



SECURITY ADVANTAGES OF SOLARIS™ ZONES SOFTWARE

Dr. Christoph Schuba, Sun Microsystems

Sun BluePrints™ Online

Part No 820-7136-10
Revision 1.0, 12/10/08

Table of Contents

Chapter 1. Introduction	1
Chapter 2. Solaris Zones Architecture	2
Branded Zones	3
Labeled Zones	4
Zones and Networking	4
Shared-IP Zones	4
Exclusive-IP Zones	4
Zone Identity, CPU Visibility, and Packaging	5
Zones and Devices	5
Chapter 3. Getting Started with Zones	6
Zones Administration	6
Creating, Installing, and Booting a Zone for Apache HTTP Server	7
Chapter 4. The Security Advantages of OS Virtualization	10
Isolation and Encapsulation	10
Offering Replicated or Redundant Services Using Zones	11
Hardening a Web-Facing Web Server Using Solaris Zones	12
A Reduced Set of Privileges for Non-Global Zones	13
Benefits of Exclusive IP Stack Instances	15
Monitoring Events in Zones	16
Auditing Events in Non-Global Zones	16
Chapter 5. For More Information	18
About the Author	18
References	18
Ordering Sun Documents	19
Accessing Sun Documentation Online	19

Chapter 1

Introduction

Virtualization is emerging as an important tool as organizations look to consolidate redundant and aging infrastructure and create a more agile and cost-effective datacenter. Indeed, virtualization technologies can help organizations quickly recover from disasters, reduce time to market for new services, and better utilize existing infrastructure to reduce space, power, and cooling requirements. It can help increase service levels while delivering security that once required the use of individual servers. In particular, operating system (OS) level virtualization allows multiple applications to share the same operating system instance while providing separate security domains for each application.

In an OS virtualized environment, the kernel provides multiple isolated user space instances. In the Solaris™ Operating System (Solaris OS) and OpenSolaris™ operating system, such instances are called Solaris Zones. A *zone* is a virtual operating system abstraction that provides a protected environment in which applications run isolated from other applications on the system. Zones look and feel to users and administrators like separate operating system instances. Fine-grained resource management limits the amount of resources applications within zones are allowed to consume. Such application containment provides a variety of security advantages.

- Damage caused by an application isolated in a zone remains contained within that zone, as if the application ran on a dedicated machine. In other words, applications are protected from each other to provide software fault isolation.
- Applications that execute in zones have little ability to interact with privileged system processes or resources, as a limited set of privileges are available to them.
- Different Internet Protocol Security (IPSec), packet filtering, and virtual LAN (VLAN) access policies can be employed for applications in different zones on the same machine using exclusive Internet Protocol (IP) stack instances.
- Zones are the primary mechanism used to implement data separation in Solaris Trusted Extensions, an advanced security feature that implements labels to protect data and applications based on their sensitivity level, not just on who owns or runs them.
- Virtual machine-based introspection is enabled from the administrative *global zone*, the place where the kernel runs, and from where the system is controlled and configured.

This Sun BluePrints™ article explains zone technology and provides detailed examples for configuration and exploration.

Chapter 2

Solaris Zones Architecture

OS virtualization can be thought of as an advanced extension of the UNIX® `chroot` mechanism. It provides an isolated operating system environment in which each zone supplies security, a name space, and fault isolation supplemented by resource management to prevent processes in one zone from using too much of a system resource, or to guarantee processes a certain service level. To ease the management of multiple applications and environments, zones co-exist within one operating system instance and usually are managed as a single entity.

Most software that runs on the Solaris OS runs unmodified in a zone. Because zones do not change APIs or the application binary interface (ABI), applications do not need to be recompiled to run in a zone. However, applications that normally run as `root` or with certain privileges may not run in a zone if they rely on being able to access or change a global resource, such as the system's time of day clock.

Examples of applications that can run in a zone include:

- Applications that access the network and files, and perform no other I/O.
- Testing environments that allow students to experiment with software that requires traditional root privileges. Such environments are possible because a superuser in a zone cannot affect or obtain privileges in other zones.
- Testing environments configured to test different versions of applications on the same operating system.
- System configurations supporting shared use, either in educational or enterprise environments. For example, each student can be allocated an IP address and a zone, allowing students to safely share a single machine.

Some applications might need to be modified to run in a zone, including:

- Applications that require direct access to certain devices, such as a disk partition, usually work if the zone is configured correctly. In some cases, doing so can increase security risks.
- Applications that require direct access to memory or a network device. Such applications can be modified to use an IP service.

More information on zones technology can be found at <http://opensolaris.org/os/community/zones>.

Branded Zones

While zones provide protected environments for Solaris applications, separate and protected runtime environments are available through the Branded Zone (BrandZ) framework that extends the zones infrastructure to create zones that contain non-native operating environments. A Branded Zone can be as simple as an environment where the standard Solaris OS utilities are replaced with GNU equivalents, or as complex as a complete Linux user space environment.

BrandZ extends the zones infrastructure in the user space in the following ways:

- A *brand* is an attribute of a zone that is set at zone configuration time.
- Each brand provides its own installation routine, enabling the installation of an arbitrary collection of software in the Branded Zone.
- Each brand can include pre-boot and post-boot scripts that provide final boot time setup or configuration.
- The `zonecfg` and `zoneadm` tools can set and report a zone's brand type.

BrandZ provides a set of interposition points in the kernel that are found in the system call path, process loading path, thread creation path, and more. At these interposition points, code is executed that is applied only to processes in a Branded Zone, and the brand can choose to supplement or replace the standard behavior of the Solaris OS. Fundamentally different brands can require new interposition points.

For example, the `linux (lx)` brand lets Linux binary applications run unmodified on a Solaris system in zones that run a complete Linux user space. In other words, user-level Linux software runs on a machine with a Solaris kernel¹. While the `linux` brand runs on x86 and x64 systems booted with a 32-bit or 64-bit kernel, only 32-bit Linux applications are supported. The Solaris OS includes tools for installing a CentOS or Red Hat Enterprise Linux distribution inside a zone on a Solaris system.

Note – The `linux` brand does not work with Solaris Trusted Extensions, as it requires a kernel loadable module that keeps Solaris Trusted Extensions from enforcing its mandatory security policies.

More information on Branded Zones can be found at <http://opensolaris.org/os/community/brandz>.

1. The `linux` brand currently is available for x86 systems from multiple vendors, and x64 systems incorporating AMD Opteron™ processors.

Labeled Zones

Solaris Zones are the primary mechanism used to implement data separation in Solaris Trusted Extensions, a feature of the operating system that enforces multilevel security (MLS) policies. Each zone is assigned a unique sensitivity label. *Solaris Trusted Extensions Administrator's Procedures* explains the Trusted Extensions security concepts and software features in detail.

Zones and Networking

A Solaris Zone can be designated as a shared-IP zone or an exclusive-IP zone.

Shared-IP Zones

Shared-IP zones share the IP stack with the global zone. As a result, shared-IP zones are shielded from the configuration details for devices, routing, and so on. Each shared-IP zone can be assigned Internet Protocol version 4 (IPv4) or Internet Protocol version 6 (IPv6) addresses, and has its own port space. However, shared-IP zones cannot change their network configuration or routing table, or see the configuration of other zones. Because the `/dev/ip` device file is not present in a shared-IP zone, Simple Network Management Protocol (SNMP) agents must open the `/dev/arp` device file. Multiple shared-IP zones can share a broadcast address and join the same multicast group.

A shared-IP zone can send packets only on an interface on which it has an address, and cannot see the traffic of other zones. Packets originating from the zone have a source address belonging to that zone. A default router can be used only if the router is directly reachable from the zone, and is in the same IP subnet as the zone. In addition, applications can bind to the `INADDR_ANY` address and receive traffic only for that zone.

Shared-IP zones have several networking limitations, including:

- A physical interface cannot be put inside a zone.
- The `ipfilter` command does not work between zones.
- The Dynamic Host Configuration Protocol (DHCP) cannot be used to obtain a the IP address for a shared-IP zone.
- Dynamic routing is not possible.

Exclusive-IP Zones

Exclusive-IP zones have a unique IP stack, and can have dedicated physical and VLAN interfaces. The configuration of exclusive-IP zones is the same as that of a physical machine. Exclusive-IP zones cannot see the traffic of other zones. Packets originating from an exclusive-IP zone have a source address belonging to that zone.

Exclusive-IP zones do not have the limitations of shared-IP zones, and can change their network configuration or routing table inside the zone. In addition, the `/dev/ip` device file is present in exclusive-IP zones.

Zone Identity, CPU Visibility, and Packaging

Zones have several properties related to identity, CPU visibility, and software packages and patches.

- *Identity*

Each zone controls its node name, time zone, and naming services, such as LDAP and the Network Information Service (NIS). The `sysidtool` can be used to configure these attributes. The ability to maintain separate and different `/etc/passwd` files is the mechanism used to delegate `root` privileges to a zone. User IDs can map to different names when domains differ.

- *CPU visibility*

By default, all zones see all CPUs. A restricted view automatically occurs when resource pools are enabled.

- *Software packages and patches*

While zones can add and patch their own packages, system patches are applied in the global zone. Non-global zones automatically boot in superuser mode (`boot -s`) to apply patches. It is important to note that the `SUNW_PKG_ALLZONES` package must be kept consistent between the global zone and all non-global zones. The `SUNW_PKG_HOLLOW` package causes the package name to appear in non-global zones for dependency purposes, however the contents are not installed.

Zones and Devices

Each zone has its own set of devices that applications reference via the logical device path listed in the `/dev` directory. A subset of safe pseudo devices is available, including the `random`, `console`, and `null` pseudo devices. While zones can modify the permissions of their devices, they cannot issue the `mknod(2)` system call. Physical device files, such as those used for raw disks, can be placed in a zone with caution.

Note – The `/devices` directory does not exist in non-global zones.

It is important to evaluate security concerns before sharing devices among zones. For example, devices can be assigned to specific zones. Allowing unprivileged users to access block devices could result in those devices causing system panics, bus resets, or other unwanted events. In addition, placing a physical device in multiple zones can create a covert channel between zones. Global zone applications that use such a device risk the possibility of compromised data or data corruption by a non-global zone.

Chapter 3

Getting Started with Zones

This chapter provides an introduction to working with zones, including the tools needed and instructions for creating, installing, and booting a zone.

Zones Administration

Table 3-1 lists administration commands, global scope properties, and resource types that are important when working with zones.

Table 3-1. Zone administration commands, global scope properties, and resource types

Administration Command	Description
<code>zonectfg</code>	Creates and configures zones (adds resources and properties), and stores the configuration in a private XML file in the <code>/etc/zones</code> directory
<code>zoneadm</code>	Performs administrative steps for zones, such as list, install, boot, reboot, and halt
<code>zlogin</code>	Allows users to log in to the zone to perform maintenance tasks
<code>zonename</code>	Displays the current zone name
Global Scope Property	Description
<code>zonepath</code>	Specifies the path in the global zone to the root directory under which the zone is to be installed
<code>autoboot</code>	Indicates whether or not to boot the zone when the global zone boots
<code>pool</code>	Defines the resource pools to which the zone is to be bound
<code>brand</code>	Identifies a particular collection of software that is present in the non-global zones that does not match the software found in the global zone
Resource Type	Description
<code>fs</code>	Specifies the file system type
<code>inherit-pkg-dir</code>	Specifies the directory contains the packages inherited from the global zone
<code>net</code>	Specifies the network device
<code>device</code>	Specifies the device

Creating, Installing, and Booting a Zone for Apache HTTP Server

The following example introduces the process of creating, installing, and booting a zone¹. The process involves creating a non-global zone and starting the Apache HTTP Server in it, resulting in only this service running. The only reachable network interface to the system is configured to be port 80, resulting in a minimized attack surface.

This and subsequent examples assume the system has a `/zone` directory with sufficient space allocated for zones. The directory can be a regular directory, or a zpool mount point (preferred.) Follow the steps to configure the new zone, named `apache1`, that uses a default shared IP stack.

1. Create a zone named `apache1`.

```
# zonecfg -z apache1
apache1: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:apache1> create
zonecfg:apache1> set zonepath=/zone/apache1
zonecfg:apache1> add net
zonecfg:apache1:net> set address=10.0.2.100/24
zonecfg:apache1:net> set physical=e1000g0
zonecfg:apache1:net> end
zonecfg:apache1> verify
zonecfg:apache1> commit
zonecfg:apache1> exit
```

2. Install the zone. The necessary directories are created automatically and the zone is ready for booting once the command completes.

```
# zoneadm -z apache1 install
Preparing to install zone
Creating list of files to copy from the global zone.
Copying <7020> files to the zone.
Initializing zone product registry.
Determining zone package initialization order.
Preparing to initialize <1075> packages on the zone.
Initialized <1075> packages on zone.
Zone <apache1> is initialized.
Installation of these packages generated warnings: ....
The file </zone/apache1/root/var/sadm/system/logs/install_log>
contains a log of the zone installation.
```

3. View the directories in the zone.

```
# ls /zone/apache1/root
bin  etc      home    mnt      platform  sbin    tmp      var
dev  export  lib     opt      proc      system  usr
```

1. An example using a linux Branded Zone can be found in *Limiting Service Privileges in the Solaris 10 Operating System*.

4. Boot the new zone and view the network interfaces.

```
# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL>
mtu 8232 index 1
    inet 127.0.0.1 netmask ff000000
e1000g0: flags=1004803<UP,BROADCAST,MULTICAST,DHCP,IPv4> mtu 1500
index 2
    inet 10.0.2.15 netmask ff000000 broadcast 10.0.2.255
    ether 8:0:27:61:75:4
# zoneadm -z apache1 boot
# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL>
mtu 8232 index 1
    inet 127.0.0.1 netmask ff000000
lo0:1: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL>
mtu 8232 index 1
    zone apache1
    inet 127.0.0.1 netmask ff000000
e1000g0: flags=1004803<UP,BROADCAST,MULTICAST,DHCP,IPv4> mtu 1500
index 2
    inet 10.0.2.15 netmask ffffffff broadcast 10.0.2.255
    ether 8:0:27:61:75:4
e1000g0:1: flags=1004803<UP,BROADCAST,MULTICAST,DHCP,IPv4> mtu 1500
index 2
    zone apache1
    inet 10.0.2.100 netmask ffffffff broadcast 10.0.2.255
```

5. Log in to the zone. The `-c` command line option to the `zlogin` command opens the zone console. The `~.` key sequence is used to exit the console connection. Upon initial console login, users must complete the system configuration, including the selection of terminal type, configuration of network identity with host name and name service, selection of time zone, and choice of root password. This configuration step is omitted in the following example output.

```
# zlogin -C apache1
[Connected to zone 'apache1' console]
```

6. Configure the zone and verify it is operational using the ping command.

```
# ifconfig -a
lo0:1: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL>
mtu 8232 index 1
    inet 127.0.0.1 netmask ff000000
e1000g0:1: flags=1000803<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu
1500 index 2
    inet 10.0.2.100 netmask ffffffff broadcast 10.0.2.255
# ping -s 10.0.2.100
64 bytes from 10.0.2.100: icmp_seq=0. time=0.392 ms
64 bytes from 10.0.2.100: icmp_seq=1. time=0.265 ms
64 bytes from 10.0.2.100: icmp_seq=2. time=0.270 ms
^C
----10.0.2.100 PING Statistics----
3 packets transmitted, 3 packets received, 0% packet loss
round-trip (ms)  min/avg/max/stddev = 0.265/0.309/0.392/0.072
# ~.
[Connection to zone 'apache1' console closed]
```

Chapter 4

The Security Advantages of OS Virtualization

Many organizations avoid potential security issues by deploying only one operating system and application per server. As consolidation strategies are employed to affect better resource utilization, protecting applications and systems becomes paramount. Zones technology provides a variety of security benefits that can help companies host multiple environments and applications on a single server without damaging or exposing one another to security risks.

Isolation and Encapsulation

Isolation can be illustrated with the example of executing applications in sandboxes, such as Java™ technology applications within the Java™ virtual machine (JVM™)¹ sandbox. Damage from an exploited or misbehaving application is contained and cannot be inflicted on other services. This idea can be employed for standalone services, such as a Web server, or for services consisting of an aggregate of middleware, such as e-commerce applications in a tiered system architecture, where the different tiers are isolated from each other. In general, different applications, or aggregates of applications or middleware, can each be run in their own zone.

The encapsulation of single applications or aggregates of applications that jointly comprise a network facing service has additional security benefits. Many popular security mechanisms, such as firewall or intrusion detection technology, are applied at the host level and are readily applied to zones. In fact, one reason for introducing exclusive IP stack instances into zones was to create the ability to apply per-zone packet filtering within a virtual operating system.

Concentrating on isolated, encapsulated application environments yields a virtual appliance model. Hardware appliances, such as load balancers or network packet filtering routers, cater to specific functions by offering high performance with low administrative overhead. Similarly, virtual appliances offer only individual application functionality and the operating system support needed for the application to run well. The operating environment surrounding the application can be hardened and minimized.

The interface that is accessible from the network, the *attack surface*, is minimized and restricted to what the application needs to expose. For example, virtual e-mail appliances expose only the Simple Mail Transport Protocol (SMTP) on port 25 to the network. Additionally, administrative access can be restricted to local access. Both the smaller trusted computing base and the reduced attack surface improve the security

1. The terms “Java Virtual Machine” and “JVM” mean a Virtual Machine for the Java™ platform.

posture of a system. Furthermore, ISVs employing the virtual appliance model can provide more hardened applications and reduce the application security burden for traditional system administrators.

Each zone has its own characteristics, such as zonename, IP address, hostname, naming services, and root and non-root users. By default, the operating system runs in the global zone. Administrators can virtualize the execution environment by defining one or more non-global zones. Network-facing services can be run in separate zones, limiting the potential for damage in the event of security violations. Because zones are implemented in software, they are not limited to the granularity defined by hardware boundaries. Instead, zones offer sub-CPU resource granularity.

Offering Replicated or Redundant Services Using Zones

The following example demonstrates how to support two separate sets of Web server user groups on one physical host. Simultaneous access to both Web servers is configured to protect each Web server and the system in the event one becomes compromised.

1. Verify the configuration of the non-global zone named `apache1`.

```
# zonecfg -z apache1 info
zonename: apache1
zonepath: /zone/apache1
brand: native
autoboot: false
bootargs:
pool:
limitpriv:
scheduling-class:
ip-type: shared
inherit-pkg-dir: dir: /lib
inherit-pkg-dir: dir: /platform
inherit-pkg-dir: dir: /sbin
inherit-pkg-dir: dir: /usr
net: address: 10.0.2.100/24
    physical: e1000g0
```

2. Create a matching non-global zone named `apache2`. Follow the directions for the zone named `apache1`, but change the network address. The network address for `apache2` is highlighted in blue in the following output.

```
# zonecfg -z apache2 info
zonename: apache2
zonepath: /zone/apache2
brand: native
autoboot: false
bootargs:
pool:
limitpriv:
scheduling-class:
ip-type: shared
inherit-pkg-dir: dir: /lib
inherit-pkg-dir: dir: /platform
inherit-pkg-dir: dir: /sbin
inherit-pkg-dir: dir: /usr
net: address: 10.0.2.200/24
      physical: e1000g0
```

3. Log in to the `apache1` zone and start the Web server application.

```
# zlogin apache1
# zonename
apache1
# cp /etc/apache2/httpd.conf-example /etc/apache2/httpd.conf
# svcadm enable svc:/network/http:apache2
```

4. Repeat step 3 for the zone named `apache2`.
5. Open two tabs in a Web browser in the global zone. Navigate to the `http://100.0.2.100/manual/` and `http://100.0.2.200/manual/` URLs.

Two separate Web servers are up and running in the zones named `apache1` and `apache2`. Each zone appears as a different system to end users. Each Web server has its own name service: `/etc/nsswitch.conf` and `/etc/resolv.conf`. Because the Web servers each reside in their own zone, a malicious attack on one Web server is contained. Port conflicts are not an issue, as each non-global zone uses its own port 80.

Hardening a Web-Facing Web Server Using Solaris Zones

Web page hijacking is a common threat in the Internet at large. Solaris Zones software can be used to construct a system that is hardened against the threat of Web hijacking. Figure 4-1 depicts a single machine with two network interfaces. One network interface is connected to the Internet, while the other network interface is connected to an internal network. Two zones are instantiated on the machine. Each zone has exclusive access to one interface. The zone named `apache1` faces the internal network and is used for staging the production content to be served by the Web server. The `apache2` zone mounts the production content of the `apache1` zone in a read-only fashion.

More information on using Solaris Zones software to harden systems can be found in *Eliminating Web Page Hijacking Using Solaris 10 Security* located at <http://www.sun.com/software/solaris/howtooguides/s10securityhowto.pdf>

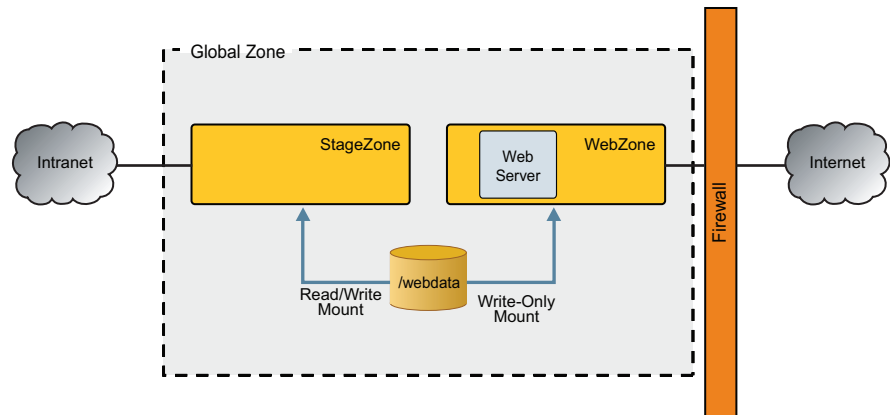


Figure 4-1. System hardening against Web page hijacking

A Reduced Set of Privileges for Non-Global Zones

The Solaris OS implements a set of privileges that provide fine-grained control over the actions of processes. Traditionally, UNIX systems rely on the concept of a superuser called `root` with a special user identifier (0). The Solaris OS breaks down the concepts of superuser and regular users into discrete privileges, and allows one or more specific privileges to be granted that enable processes to perform operations that usually are restricted.

There are some 68¹ privileges that can be assigned individually to processes using the Service Management Facility (SMF), role-based access control (RBAC), or command-line programs such as `ppriv`, and more. Processes in the global zone potentially have access to all privileges. The `ppriv -l` command can be used in the global zone to list available privileges.

```
# zonename
global
# ppriv -l
contract_event contract_observercpc_cpu          dtrace_kernel  dtrace_proc
dtrace_user    file_chown      file_chown_self file_dac_execute file_dac_read
file_dac_search file_dac_write  file_downgrade_sl file_link_any    file_owner
file_setid     file_upgrade_sl graphics_access  graphics_map     ipc_dac_read
ipc_dac_write  ipc_owner      net_bindmlp     net_icmpaccess  net_mac_aware
net_privaddr   net_rawaccess  proc_audit      proc_chroot     proc_clock_highres
proc_exec     proc_fork      proc_info       proc_lock_memory proc_owner
proc_prioctl  proc_session   proc_setid      proc_taskid     proc_zone
sys_acct      sys_admin      sys_audit       sys_config      sys_devices
sys_ipc_config sys_linkdir    sys_mount       sys_ip_config   sys_net_config
sys_nfs       sys_res_config sys_resource     sys_suser_compat sys_time
sys_trans_label win_colormap   win_config      win_dac_read    win_dac_write
win_devices   win_dga        win_downgrade_sl win_fontpath    win_mac_read
win_mac_write win_selection  win_upgrade_sl
```

1. Solaris 10 OS, Update 5 has 68 discrete privileges available within the global zone.

Processes running in a non-global zone are restricted to a subset of privileges that keep one zone from performing operations that might affect other zones, and the global zone in particular. Restricted privileges are mandatory in a zone, and add to the benefit of running processes with reduced privileges.

Use the `ppriv -l` command in the zone console to see the list of available privileges. The output indicates that the non-global zone `apache1` is restricted to fewer privileges than the global zone. Note that the example output shows the default set of privileges available to zones.

```
# zonename
apache1
# ppriv -l
contract_event contract_observerfile_chown    file_chown_self  file_dac_execute
file_dac_read  file_dac_search  file_dac_write  file_link_any    file_owner
file_setid     ipc_dac_read     ipc_dac_write   ipc_owner        net_bindmlp
net_icmpaccess net_mac_aware   net_privaddr    proc_audit       proc_chroot
proc_exec      proc_fork        proc_info       proc_lock_memory proc_owner
proc_session   proc_setid       proc_taskid     sys_acct         sys_admin
sys_audit      sys_mount        sys_nfs         sys_resource
```

This zone-limiting set can be modified. A number of privileges can be added, and existing ones can be removed. The following example shows how optional zone privileges are added, and how existing zone privileges are removed.

```
# zonename
global
# zonecfg -z apache1 info limitpriv
limitpriv:
# zonecfg -z apache1 set limitpriv="default,sys_time,!net_icmpaccess"
# zonecfg -z apache1 info limitpriv
limitpriv: default,sys_time,!net_icmpaccess
# zonecfg -z apache2 info limitpriv
limitpriv:
```

The `ping` command fails to reach the zone named `apache1` because the `PRIV_NET_ICMPACCESS` privilege is missing. However, the `ping` command can reach the zone named `apache2`.

```
# zonename
global
# zlogin apache1 ppriv -e ping localhost
ping: socket Permission denied
# zlogin apache2 ppriv -e ping localhost
localhost is alive
```

To further demonstrate the effect of the restrictions, run the `ndd` command to set a driver configuration parameter. Set the TCP stack into debug mode. Note that this command succeeds in the global zone.

```
# zonename
global
# ndd /dev/tcp tcp_debug
0
# ndd -set /dev/tcp tcp_debug 1
# ndd /dev/tcp tcp_debug
1
# ndd -set /dev/tcp tcp_debug 0
```

Run the same `ndd` command in the non-global zone, and note that the command fails. Run the command again, this time as an argument to the `ppriv -e -D` debugging command to determine why the command failed.

```
# zonename
apache1
# ndd /dev/tcp tcp_debug
0
# ndd -set /dev/tcp tcp_debug 1
operation failed: Not owner
# ppriv -e -D ndd -set /dev/tcp tcp_debug 1
ndd[4335]: missing privilege "sys_ip_config" (euid = 0, syscall = 54)
needed at nd_getset+0xcf
operation failed: Not owner
```

The missing `PRIV_SYS_IP_CONFIG` privilege allows a process to perform actions, such as configuring network interfaces and routes, or accessing TCP/IP parameters using the `ndd` command.

Benefits of Exclusive IP Stack Instances

The use of exclusive IP instances lets organizations use zones with network-based partitioning for large-scale deployments. In a large-scale deployment, aggregates of applications that work in concert are deployed on physically separate machines. These aggregates communicate with each other over the network. The network connection is typically secured using IPsec or by being encapsulated within VLANs.

Exclusive IP stack instances make per-zone IPsec and VLAN access available to applications running in the associated zones. It is also possible to install a packet sniffer inside a zone with an exclusive IP stack, a capability that is not available with a shared IP stack. However, this feature can be a potential drawback — a successful attacker could be the one who installs a sniffer in a zone.

Monitoring Events in Zones

Traditional architectures for intrusion detection and auditing provide two options.

- With a host-based option, the detection and auditing software runs on the system being monitored. While this option provides an excellent view of what is happening, it is susceptible to attack.
- With a network-based option the monitoring software resides on the network. While this option is more resilient to host-based attacks, it has limited visibility into what is happening inside the host.

A zones model offers another option. The monitoring software can run in the host operating system or hypervisor, while the monitored software consists of the application software and the guest operating system. This approach separates and isolates the auditor from the monitored software, while maintaining visibility into a guest's state.

Auditing can be enabled on a system running the Solaris OS, giving administrators the ability to monitor and record what is happening in all zones. Non-global zones cannot interfere. Other monitoring examples include:

- Examining the state of processes or threads that run in non-global zones
- Extending Solaris Dynamic Tracing (DTrace) probes into processes that are running in non-global zones
- Monitoring packet filtering actions in non-global zones that are running exclusive IP stack instances

Auditing Events in Non-Global Zones

The following example enables auditing in the global zone and records events in non-global zones (`apache1`) in the audit trail.

1. Enable auditing in the Solaris OS.

```
# zonename
global
# /etc/security/bsmconv
This script is used to enable the Basic Security Module (BSM).
Shall we continue with the conversion now? [y/n] y
bsmconv: INFO: checking startup file.
bsmconv: INFO: turning on audit module.
bsmconv: INFO: initializing device allocation.

The Basic Security Module is ready.
If there were any errors, please fix them now.
Configure BSM by editing files located in /etc/security.
Reboot this system now to come up with BSM enabled.
```

2. Reboot the system to make auditing effective for all processes.

```
# reboot
```

3. Login to the non-global zone named `apache1` and display the audit record in the global zone. The final entry in the audit file, highlighted in blue, records the `root` login to the zone named `apache1` (IP address `10.0.2.100`).

```
# # zonename
global
# /praudit /var/audit/20080703221826.not_terminated.unknown
file,2008-07-03 15:18:26.774 -07:00,
header,36,2,AUE_SYSTEMBOOT,na,2008-07-03 15:17:10.301 -07:00
text,booting kernel
header,89,2,AUE_login,,unknown,2008-07-03 15:19:55.310 -07:00
subject,root,root,root,root,root,716,716,0 0 unknown
text,successful login
return,success,0
header,69,2,AUE_login,,unknown,2008-07-03 15:25:06.245 -07:00
subject,root,root,root,root,root,1439,3290887400,0 0 10.0.2.100
return,success,0
```

Chapter 5

For More Information

About the Author

Christoph Schuba studied mathematics and management information systems at the University of Heidelberg and the University of Mannheim in Germany. As a Fulbright Scholar, he earned his M.S. and Ph.D. degrees in Computer Science from Purdue University in 1993 and 1997, performing most of his dissertation research in the Computer Science Laboratory at the Xerox Palo Alto Research Center (PARC). Christoph has taught undergraduate and graduate courses in computer and network security, cryptography, operating systems, and distributed systems at San Jose State University, U.S., at the Universität Heidelberg, Germany, at the International University in Bruchsal, Germany, and at Linköpings universitet in Linköping, Sweden where he held the chair in information security.

Christoph has worked at Sun Labs since 1997, and most recently in the Solaris Software Security Organization at Sun. He holds 11 patents and is the author or co-author of numerous scientific articles in computer and network security.

References

Brunette, Glenn. "Limiting Service Privileges in the Solaris 10 Operating System," *Sun BluePrints Online*, May 2005. To access this article online, go to:

<http://sun.com/blueprints/0505/819-2680.pdf>

Brunette, Glenn. "Understanding the Security Capabilities of Solaris Zones," *Sun BluePrints Online*, December 2008. To access this article online, go to:

<http://sun.com/blueprints/1208/820-7017-10.pdf>

Thacker, Mark. "Eliminating Web Page Hijacking Using Solaris 10 Security," May 2005.

<http://sun.com/software/solaris/howtoguides/s10securityhowto.pdf>

Solaris Trusted Extensions Administrator's Procedures

<http://docs.sun.com/app/docs/doc/819-7309>

OpenSolaris Community: Zones

<http://opensolaris.org/os/community/zones>

OpenSolaris Community: BrandZ

<http://opensolaris.org/os/community/brandz>

Jaeger, Trent. "Operating System Security." Faden, Glenn and Schuba, Christoph.

"Chapter 8, Case Study: Solaris Trusted Extensions," *Synthesis Lectures on Information Security, Privacy and Trust*, Morgan & Claypool Publishers, 2008.

<http://blogs.sun.com/schuba/resource/papers/ch08sun.pdf>

Pechanec, Jan and Schuba, Christoph, and Phalan, Mark. "New Security Features in OpenSolaris and Beyond," *In Proceedings of the Second OpenSolaris Developer Conference*, Prague, Czech Republic, June 2008.

<http://blogs.sun.com/schuba/resource/papers/200807-osdevcon-sol-sec-features.pdf>

Ordering Sun Documents

The SunDocsSM program provides more than 250 manuals from Sun Microsystems, Inc. If you live in the United States, Canada, Europe, or Japan, you can purchase documentation sets or individual manuals through this program.

Accessing Sun Documentation Online

The docs.sun.com Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is

<http://docs.sun.com/>

To reference Sun BluePrints Online articles, visit the Sun BluePrints Online Web site at:

<http://www.sun.com/blueprints/online.html>

Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 USA **Phone** 1-650-960-1300 or 1-800-555-9SUN (9786) **Web** sun.com

© 2008 Sun Microsystems, Inc. All rights reserved. Sun, Sun Microsystems, the Sun logo, Java, JVM, OpenSolaris, Solaris, Sun BluePrints, and SunDocs are trademarks or registered trademarks of Sun Microsystems, Inc. or its subsidiaries in the United States and other countries. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd. AMD and Opteron are trademarks or registered trademarks of Advanced Micro Devices. Information subject to change without notice.



Printed in USA 12/08