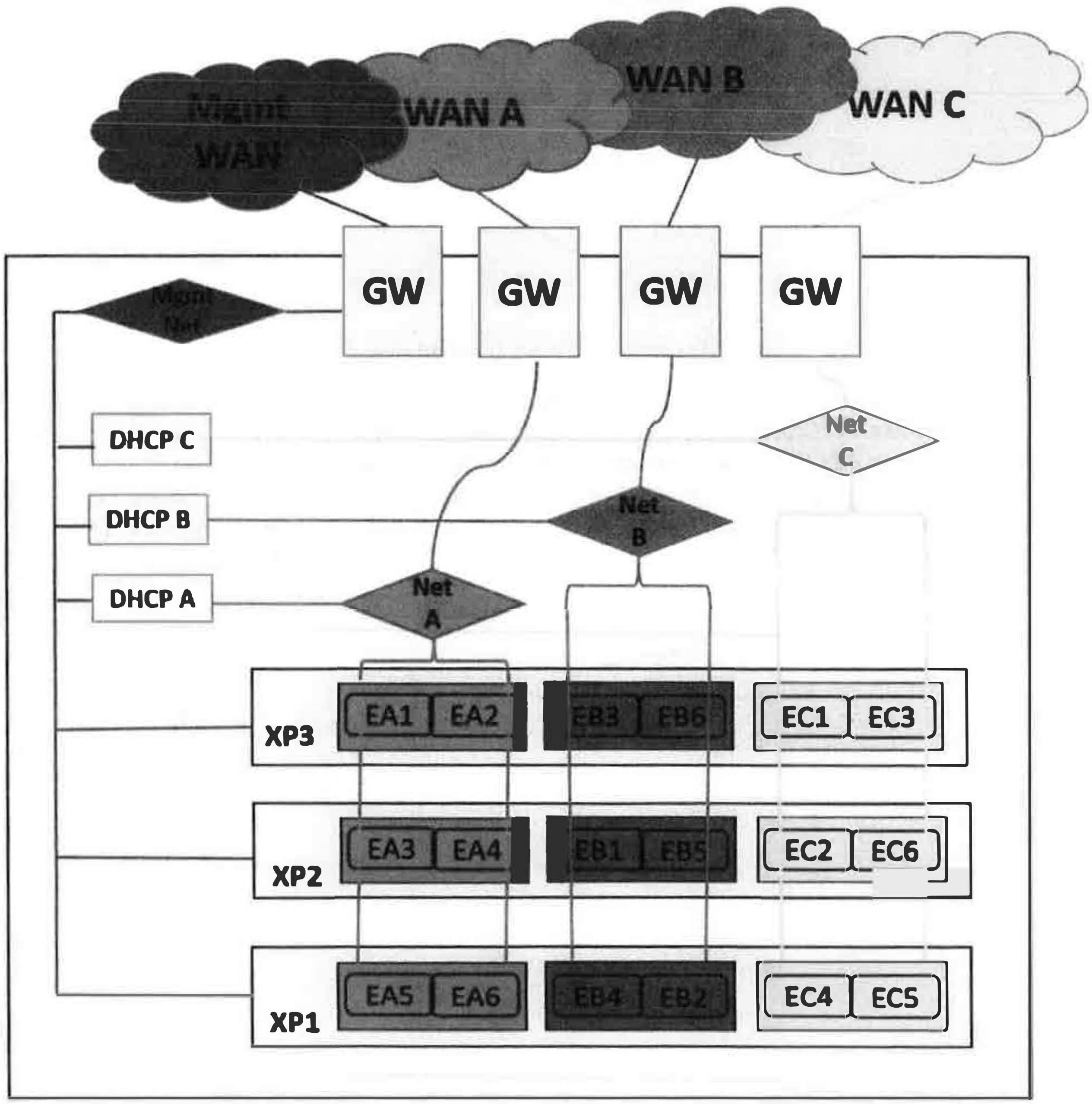


Figure 35 - Network Configuration for VMs on XE Security Policy

By design, every dnsmasq process (local or remote) serving Xenon host needs to keep track of the network related information with a configuration file. Typically, for the local dnsmasq, XPS updates any changes made to the file automatically. This file, named xenon-policy, is stored under the /etc/dnsmasq.d/ directory and looks something like this:

```
# xenon-policy dnsmasq configuration (auto generated on xps)
dhcp-ignore=tag:!known

# Configuration for xenonmgmt
# xenonmgmt: untagged
interface=xenonmgmt
dhcp-range=xenonmgmt,tag:untagged,172.31.255.1,static,255.255.255.0
dhcp-option=xenonmgmt,tag:untagged,option:netmask,255.255.255.0
```

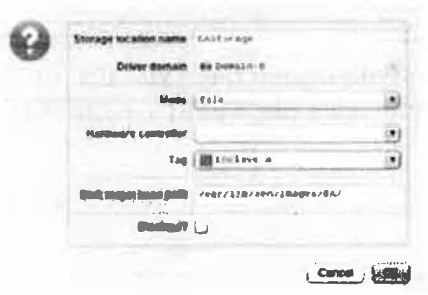
For the remote dnsmasq process, however, we need to manufacture this file and store it on the remote DHCP server. We also need to perform manual updates whenever changes occur, followed by reloading the updated configuration in the dnsmasq service. Changes include adding or modifying network switches and creating and or updating VM network interfaces, etc. For example, the following file was constructed for the bridged network we created earlier. The file specifies the network switch and user domain related network information.

```
# xenon-enterprise dnsmasq configuration (manually created)
dhcp-ignore=tag:!known
# Configure eth3 to serve dnsmasq requests with subnet 172.19.0.0/16
interface=eth3
dhcp-range=eth3,tag:EnclaveA,172.19.1.3,static,255.255.0.0
dhcp-option=eth3,tag:EnclaveA,option:router,172.19.0.1
dhcp-option=eth3,tag:EnclaveA,option:dns-server,172.19.0.1
dhcp-option=eth3,tag:EnclaveA,option:domain-search,172.19.0.1
dhcp-host=00:16:3e:f3:e2:d4,172.19.1.102,EA-domu-2,set:EnclaveA
```

4.3.3 Virtual Storage Setup

VM storage comes in many formats and modes. Disk formats can be either qcow2 or img, and *mode* specifies whether the storage is one physical disk or some allocated storage space on the host pointed to by a filename. Remote storage allocation is another option for VMs as long as the storage is accessible by the Xenon hosts on which the VM may run. We will discuss remote storage allocation later.

For the VMs in the security policy, we pre-allocated local storage using files under a specified directory path. Right-click on Domain-0 and select *Add storage location*. Enter a name for the storage location and choose file as its *Mode*. Leave *Hardware controller* unspecified but select the tag and specify *Disk target base path* using an absolute path followed by a "/" [FIGURE 36].


Storage location name EAstorage

Driver domain [illegible]

Mode File

Hardware controller

Tag Enclave A

[illegible] /var/lib/xen/images/EA/

[illegible]

Figure 36 - Allocate Storage Location for VM

Next, we assign storage for each VM in Enclave A. For example, right-click on *EA-domu-1 -> Configure domain…*. Select the *Disks* tab and the *Add disk* button. Highlight *EAstorage* as the storage resource. Click *OK* to continue. On the *Add disk* widget, provide the following user input [FIGURE 37]:

```
Enable disk: checked
Relative target: xxxxxx.qcow2
Format: qcow2
Vdev: xvda
```

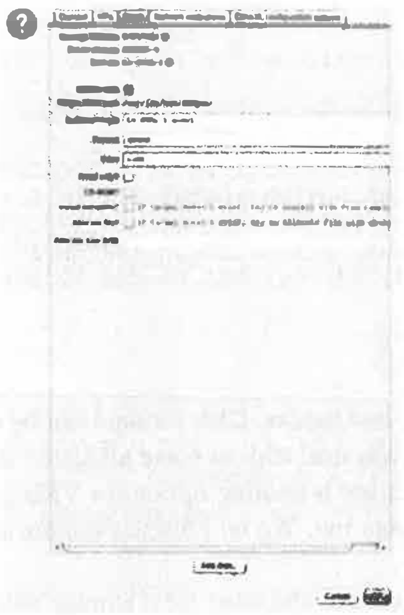


Figure 37 - Allocate Individual Storage using VM Configuration Disks Tab

Save and activate the configuration. Switch to Live mode and start each VM accordingly. Access them with either a VNC viewer or using the XL toolstack (*xl* command) on the command line. At this point, host-specific VM configuration is complete. FIGURE 38 shows the local resource allocation on one of the Xenon hosts. Notice the host allocated a bridge and a storage location for each enclave it manages.

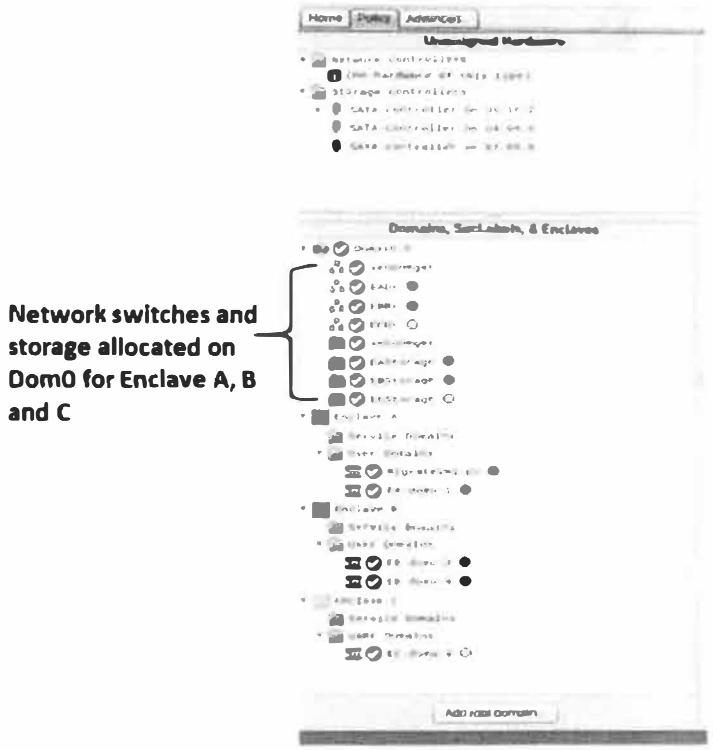
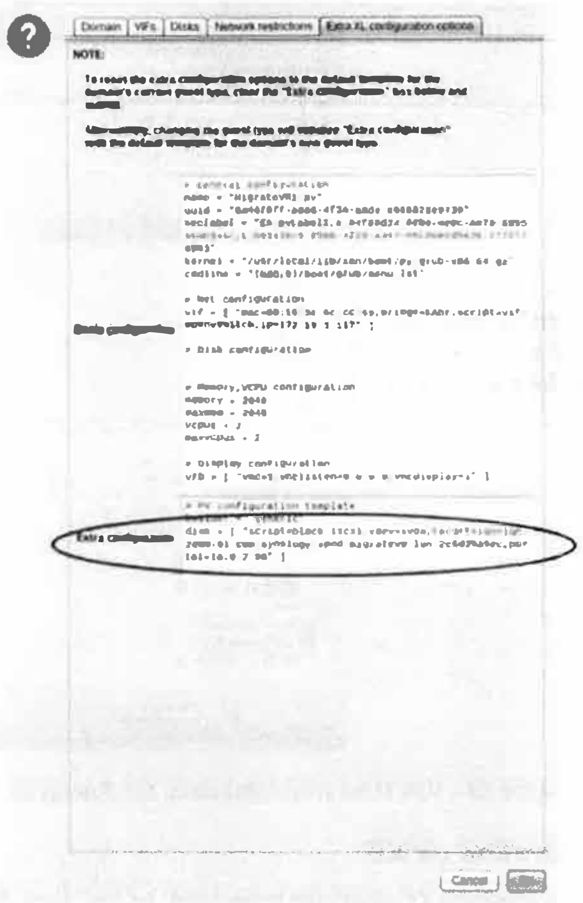

Home
Policy
Advanced
Unassigned Hardware
Domains, SecLabels, & Enclaves
Network switches and storage allocated on Dom0 for Enclave A, B and C
Add root domain

Figure 38 - VM Resource Allocation for Enclaves

4.3.4 Preserve VM Configuration on XE

As we have seen in the previous section, XE does not keep track of any host-dependent hardware information of its VMs. However, in certain situation such as VM migration, XE must be made aware of VM-specific hardware information so that the compute service can migrate a VM from one host to another. Hardware information include for example, the location of the network storage device used by the VM or network configuration.

XEGUI provides an option for preserving VM configuration in the interactive VM management tool. Right-click on the target domain to bring up the context menu. Select *Configure domain...* and then the *Extra XL configuration options* tab [FIGURE 28]. Enter the extra VM configuration in the extra configuration textbox. In this example, we specify the location of a network storage device used by the VM. Click *OK* to save. Once the input is complete, the policy service propagates the VM's extra configuration downstream to its Xenon host. As shown in FIGURE 39, the VM's extra XL configuration on the local XPGUI has been updated accordingly.



NOTE:

To reset the extra configuration options to the default template for the domain's current guest type, clear the "Extra configuration" box below and submit.

[illegible], changing the guest type will replace "Extra configuration" with the default template for the domain's new guest type.

Basic configuration

```
# General configuration
name = "MigrateVM1 pv"
uuid = "[illegible]"
seclabel = "[illegible]"
kernel = "/usr/local/lib/xen/boot/pv-grub-x86_64.gz"
cmdline = "(hd0,0)/boot/grub/menu.lst"

# Net configuration
vif = [ "mac=00:16:3e:[illegible],bridge=[illegible],script=vif-openvswitch,ip=172.[illegible]" ]

# Disk configuration


# Memory,VCPU configuration
memory = 2048
maxmem = 2048
vcpus = 1
maxvcpus = 1

# Display configuration
vfb = [ "vnc=1 vnclisten=0.0.0.0 vncdisplay=1" ]
```

Extra configuration

```
# PV configuration template
builder = "generic"
disk = [ "script=block-iscsi,vdev=xvda,target=iqn=[illegible] 2000-01.com.synology:[illegible] [illegible]=10.0.7.98" ]
```

Figure 39 - Persistent VM Configuration on XEGUI

4.4 VM Migration

VM migration is an important feature for managing VMs in a multi-tenant and cloud enabled environment. One Xenon security policy ruleset in SECTION 2 stated that Xenon VMs can only migrate to hosts that its enclave spans. Other preconditions must be met as well for VM migration:

- VMs can only migrate between servers with identical (or similar enough to be compatible) CPU capabilities
- VM storage must be shared and accessible to servers involved in the migration process
- Persistent network connectivity is required to prevent VM downtime and network disruption.

To migrate the VM, simply right mouse click on the VM and select Migrate to bring up a list of available hosts to which to migrate [FIGURE 40].

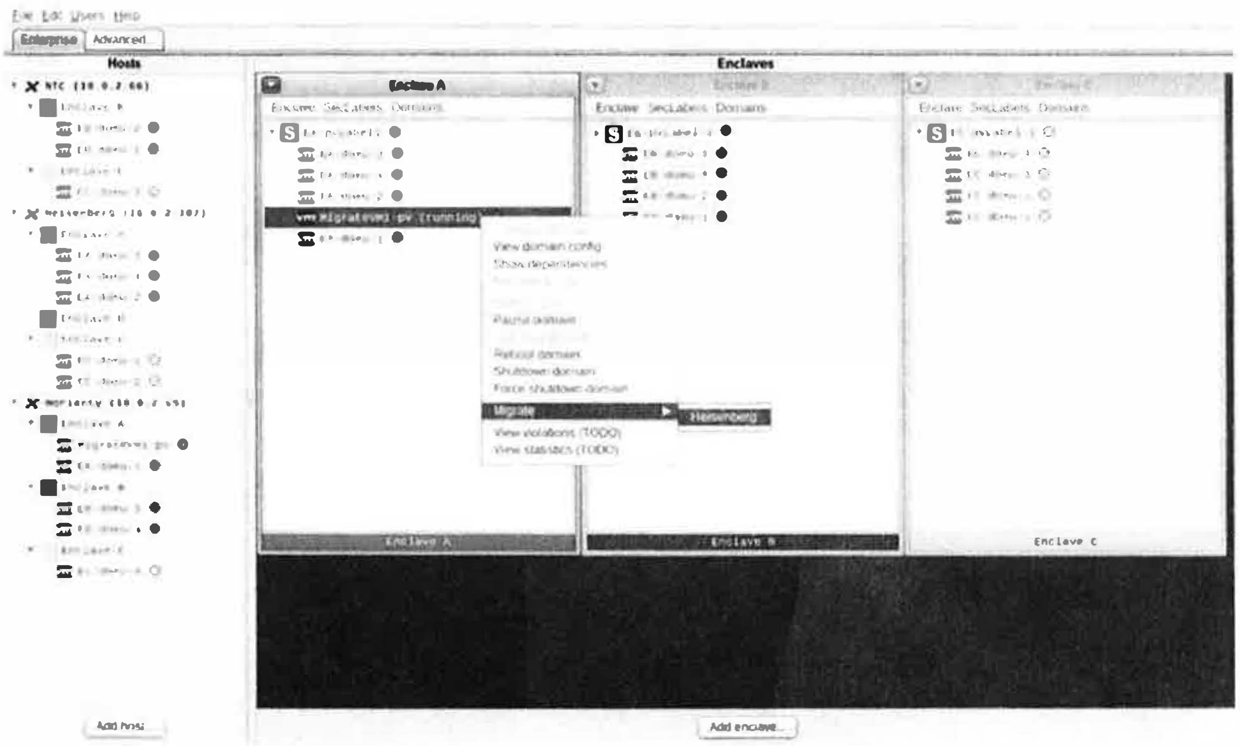


Figure 40 - VM Migraton

5 Conclusion

Our Xenon Enterprise prototype confirms that strong Xenon VMM separation with security enforcement can be achieved at enterprise scale. The Xenon Enterprise GUI management tool is intuitive and easy to use. The enterprise services enable constructing a security policy that meets separation and security requirements for complex VMM systems and is easy to verify. Each policy component is configurable by selection for efficiency during policy construction and modification. On the VMM level, each autonomous Xenon host managed by Xenon Enterprise is equipped with Xenon management toolset for implementing and verifying the security policy rules put in place at the enterprise level.