Virtuozzo

Virtuozzo Hybrid Infrastructure 6.3

Security Hardening Guide

1/27/2025

Table of contents

About this guide

This guide is intended for system administrators and provides a comprehensive set of instructions recommended for hardening a Virtuozzo Hybrid Infrastructure cluster.

Password policy

Configure password creation requirements

Enforce the use of strong passwords to protect systems against brute-force attacks, which involve guessing password combinations.

To set password creation requirements:

1. In the `/etc/security/pwquality.conf` file, add or modify the following line for password length:

   ```
   minlen = 14
   ```

2. In the `/etc/security/pwquality.conf` file, add or modify the following line for password complexity:

   ```
   minclass = 4
   ```

3. Run the following script to update the `system-auth` and `password-auth` files:

   ```
   CP=$(authselect current | awk 'NR == 1 {print $3}' | grep custom/) for FN in system-auth password-auth; do [[ -n $CP ]] && PTF=/etc/authselect/$CP/$FN || PTF=/etc/authselect/$FN [[ -z $(grep -E '^\s*password\s+requisite\s+pam_pwquality.so\s+.*enforce-for-root\s*.*$' $PTF) ]] && sed -ri 's/^\s*(password\s+requisite\s+pam_pwquality.so\s+)(.*)$/\1\2 enforce-for-root/' $PTF [[ -n $(grep -E '^\s*password\s+requisite\s+pam_pwquality.so\s+.*\s+retry=\S+\s*.*$' $PTF) ]] && sed -ri '/pwquality/s/retry=\S+/retry=3/' $PTF || sed -ri 's/^\s*(password\s+requisite\s+pam_pwquality.so\s+)(.*)$/\1\2 retry=3/' $PTF done authselect apply-changes
   ```

Limit password reuse

Enforce a policy preventing users from reusing their last five passwords. This ensures that compromised credentials cannot be reused and applies only to local system accounts.

To configure remembered password history, run the following script that will add or modify the `pam_pwhistory.so` and `pam_unix.so` lines to include the `remember` option:

```
CP=$(authselect current | awk "NR == 1 {print $3}" | grep custom/) [[ -n $CP ]] && PTF=/etc/authselect/$CP/system-auth || PTF=/etc/authselect/system-auth [[ -n $(grep -E "^\s*password\s+(sufficient\s+pam_unix|requi(red|site)\s+pam_pwhistory).so\s+([^#]+\s+)*remember=\S+\s*.*$" $PTF) ]] && sed -ri "s/^\s*(password\s+(requisite|sufficient)\s+(pam_pwquality\.so|pam_unix\.so)\s+)(.*)(remember=\S+\s*)(.*)$/\1\4 remember=5 \6/" $PTF || sed -ri "s/^\s*(password\s+(requisite|sufficient)\s+(pam_pwquality\.so|pam_unix\.so)\s+)(.*)$/\1\4 remember=5/" $PTF authselect apply-changes
```

Set password expiration to 90 days or less

Limit the maximum password age to enforce regular credential updates. Shorter expiration periods reduce the time compromised passwords remain valid.

To change the number of days a password is active before it expires:

1. In `/etc/login.defs`, set the `PASS_MAX_DAYS` parameter to 90 days:

   ```
   PASS_MAX_DAYS 90
   ```

   Note that changes made to `/etc/login.defs` affect only new users.
2. For existing users, modify user password expiration by running:

   ```
   # chage --maxdays 90 <user>
   ```

Set minimum days between password changes to 7 or more

Restrict frequent password changes to prevent users from cycling through old credentials, ensuring adherence to password history policies.

To change the number of days a password must be active before it can be changed by a user:

1. In `/etc/login.defs`, set the `PASS_MIN_DAYS` parameter to 7:

   ```
   PASS_MIN_DAYS 7
   ```

   Note that changes made to `/etc/login.defs` affect only new users.
2. For existing users, modify minimum days between password changes by running:

   ```
   # chage --mindays 7 <user>
   ```

Set inactive password lock to 30 days or less

Disable inactive accounts as they pose a security risk. In certain situations, without monitoring login attempts or any anomalies, such accounts become a prime target for attackers who are willing to gain undetected or unauthorised access.

To change the number of days of inactivity after a password has expired before the account is locked:

1. Set the default password inactivity period to 30 days:

   ```
   # useradd -D -f 30
   ```

   Note that changes made by `useradd` affect only new users.

2. For existing users, modify the number of days of inactivity by running:

```
# chage --inactive 30 <user>
```

Access control

Ensure sudo commands use pty

Limit `sudo` access and prevent attackers from exploiting `sudo` privileges to execute malicious programs that persist even after termination.

1. Open the `/etc/sudoers` file or a file in `/etc/sudoers.d/` for editing with `visudo -f`.
2. Add the following line:

```
Defaults use_pty
```

Limit SSH access

Restrict SSH access to authorized users only mitigates the risk of unauthorized logins and brute-force attacks. Configure SSH allow lists or group-based access policies to enforce this restriction.

In the `/etc/ssh/sshd_config` file, set one or more parameters as follows:

```
AllowUsers <userlist>
AllowGroups <grouplist>
DenyUsers <userlist>
DenyGroups <grouplist>
```

Prohibit root login using passwords over SSH

Disallow root login with password-based authentication as it makes the system vulnerable to brute-force attacks. To strengthen security, enforce key-based authentication for root accounts.

1. Add an SSH key for the root user, as described in Securing root access to cluster nodes over SSH in the Administrator Guide.
2. Prohibit password authentication for the root user over SSH by running:

```
# echo 'PermitRootLogin prohibit-password' > /etc/ssh/sshd_config.d/01-
permitrootlogin.conf
```

Set SSH LoginGraceTime to one minute or less

Configure the `LoginGraceTime` parameter in SSH to a low value, such as 60 seconds, to limit the time allowed for successful authentication before a connection is terminated. This minimize the risk of brute-force attacks and limit unauthenticated connection attempts.

In the `/etc/ssh/sshd_config` file, set the `LoginGraceTime` parameter to 60:

```
LoginGraceTime 60
```

Enable CHAP authentication for iSCSI targets

Implement Challenge Handshake Authentication Protocol (CHAP) for iSCSI connections to authenticate both initiators and targets using a shared secret, preventing unauthorized devices from accessing iSCSI targets.

To configure CHAP authentication for iSCSI targets, refer to Managing CHAP users in the Administrator Guide.

System hardening

Disable USB storage

Restrict USB access on the system to reduce the physical attack surface and prevent unauthorized device connections.

1. Edit or create a configuration file in the `/etc/modprobe.d/` directory. For example:

   ```
   # vim /etc/modprobe.d/usb-storage.conf
   ```

2. Add the following line:

   ```
   install usb-storage /bin/true
   ```

3. Unload the `usb-storage` module by running:

   ```
   # rmmod usb-storage
   ```

Ensure idle shell timeout is 900 seconds or less

Configure idle shell timeouts, such as the `TMOUT` environment variable in shell profiles, to ensure unattended sessions are terminated automatically. This reduces the risk of unauthorized access.

1. Open the `/etc/bashrc`, `/etc/profile`, and `/etc/profile.d/*.sh` files (and the appropriate files for any other shell supported on your system) for editing.
2. Add or edit any umask parameters as follows:

   ```
   readonly TMOUT=900
   export TMOUT
   ```

   Note that setting the value to read-only prevents unwanted modification during runtime.

Ensure /tmp is configured

Create a separate file system for `/tmp` with restrictive mount options like `noexec` to block execution of malicious scripts. Use `tmpfs` or a dedicated partition for `/tmp` to mitigate risks from hardlink exploitation and unauthorized script execution.

Configure `/etc/fstab` appropriately. For example:

```
tmpfs  /tmp  tmpfs defaults,rw,nosuid,nodev,noexec,relatime  0 0
```

Alternatively, you can do the following:

1. Enable `systemd` `/tmp` mounting by running:

```
# systemctl unmask tmp.mount
# systemctl enable tmp.mount
```

2. Edit `/etc/systemd/system/local-fs.target.wants/tmp.mount` to configure the `/tmp` mount:

```
[Mount]
What=tmpfs
Where=/tmp
Type=tmpfs
Options=mode=1777,strictatime,noexec,nodev,nosuid
```

Ensure NTP is configured and in use

The redundant setup of NTP servers prevents time discrepancies and ensures accurate timekeeping across networks, servers, or network devices. It is highly recommended to ensure redundancy in NTP sources to avoid discrepancies.

Configure at least two NTP servers in your network and point `chronyd` on each cluster node to them:

1. Remove the default pool configuration option from `/etc/chronyd.conf` and add the server option with local NTP servers:

```
#pool pool.ntp.org iburst
server ntp1.local.example.com
server ntp2.local.example.com
```

2. Restart `chronyd`:

```
# systemctl restart chronyd.service
```

Install updates regularly

Apply regular system updates and security patches to ensure that servers are running stable, up-to-date, and secure software. This helps protect against known exploits or exploitation techniques targeting outdated or vulnerable software and possibly against zero-day vulnerabilities. Timely updates are crucial for maintaining high security.

To perform a cluster update, refer to Installing updates in the Administrator Guide.

Network security

Use static DNS servers

It is recommended to use trusted static caching DNS servers to offload risks associated with untrusted DNS servers obtained through DHCP. Trusted servers ensure reliable name resolution, mitigate security vulnerabilities, and prevent DNS hijacking attacks. Ensure that at least two or three reliable DNS servers are configured, and avoid using public ones.

To configure static DNS servers, refer to Adding external DNS servers in the Administrator Guide.

Configure inbound/outbound restrictions

Block unnecessary outbound and inbound traffic to reduce the risk of unauthorized communications, such as botnets, email spam, etc. It is a good practice to control outbound SMTP traffic (on TCP port 25) to prevent spam or phishing emails sent by botnets, malware, etc. Additionally, restrict inbound RPC, DNS, and NETBIOS traffic to protect against lateral movement attacks or amplification attacks that could lead to DDoS attacks and cloud outages. It is recommended to use a firewall to allow legitimate inbound and outbound traffic flows.

- To configure inbound firewall rules, refer to Configuring inbound firewall rules in the Administrator Guide.
- To configure outbound firewall rules, refer to Configuring outbound firewall rules in the Administrator Guide.

Implement API allow lists

API allowlisting is mandatory to restrict external network access to critical system components. This can be implemented in various ways, with firewalls being the preferred method. It is recommended to use IP-based access controls or allow lists. This will help reduce exposure to external attacks, unauthorized access, data breaches, and service exploitation.

Implement API allowlisting:

- Mandatory for Keystone, Nova, and Barbican due to service sensitivity and exposure risk
- Recommended for other services, if applicable

Implement DNSSEC

Implement Domain Name System Security Extensions (DNSSEC) to digitally sign DNS records and protect their integrity. This prevents DNS spoofing and man-in-the-middle attacks, ensuring secure communication between clients and their environments.

To check your domain DNSSEC setup, you can use the DNS Debugger tool or similar.

Ensure ICMP redirects are not accepted

Configure the system to ignore ICMP redirect messages unless absolutely necessary or disable ICMP redirects entirely to prevent attackers from maliciously altering the system's routing table. This ensures traffic is not redirected to attacker-controlled routes.

1. In the `/etc/sysctl.conf` or `/etc/sysctl.d/*` file, set the following parameters:

```
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0
```

2. Set the active kernel parameters by running:

```
# sysctl -w net.ipv4.conf.all.accept_redirects=0
# sysctl -w net.ipv4.conf.default.accept_redirects=0
# sysctl -w net.ipv6.conf.all.accept_redirects=0
# sysctl -w net.ipv6.conf.default.accept_redirects=0
# sysctl -w net.ipv4.route.flush=1
# sysctl -w net.ipv6.route.flush=1
```

Ensure secure ICMP redirects are not accepted

Protect the system from receiving updates to its routing table from known, but potentially compromised, gateways by setting `net.ipv4.conf.all.secure_redirects` to 0. This measure ensures that the system maintains control over its routing configurations and reduces the likelihood of man-in-the-middle attacks.

1. In the `/etc/sysctl.conf` or `/etc/sysctl.d/*` file, set the following parameters:

```
net.ipv4.conf.all.secure_redirects = 0
net.ipv4.conf.default.secure_redirects = 0
```

2. Set the active kernel parameters by running:

```
# sysctl -w net.ipv4.conf.all.secure_redirects=0
# sysctl -w net.ipv4.conf.default.secure_redirects=0
# sysctl -w net.ipv4.route.flush=1
```

Ensure IPv6 router advertisements are not accepted

Disable router advertisements to prevent traffic from being routed through malicious or compromised devices. Configure trusted routes manually, such as a default route to a verified router, to ensure traffic follows secure paths. This approach prevents attacks that exploit automatic routing updates, such as those using rogue routers to capture sensitive data.

1. In the `/etc/sysctl.conf` or `/etc/sysctl.d/*` file, set the following parameters:

```
net.ipv6.conf.all.accept_ra = 0
net.ipv6.conf.default.accept_ra = 0
```

2. Set the active kernel parameters by running:

```
# sysctl -w net.ipv6.conf.all.accept_ra=0
# sysctl -w net.ipv6.conf.default.accept_ra=0
# sysctl -w net.ipv6.route.flush=1
```

Ensure suspicious packets are logged

Enable logging of suspicious network packets to detect spoofed packets, unusual traffic patterns, or attempts to exploit vulnerabilities. Regularly review logs to identify potential network threats and mitigate them early.

1. In the `/etc/sysctl.conf` or `/etc/sysctl.d/*` file, set the following parameters:

```
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.default.log_martians = 1
```

2. Set the active kernel parameters by running:

```
# sysctl -w net.ipv4.conf.all.log_martians=1
# sysctl -w net.ipv4.conf.default.log_martians=1
# sysctl -w net.ipv4.route.flush=1
```

Service security

Ensure SNMP is not enabled unless absolutely necessary

Avoid using SNMPv1, which transmits data in the clear text and does not require authentication to execute commands. It is recommended to switch to SNMPv3, as it offers improved security features, including authentication and encryption, which protect sensitive information from unauthorized access.

To disable `snmpd`, run:

```
# systemctl --now disable snmpd
```

To switch to SNMPv3, do the following:

1. In `/etc/snmp/snmpd.conf`, add the following lines:

   ```
   com2sec snmpv3test localhost    dummycontext
   com2sec snmpv3test pan51        dummycontext
   group snmpv3group       usm     snmpv3test
   access snmpv3group        ""      usm       priv    exact  all     all   all
   rouser rousername
   ```

2. Create an SNMPv3 user:

   ```
   # systemctl stop snmpd.service
   # net-snmp-create-v3-user
   # systemctl start snmpd.service
   ```

Ensure the default SNMP community is changed

If SNMP is required, configure it securely by restricting access to private network interfaces and changing default community strings.

To change the default community strings (which act like passwords):

1. Check the `/etc/snmp/snmpd.conf` file for configured communities (the `rwcommunity` and `rocommunity` options) and change them. For example:

   ```
   rocommunity somesecom
   ```

2. Restart the `snmpd` service:

   ```
   # systemctl restart snmpd.service
   ```

Ensure RPC is not enabled unless absolutely necessary

If the system does not require RPC-based services, it is recommended to disable `rpcbind` to reduce the remote attack surface.

To disable `rpcbind`, run:

```
# systemctl --now disable rpcbind
```

Ensure Telnet is not installed

Replace Telnet with SSH where possible for encrypted and secure remote communications. The SSH package provides an encrypted session and stronger security communication.

To uninstall Telnet, run:

```
# dnf remove telnet
```

Data encryption

Encryption at rest

Encrypt data at rest to protect sensitive information from unauthorized access, theft, or breaches. Use AES-256 encryption to meet compliance standards while maintaining a balance between security and performance.

To enable data encryption at rest, refer to Enabling data encryption in the Administrator Guide.

Encryption in transit

Encrypt data in transit by enabling encrypted communication channels with SSL certificates. This ensures that any sensitive information transmitted between the client and server, cluster nodes, and management systems is protected from interception or tampering.

To enable data encryption in transit, refer to Configuring data-in-transit encryption in the Administrator Guide.

Monitoring and logging

Configure remote syslog logging

Store logs on a remote host to protect log integrity from tampering or loss and ensure centralized storage. Enable automated alerts for suspicious activity.

1. On each cluster node, prepare a configuration in the `/etc/rsyslog.d/XX-remotelog` file with the following content:

   ```
   *.* action(type="omfwd" queue.filename="fwdallfile" queue.maxdiskspace="1g"
   queue.saveonshutdown="on" queue.type="LinkedList" action.resumeRetryCount="-1"
   target="syslog.example.com" port="514" protocol="tcp")
   ```

2. Restart `syslogd`:

   ```
   # systemctl restart rsyslog.service
   ```

Configure journald to send logs to rsyslog

Use `rsyslog` for consistent and reliable remote log collection. This standardizes `journald` log exports and simplifies long-term retention and analysis.

In the `/etc/systemd/journald.conf` file, add the following line:

```
ForwardToSyslog=yes
```