Table of Contents

1. About This Document .............................................................. 7
   1.1. What Is PrivX ............................................................... 7
   1.2. Intended Audience ....................................................... 8
   1.3. Deployment Workflow ............................................... 8
   1.4. Terminology ............................................................... 9
   1.5. Documentation Conventions ....................................... 10
2. Preparing for Deployment .................................................... 13
   2.1. Prerequisites for Installation ...................................... 13
       2.1.1. GDPR Compliance ............................................. 13
       2.1.2. Mandatory Components .................................... 13
       2.1.3. Optional Components ....................................... 14
   2.2. Expected System Performance ................................... 16
   2.3. Features Dependent on License ................................... 16
   2.4. NTP Clock Synchronization ....................................... 17
   2.5. High-Availability Installation Requirements .................. 17
   2.6. HSM Integration ....................................................... 18
       2.6.1. Keys Stored in HSM ......................................... 18
       2.6.2. Database/Filesystem Keys Encrypted by HSM Integration ........................................ 19
   2.7. Supported Key-Exchange Algorithms ......................... 19
   2.8. Product Limitations .................................................. 19
3. Setting Up PrivX ............................................................... 21
   3.1. PrivX Server Setup ................................................... 21
   3.2. Setting Up Optional Components ............................... 23
       3.2.1. PrivX Extender Setup ...................................... 23
       3.2.2. PrivX Carrier and Web Proxy Setup .................... 24
       3.2.3. PrivX Agent Setup .......................................... 27
   3.3. PrivX License Management ....................................... 32
       3.3.1. Online Licensing ............................................. 32
       3.3.2. Deactivating PrivX Licenses ............................... 33
   3.4. High-Availability Deployment ................................... 33
   3.5. Backing Up and Restoring PrivX Deployments ............. 35
       3.5.1. Manual PrivX-Server Backup and Restore ........... 36
   3.6. Upgrading PrivX Deployment ..................................... 37
<table>
<thead>
<tr>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.6.1. Single-Server-Deployment Upgrade</td>
</tr>
<tr>
<td>3.6.2. High-Availability-Deployment Upgrade</td>
</tr>
<tr>
<td>3.6.3. Upgrading Optional Components</td>
</tr>
<tr>
<td>3.7. Uninstalling PrivX</td>
</tr>
<tr>
<td>4. Getting Started with PrivX</td>
</tr>
<tr>
<td>4.1. Creating Local Users and Roles</td>
</tr>
<tr>
<td>4.2. Adding Target Hosts</td>
</tr>
<tr>
<td>4.3. Connecting to Target Hosts</td>
</tr>
<tr>
<td>4.4. Enabling Certificate Authentication for OpenSSH Connections</td>
</tr>
<tr>
<td>4.5. Importing Users from Directories</td>
</tr>
<tr>
<td>4.6. Importing Hosts from Directories</td>
</tr>
<tr>
<td>5. Advanced Configuration</td>
</tr>
<tr>
<td>5.1. SSL/TLS Security</td>
</tr>
<tr>
<td>5.1.1. Setting Up a Trusted Server Certificate</td>
</tr>
<tr>
<td>5.1.2. Setting Up a Trusted CA Certificate</td>
</tr>
<tr>
<td>5.1.3. Allowed SSL Protocols and Ciphers for GUI Connections</td>
</tr>
<tr>
<td>5.1.4. Rotating Encryption Keys</td>
</tr>
<tr>
<td>5.2. PrivX Log Settings</td>
</tr>
<tr>
<td>5.3. PrivX-Extender Configuration</td>
</tr>
<tr>
<td>5.3.1. Configuring Extender Log Location</td>
</tr>
<tr>
<td>5.3.2. Proxying Native-Client Connections</td>
</tr>
<tr>
<td>5.4. PrivX-Carrier and Web Proxy Configuration</td>
</tr>
<tr>
<td>5.4.1. Allowing HTTP/HTTPS Targets at Unusual Ports</td>
</tr>
<tr>
<td>5.4.2. Changing Web Proxy Port Numbers</td>
</tr>
<tr>
<td>5.5. Setting Custom Instance Names</td>
</tr>
<tr>
<td>5.6. Resetting the Superuser Password</td>
</tr>
<tr>
<td>6. PrivX Users and Permissions</td>
</tr>
<tr>
<td>6.1. Adding PrivX Users</td>
</tr>
<tr>
<td>6.2. Granting Permissions for PrivX Users</td>
</tr>
<tr>
<td>6.2.1. Managing Roles</td>
</tr>
<tr>
<td>6.2.2. Requesting and Approving Memberships</td>
</tr>
<tr>
<td>6.2.3. Granting Access to Target Hosts</td>
</tr>
<tr>
<td>6.2.4. Granting Administrator Permissions</td>
</tr>
<tr>
<td>6.3. Managing Workflows</td>
</tr>
<tr>
<td>6.3.1. Enabling Email Notifications</td>
</tr>
<tr>
<td>6.4. User Configuration</td>
</tr>
<tr>
<td>6.4.1. Automatic Logout</td>
</tr>
<tr>
<td>6.4.2. Require Password Change</td>
</tr>
<tr>
<td>6.4.3. Limiting Login Rate</td>
</tr>
<tr>
<td>6.5. AD/LDAP Directory Configuration</td>
</tr>
<tr>
<td>6.5.1. Secure-Connection Setup</td>
</tr>
<tr>
<td>6.5.2. Adjusting User Matching</td>
</tr>
<tr>
<td>6.5.3. Refreshing Directory Data</td>
</tr>
<tr>
<td>6.6. OpenID-Connect Configuration</td>
</tr>
</tbody>
</table>
6.7. Advanced Authentication for PrivX Users .............................................................. 71
  6.7.1. Kerberos Authentication for PrivX Users .......................................................... 72
  6.7.2. Multi-Factor Authentication for PrivX Users ...................................................... 74
  6.7.3. Client-Certificate Authentication for PrivX Users .............................................. 75

7. Authentication Methods for Host Connections ...................................................... 77
  7.1. Supported Authentication Methods ...................................................................... 77
  7.2. Enabling Certificate-Based Authentication for SSH Connections ....................... 78
    7.2.1. Requirements for Certificate Authentication Over SSH ................................. 78
    7.2.2. Script-Based Certificate-Authentication Setup ................................................. 79
    7.2.3. Certificate-Authentication Setup via Chef ...................................................... 83
    7.2.4. Manual Certificate-Authentication Setup ........................................................ 84
  7.3. Enabling Role-Based Public-Key Authentication ................................................ 89
  7.4. Enabling Authentication with Stored Passwords .................................................. 90
  7.5. Enabling Certificate Authentication for RDP Connections ................................. 90
    7.5.1. Requirements for Certificate-Based Authentication Over RDP ..................... 90
    7.5.2. Steps for RDP Certificate-Authentication-Setup ............................................ 91
  7.6. Trusting Target-Host Identities ......................................................................... 92
    7.6.1. SSH Target-Host Authentication .................................................................... 92
    7.6.2. RDP Target-Host Authentication ................................................................... 93
  7.7. Authentication to AWS Services ...................................................................... 94
    7.7.1. Prerequisites ................................................................................................. 94
    7.7.2. PrivX Authentication to AWS Services ........................................................... 94
    7.7.3. Using AWS Services ...................................................................................... 96

8. Establishing and Managing Connections .............................................................. 97
  8.1. Setting up Known Targets .................................................................................. 97
    8.1.1. Proxying Connections to Hosts ...................................................................... 98
    8.1.2. SSH Targets ................................................................................................ 98
    8.1.3. RDP Targets ................................................................................................ 99
    8.1.4. Web Targets ................................................................................................ 99
  8.2. Connecting via the PrivX GUI ........................................................................... 101
    8.2.1. SSH GUI Features ....................................................................................... 101
    8.2.2. RDP GUI Features ...................................................................................... 102
  8.3. Importing Known Targets from Directories ...................................................... 103
    8.3.1. Removing Hosts from Directories ................................................................ 105
  8.4. SSH Connections with Native Clients .............................................................. 105
    8.4.1. Prerequisites ............................................................................................... 106
    8.4.2. Connecting with Native SSH Clients using PrivX Bastion ............................. 106
    8.4.3. Connecting with Native SSH Clients Using PrivX Agent (Unix and MacOS) ....... 108
    8.4.4. Connecting with Native SSH Clients Using PrivX Agent (Windows) ............... 109
  8.5. RDP Connections with Native Clients ............................................................. 110
  8.6. Website Access via PrivX ................................................................................ 111
  8.7. AWS CLI Connection with Native Client .......................................................... 112
  8.8. Monitoring and Managing Connections ........................................................... 112
  8.9. Restricting Users' Access to Applications in RDP Connections ............................ 112

9. Auditing .............................................................................................................. 113
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1. Viewing Audit Data</td>
<td>113</td>
</tr>
<tr>
<td>9.2. Using CEF Format for Audit Data</td>
<td>113</td>
</tr>
<tr>
<td>9.2.1. Logging CEF Audit Messages to External SIEM</td>
<td>113</td>
</tr>
<tr>
<td>9.2.2. Logging CEF Audit Messages Internally</td>
<td>114</td>
</tr>
<tr>
<td>9.3. Session Recording Setup</td>
<td>114</td>
</tr>
<tr>
<td>9.4. Log-Collector Setup</td>
<td>116</td>
</tr>
<tr>
<td>9.5. Matching Certificate-Based-Login Messages</td>
<td>117</td>
</tr>
<tr>
<td>9.6. Audit Events in PrivX</td>
<td>118</td>
</tr>
<tr>
<td><strong>10. Troubleshooting</strong></td>
<td>123</td>
</tr>
<tr>
<td>A. Example Nginx Load-Balancer Configuration</td>
<td>127</td>
</tr>
<tr>
<td>A.1. Configuration with TLS Passthrough</td>
<td>128</td>
</tr>
<tr>
<td>A.2. Configuration with TLS Termination</td>
<td>129</td>
</tr>
<tr>
<td>B. PrivX Architecture</td>
<td>131</td>
</tr>
<tr>
<td>C. PrivX Analytics</td>
<td>135</td>
</tr>
<tr>
<td>Index</td>
<td>137</td>
</tr>
</tbody>
</table>
Chapter 1 About This Document

In this Administrator Manual you will find instructions for deploying and operating PrivX.

PrivX provides you with:

- A web-based solution for accessing target hosts using SSH and RDP, removing the need for dedicated clients.
- A customizable workflow-based mechanism for requesting and granting access through roles.
- Integration with existing user and host directories.
- Just-in-time, role-based access to target hosts – no SSH keys used, no need to rotate keys – access can be revoked at any time.
- Roles can be granted within PrivX, or they can be mapped from existing user groups and/or roles.

1.1 What Is PrivX

Problems with Common PAM Solutions

Traditional ways of provisioning access to servers rely on outdated methods and processes that are not compatible with the cloud. Additionally, credentials that provide access for privileged users are often left unprotected on client endpoints susceptible to theft or misuse.

Traditional Privileged Access Management (PAM) is based on password vaulting and automatic password rotation with special agent/client software to be installed on network machines. Password vaults become single points of failure, they are also expensive and difficult to deploy. Similarly, agents and clients need to be patched and kept up to date. All of this makes deploying, maintaining, and using traditional PAM expensive and complex.

The Solution: PrivX

PrivX uses short-term credentials to provide just-in-time access to servers. Access is granted based on the user’s role. Provisioned credentials are created on demand, only valid for a short time, and never stored to disk.
Improve security

Access to endpoints is provisioned using on-demand short-lived certificates that are valid for only a few minutes and never written to disk nor exposed to end-users. This completely eliminates the risk of credential theft, removing the greatest security risk in privileged access management.

Reduce costs

PrivX works using existing SSH clients and servers, so there is no need to add or replace components of the SSH infrastructure. The lack of dependence on external components translates to minimal disruptions to network infrastructure, and reduced maintenance costs.

Using short-lifetime access certificates provided by PrivX eliminates the need for certificate revocation or key rotation.

1.2 Intended Audience

This Administrator Manual is intended for technical personnel responsible for installing software to the company network, those responsible for security software, and personnel maintaining and auditing secure access in the corporate network.

Readers are expected to be familiar with the operating systems on which the PrivX will be installed.

Readers should also be familiar with the Tectia Server and any Secure Shell software used in their environment.

1.3 Deployment Workflow

Deploying PrivX involves the following steps:

1. Set up PrivX servers (Chapter 2 and Chapter 3).

2. Onboard users. Define roles for granting access privileges (Chapter 6).

3. Set up passwordless authentication methods for target hosts (Chapter 7).

   To prevent having to reconfigure hosts later, we strongly recommend you set up the necessary roles in your environment before configuring authentication methods on target hosts.

4. Use PrivX to connect to target hosts (Chapter 8).

   Tip

   For an introduction to the PrivX main features, see the examples in Chapter 4.
1.4 Terminology

The following terms are used throughout the documentation.

authorizer
Authorizer creates certificates with user’s roles as needed for users connecting to target hosts.

certificate
A certificate is a signed document that binds together the trusted issuer, and subject information such as public key, subject name, list of principals (role memberships), and information about access restrictions. Certificates on PrivX are short term, issued by the Authorizer, and verifiable using the Authorizer public key.

directory (in UI)
A directory in the PrivX UI refers to a source of user accounts, for example, an AD/LDAP directory.

host service
Services by which PrivX users establish connections: SSH servers, RDP servers, and Web-login pages.

host store
Host stores save host information, such as addresses, SSH/RDP services, and target-user-to-role mappings. Host stores also import hosts from existing directories.

known host
When the connection information of a host is stored in PrivX, that host is considered a known host. Known hosts enable PrivX features including passwordless and certificate-based connections.

local user directory
Local user directory provides an easy way to create local users for authentication and role mapping. Authentication is done via username or email, and a password.

native client
SSH and RDP clients supported by users’ operating systems, such as OpenSSH client (ssh) or Remote Desktop client (mstsc).

OAuth2 service
OAuth2 service provides an authentication mechanism for a user to provide a username and password, and provides the given credentials against SSH PrivX Local User Store, and authentication providers, such as LDAP and AD.

principal
Principals are unique identities used in OpenSSH certificates, such as user names, or the UUIDs of PrivX roles.

PrivX deployment
PrivX servers using the same database. A PrivX deployment consists of one or more PrivX servers.
PrivX user

Any user account that is available via PrivX. Includes both PrivX local users, and users from AD/LDAP directories users that have been added to PrivX.

role

PrivX provides role-based access permissions: For a user to receive access permissions, they must be assigned to a role. Each role in the system has a unique principal (UUID) that represents the role in certificates and target host configurations.

role store

In PrivX, role store integrates against user directories and identity providers, for example, LDAP and AD. Role store contains rules which are evaluated to automatically map existing LDAP/AD user groups and roles into PrivX roles which are in turn used to access target hosts.

target

An identity and a host service that PrivX user(s) may connect to (for example, connecting to a host as root using SSH).

PrivX regards each unique identity-service combination as a separate target. For example, connecting to a host as root using SSH is different from connecting to the host as root using RDP.

See also host service, target host and target user.

target host

Any destination host for a connection that has been authenticated/authorized using PrivX. In other words, any host to which access is granted using PrivX.

target user

The identity that PrivX users assume on target hosts.

1.5 Documentation Conventions

The following typographical conventions are used in SSH Communications Security documentation:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Usage</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bold</strong></td>
<td>Menus, commands, GUI elements, strong emphasis</td>
<td>Click Apply or OK.</td>
</tr>
<tr>
<td>→</td>
<td>Series of menu selections</td>
<td>Select File → Save</td>
</tr>
<tr>
<td>Monospace</td>
<td>Filenames, directories, URLs etc.</td>
<td>Refer to readme.txt</td>
</tr>
<tr>
<td><em>Italics</em></td>
<td>Placeholder values in examples, reference to other documents or products, emphasis</td>
<td>See the Tectia SSH Client User Manual</td>
</tr>
<tr>
<td>#</td>
<td>In front of a command, # indicates that the command is run as a privileged user (root).</td>
<td># rpm --install package.rpm</td>
</tr>
<tr>
<td>Convention</td>
<td>Usage</td>
<td>Example</td>
</tr>
<tr>
<td>------------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>$</td>
<td>In front of a command, $ indicates that the command is run as a non-privileged user.</td>
<td>$ sshg3 user@host</td>
</tr>
<tr>
<td><strong>OS#, OS$</strong></td>
<td>In front of a command, OS# or OS$ indicates that the command is specific for certain operating systems. Multiple operating systems are separated with a /</td>
<td>SUSE$ sudo zypper update  RedHat/CentOS# yum upgrade</td>
</tr>
<tr>
<td>\</td>
<td>At the end of a line in a command, \ indicates that the command continues on the next line, but there was not enough space to show it on one line.</td>
<td>$ ssh-keygen-g3 -t rsa \ -F -c mykey</td>
</tr>
</tbody>
</table>

Table 1.1. Documentation conventions

**Note**

A Note indicates neutral or positive information that emphasizes or supplements important points of the main text. A Note supplies information that may apply only in special cases (for example, memory limitations, equipment configurations, or specific versions of a program).

**Caution**

A Caution advises users that failure to take or to avoid a specified action could result in loss of data.
Chapter 2 Preparing for Deployment

This chapter describes the prerequisites of PrivX deployment.

2.1 Prerequisites for Installation

This section describes the system requirements and specifications for PrivX-system components.

2.1.1 GDPR Compliance

Please note that as a PrivX handles user data, that data will be classified as personal information, or Personally Identifiable Information. You must ensure your GDPR compliance and inform your users of handling of their data.

2.1.2 Mandatory Components

A PrivX deployment must include at least one PrivX server for running PrivX services.

<table>
<thead>
<tr>
<th>Server</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>System Configuration</td>
<td>4 GB RAM, 2-core CPU, and 15 GB storage for &lt; 10k users</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 GB RAM, 8-core CPU, and 100 GB storage for &lt; 100k users</td>
<td></td>
</tr>
<tr>
<td>Supported architecture</td>
<td>x86-64</td>
<td></td>
</tr>
<tr>
<td>Supported operating systems</td>
<td>Red Hat Enterprise Linux 7.4 or later 7.x, 8.x (x86-64)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CentOS 7.4 or later 7.x, 8.x (x86-64)</td>
<td></td>
</tr>
<tr>
<td>Supported databases</td>
<td>Local or external PostgreSQL and Redis</td>
<td></td>
</tr>
<tr>
<td>Network Connectivity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internet connectivity required</td>
<td>• For installing dependant libraries during installation and upgrades</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• For PrivX-license activation and verification. License server address is:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>184.106.60.185:443</td>
<td></td>
</tr>
</tbody>
</table>
Default network ports in use

<table>
<thead>
<tr>
<th>Service</th>
<th>Port(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web UI</td>
<td>443 in</td>
</tr>
<tr>
<td>Client-certificate authentication</td>
<td>8443 in</td>
</tr>
<tr>
<td>SSH sessions</td>
<td>22 out</td>
</tr>
<tr>
<td>SSH Bastion</td>
<td>2222 in</td>
</tr>
<tr>
<td>RDP sessions</td>
<td>3389 out</td>
</tr>
<tr>
<td>RDP Bastion</td>
<td>3389 in</td>
</tr>
<tr>
<td>Email notifications</td>
<td>25, 465, 587 out</td>
</tr>
<tr>
<td>CRL lists for RDP connections</td>
<td>80 in</td>
</tr>
<tr>
<td>DNS</td>
<td>53 out</td>
</tr>
<tr>
<td>External PostgreSQL</td>
<td>5432 out</td>
</tr>
<tr>
<td>External Redis</td>
<td>6379 out</td>
</tr>
<tr>
<td>NTP</td>
<td>123 out</td>
</tr>
</tbody>
</table>

**Target-host Authentication**

**Supported OpenSSH versions**
- Certificate-based authentication - OpenSSH 6.9 or later, see Section 7.2
- Other authentication methods - OpenSSH 5.6 or later

**Supported platforms for Windows Domain Controller and Certificate Authority Server**
- Windows Server 2012 R2, 2016, 2019 (with the latest service packs and updates)

**Client Experience**

**Supported browsers**
- Latest versions of:
  - Firefox
  - Chrome
  - Safari
  - Edge

**System Security**

**HSM support**
- SafeNet Network HSM:
  - Luna SA 5

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Other applications and users with access to PrivX hosts can gain potentially sensitive information from unprotected system memory. For best security we strongly recommend running PrivX on dedicated hosts.</td>
</tr>
</tbody>
</table>

**2.1.3 Optional Components**

This section describes the system requirements and specifications for optional PrivX components:
PrivX Extenders proxy connections to target hosts. They are needed for connecting to hosts not directly accessible from PrivX servers.

<table>
<thead>
<tr>
<th><strong>Extender</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System Configuration</strong></td>
</tr>
<tr>
<td>4 GB RAM, 2-core CPU, and 15 GB storage</td>
</tr>
<tr>
<td><strong>Supported operating systems</strong></td>
</tr>
<tr>
<td>Red Hat Enterprise Linux 7.4 or later 7.x, 8.x (x86-64)</td>
</tr>
<tr>
<td>CentOS 7.4 or later 7.x, 8.x (x86-64)</td>
</tr>
<tr>
<td><strong>Network Connectivity</strong></td>
</tr>
<tr>
<td>Internet connectivity required for installing dependant libraries during installation and upgrades</td>
</tr>
<tr>
<td>Default network ports in use</td>
</tr>
<tr>
<td>PrivX connection: Port 443 out</td>
</tr>
<tr>
<td>SSH sessions to hosts: Port 22 out</td>
</tr>
<tr>
<td>DNS: Port 53 out</td>
</tr>
<tr>
<td>RDP sessions to hosts: Port 3389 out</td>
</tr>
<tr>
<td>Host-deployment listener: Port 8443 in</td>
</tr>
</tbody>
</table>

Table 2.2. PrivX-Extender Requirements and Specifications

PrivX Carriers provide web functionality. Needed if you want to connect to HTTP/HTTPS targets via PrivX.

<table>
<thead>
<tr>
<th><strong>Carrier</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System Configuration</strong></td>
</tr>
<tr>
<td>64 GB RAM, 16-core CPU, and 100 GB storage for &lt; 50 concurrent web connections</td>
</tr>
<tr>
<td><strong>Supported operating systems</strong></td>
</tr>
<tr>
<td>Red Hat Enterprise Linux 7.4 or later 7.x, 8.x (x86-64)</td>
</tr>
<tr>
<td>CentOS 7.4 or later 7.x, 8.x (x86-64)</td>
</tr>
<tr>
<td><strong>Network Connectivity</strong></td>
</tr>
<tr>
<td>Internet connectivity required for installing dependant libraries during installation and upgrades</td>
</tr>
<tr>
<td>Default network ports in use</td>
</tr>
<tr>
<td>PrivX connection: Port 443 out</td>
</tr>
<tr>
<td>Web-proxy connection: Ports 18080, 18443, and 18444 out</td>
</tr>
</tbody>
</table>

Table 2.3. PrivX-Carrier Requirements and Specifications

Similar to carriers, PrivX Web Proxies also provide web functionality. Needed if you want to connect to HTTP/HTTPS targets via PrivX.

<table>
<thead>
<tr>
<th><strong>Web Proxy</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System Configuration</strong></td>
</tr>
<tr>
<td>4 GB RAM, 2-core CPU, and 15 GB storage for &lt; 50 concurrent web connections</td>
</tr>
<tr>
<td><strong>Supported operating systems</strong></td>
</tr>
<tr>
<td>Red Hat Enterprise Linux 7.4 or later 7.x, 8.x (x86-64)</td>
</tr>
</tbody>
</table>
### Chapter 2 Preparing for Deployment

** CentOS 7.4 or later 7.x, 8.x (x86-64)  

#### Network Connectivity

<table>
<thead>
<tr>
<th>Internet connectivity required</th>
<th>For installing dependant libraries during installation and upgrades</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default network ports in use</td>
<td>Carrier connection: Ports 18080, 18443, and 18444 in</td>
</tr>
<tr>
<td></td>
<td>Target-host connections: Port 80 and 443 out</td>
</tr>
<tr>
<td></td>
<td>PrivX connection: Port 443 out</td>
</tr>
<tr>
<td></td>
<td>ICAP-listener: Port 1344 in</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Supported login methods for autofill</th>
<th>Multi-part form</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JSON</td>
</tr>
<tr>
<td></td>
<td>XML</td>
</tr>
</tbody>
</table>

Table 2.4. PrivX-Web-Proxy Requirements and Specifications

#### 2.2 Expected System Performance

A PrivX server that satisfies or exceeds the production requirements is expected to support:

- 100 000 PrivX users total, with 700 concurrent users.
- 25 000 target hosts total, with:
  - Up to 1000 hosts added/deployed concurrently.
  - 20 000 target hosts scanned in 2 minutes.
  - Up to 200 concurrent RDP connections for performing typical user operations. Graphically intensive sessions (including video streaming) may reduce the number of supported concurrent connections.

**Note**

The PrivX microservice architecture supports multiprocessing and benefits from using multiple CPUs or multiple CPU cores.

Reserve enough space for the log data generated by PrivX. Also monitor the log-data growth periodically. In large deployments, PrivX may generate a considerable amount of log data over time. You may configure the PrivX machine to write its log data to an external logging server.

#### 2.3 Features Dependent on License

The enabled features and limitations can be viewed on the **Settings → License** page.

The maximum amount of concurrent SSH, RDP, and HTTPS connections depends on the type of PrivX license. Connections exceeding the maximum allowed connections are disconnected.

The standard PrivX license gives you all the PrivX features except the ones listed below:

- **PrivX Extender:** For more information, see Section 8.1.1.
2.4 NTP Clock Synchronization

We strongly recommend enabling NTP clock synchronization for the system. Otherwise, please verify the system time and date manually before continuing with PrivX installation. We also recommend enabling time-server authentication for NTP connections.

The short-term user certificates issued by PrivX may be valid at incorrect times if system times are not synchronized correctly. Due to the just-in-time nature of the certificates issued by PrivX, even deviations of just a few minutes may cause authorizations to fail.

2.5 High-Availability Installation Requirements

If you plan to have multiple PrivX servers in your deployment (for high-availability and/or load balancing), you must set up an external PostgreSQL database and an external Redis server. You may also need to set up a load balancer to distribute connections to PrivX servers.

- Set up a PostgreSQL-database instance and a Redis-server instance. We strongly recommend employing dedicated instances for PrivX.

  The PostgreSQL superuser (typically `postgres`) must have a valid password: During initial setup PrivX requires superuser permissions, for creating a PrivX database and a PrivX database user.

  Access to the Redis server may be password-protected.

- PrivX servers require access to the PostgreSQL database and the Redis server.

  For example with PostgreSQL on Unix, you will need to edit the `pg_hba.conf`, and insert entries similar to the following:

  ```
  hostssl all all <privx_server_ip> md5
  ```

- External-database connections must be SSL-protected:

  - Enable SSL mode in your PostgreSQL configuration (`ssl = true`).

  - The PostgreSQL server must be configured with a server certificate where the `SubjectAltName` specifies the DNS and IP address(es) of the server.

  PrivX servers should also be configured to trust the PostgreSQL-server certificate: On each PrivX server, add the PostgreSQL-server CA chain to the system trust anchors. For more information about adding trust anchors to PrivX servers, see Section 6.5.1.
• Select a load-balancer platform that satisfies the following requirements:

• The requests from a PrivX session must be handled by one PrivX server; for example the load balancer could be configured to use either IP hashing or sticky sessions.

• If you need certificate authentication to PrivX, or SSH/RDP-Bastion connectivity, the load balancer must support TCP-level load balancing.

A possible load-balancer platform is Nginx. For example load-balancer configurations on Nginx, see Appendix A.

2.6 HSM Integration

For added security, the PrivX can be integrated with a Hardware Security Module (HSM). This allows storing cryptographic keys on HSM, and encrypting database/filesystem keys using a secret from HSM.

PrivX integrates to HSM providers using PKCS #11. For more information about setting up PrivX with HSM, see the Advanced Deployment articles from https://help.ssh.com/

2.6.1 Keys Stored in HSM

The following types of keys may be stored in HSM.

Asymmetric keys

• CA for issuing just-in-time certificates when users connect using certificate authentication.

• CA for signing server certificates.

Symmetric keys

• Master key for encoding/decoding session recordings.

• Session-authentication keys.

• Keys for encrypting user and role data.

To store the default CA keys in HSM, your HSM must support the following key-pair-generation mechanisms:

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Mechanisms</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA keys</td>
<td>CKM_RSA_PKCS_KEY_PAIR_GEN</td>
</tr>
<tr>
<td></td>
<td>CKM_RSA_PKCS</td>
</tr>
<tr>
<td>Symmetric keys</td>
<td>CKM GENERIC_SECRET_KEY_GEN</td>
</tr>
<tr>
<td></td>
<td>CKM_SHA_1_HMAC</td>
</tr>
<tr>
<td></td>
<td>CKM_SHA* HMAC (256/384/512)</td>
</tr>
<tr>
<td></td>
<td>CKM_AES_KEY_GEN</td>
</tr>
</tbody>
</table>
### Table 2.5. Required key-generation mechanisms for HSM storage

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Mechanisms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CKM_AES_GCM</td>
</tr>
</tbody>
</table>

Keys not supported by the HSM are stored in the PrivX database/filesystem, but encrypted using the PKCS #11 instance secret located on the HSM.

#### 2.6.2 Database/Filesystem Keys Encrypted by HSM Integration

Authentication-signing secrets and passphrases are stored in the PrivX database/filesystem. However when HSM integration is enabled, such keys are encrypted using the PKCS #11 instance secret located on the HSM.

#### 2.7 Supported Key-Exchange Algorithms

- curve25519-sha256@libssh.org
- ecdh-sha2-nistp256
- ecdh-sha2-nistp384
- ecdh-sha2-nistp521
- diffie-hellman-group14-shal

#### 2.8 Product Limitations

- Key-combinations using keys unavailable to UK or US keyboards (such as `CTRL+Ä`) do not work on Edge browsers.
- You cannot transfer folders through file transfer.
- `Ctrl+W` key-combination closes the open tab on Firefox.
Chapter 3 Setting Up PrivX

This chapter provides basic instructions for setting up and maintaining PrivX components.

To get started with PrivX you will need to set up at least one PrivX server as described in Section 3.1.

3.1 PrivX Server Setup

PrivX servers provide PrivX services, such as the PrivX GUI and certificate-based authentication services.

To set up a PrivX server:

1. Set up repositories for downloading PrivX packages and dependencies.

   Add the EPEL repository:

   • On Red Hat 7:
     
     ```bash
     # yum install \\
     ```

   • On Red Hat 8:
     
     ```bash
     # yum install \\
     ```

   • On CentOS 7 and 8:
     
     ```bash
     # yum install epel-release
     ```

   Then add the PrivX repository:

   • On Red Hat or CentOS 7:
     
     ```bash
     # rpm --import https://product-repository.ssh.com/info.fi-ssh.com-pubkey.asc
     # curl https://product-repository.ssh.com/ssh-products.repo \\
     -o /etc/yum.repos.d/ssh-products.repo
     ```

   • On Red Hat or CentOS 8:
2. Install the latest PrivX packages with:

```bash
# rpm --import https://product-repository.ssh.com/info.fi-ssh.com-pubkey.asc
# curl https://product-repository.ssh.com/rhel8/ssh-products.repo \
-o /etc/yum.repos.d/ssh-products.repo
```

```
# yum install PrivX
```

The application binaries are installed to the `/opt/privx/bin/` directory. The configuration is installed to `/opt/privx/etc/` directory. Utility scripts are found in the `/opt/privx/scripts/` directory.

3. To automate PrivX configuration, you may optionally define post-installation settings using environment variables. The post-installation script skips prompting for any settings found in environment variables.

Environment variables for the post-installation script are listed in:

```
/opt/privx/scripts/postinstall_env
```

Export the environment variables with:

```
# source /opt/privx/scripts/postinstall_env
```

4. Run the post-installation script to configure the server:

```
# /opt/privx/scripts/postinstall.sh
```

Unless specified in environment variables, you will be prompted for:

- PKCS #11-keyvault settings. Enable and configure this only if using external HSM. For additional HSM-setup instructions, see the Advanced Deployment articles from https://help.ssh.com/
- DNS and IP address(es) of the server.
- Database settings (if using an external database):
  - Address of the database in IP or DNS-name format. For example: `database.example.com`
  - Arbitrary name for the PrivX database.
  - Arbitrary name and password for the PrivX-database user.
  - Password of the PostgreSQL user `postgres`.
  - Address and password to the external Redis server (if using an external database).
- Credentials for the initial `superuser` account.

5. Install the PrivX license to enable product functionality. You can do this by browsing to the PrivX GUI located at (replace `privx.example.com` with the DNS or IP address of the PrivX server):

```
https://privx.example.com/
```

Log in using the `superuser` credentials.
In the GUI, navigate to Settings → License in the PrivX UI, and provide your license code under Online license update. Click Update, PrivX automatically contacts the license server to obtain and install your license.

**Note**

PrivX licenses provide a limited number of activations. When deactivating PrivX servers, also remember to free up activations according to instructions in Section 3.3.2.

Unlicensed product does not allow adding hosts nor establishing SSH or RDP connections. For your license to be activated, ensure that the system time is set correctly.

You have now set up a PrivX server.

6. You may verify the status of the PrivX microservices on the status page, located at the following address (replace privx.example.com with the DNS or IP address of the PrivX server):

   https://privx.example.com/status.html

**Note**

Particularly in large production deployments, PrivX may accumulate large amounts of log data over time. We recommend that you configure log rotation for syslog and the microservice logs to prevent PrivX servers from running out of disk space.

### 3.2 Setting Up Optional Components

This section provides setup instructions for optional PrivX components.

#### 3.2.1 PrivX Extender Setup

PrivX Extenders relay host connections, allowing connections to target hosts that are inaccessible from PrivX servers.

To set up PrivX Extender:

1. Create a PrivX Extender.

   On Settings→Deployment→Deploy PrivX VPC/VPN Extenders, click Add Extender.

   Provide at least the Extender Name. Addresses and Subnets are optional, and should only be set after you have verified successful connections via the Extender.

   Click Save to create the Extender, and to download the Extender configuration. This file is later required for setting up the Extender host.
2. Install the PrivX Extender software on the Extender host in either of the following ways:

- Install from the SSH product repository. To do this, set up the repository and install the software package:
  
  On Red Hat or CentOS 7:
  
  ```
  # rpm --import https://product-repository.ssh.com/info.fi-ssh.com-pubkey.asc
  # curl https://product-repository.ssh.com/ssh-products.repo \
  -o /etc/yum.repos.d/ssh-products.repo
  ```

  On Red Hat or CentOS 8:
  
  ```
  # rpm --import https://product-repository.ssh.com/info.fi-ssh.com-pubkey.asc
  # curl https://product-repository.ssh.com/rhel8/ssh-products.repo \
  -o /etc/yum.repos.d/ssh-products.repo
  ```

  Run:
  
  ```
  # yum install PrivX-Extender
  ```

- To manually obtain the rpm, go to Settings→Deployment→Deploy PrivX VPC/VPN Extenders and click Download PrivX Extender:

  Then copy the rpm to the Extender host and install it with (replace /path/to/PrivX-Extender-*.*.rpm with the path to which you placed the Extender package):

  ```
  # yum install /path/to/PrivX-Extender-*.*.rpm
  ```

3. Save the Extender configuration to the following path on the Extender host:

   `/opt/privx/etc/extender-config.toml`

4. Run postinstall to complete the setup:

   ```
   # /opt/privx/scripts/extender-postinstall.sh
   ```

   The PrivX Extender is now set up. You may verify back on the Settings→Deployment→Deploy PrivX VPC/VPN Extenders page that the Status is Registered.

   If the Extender failed to register to PrivX, you may find additional troubleshooting information from the Extender host at `/var/log/privx/privx-extender.log`

5. Configure the target host(s) that are to be accessed via a PrivX Extender. For more information about targets accessible via Extenders, see Section 8.1.1.

### 3.2.2 PrivX Carrier and Web Proxy Setup

This section provides setup instructions for PrivX Carriers and Web Proxies, which allow connecting to HTTP/HTTPS targets. The high-level workflow involves:

1. Create a Carrier and Web-Proxy configuration.
2. Set up a PrivX Carrier.

3. Set up a PrivX Web Proxy.

First, to create and download the necessary configurations:

1. In the PrivX GUI navigate to **Settings→Deployment→Deploy PrivX web-access gateways**.

2. To create the configurations, click **Add Web-Access Gateway**. Provide at least the Name and Proxy Address for the configuration.

   **Note**

   The Web Proxy address must be a valid IP or DNS address without schema headers and port numbers, pointing to the address of the server where the Web Proxy will be installed to. Loopback addresses are not allowed.

3. Download the configurations (required later for setting up Carriers and Web Proxies). To do this, click next to your configuration, then click **Download Carrier Config** and **Download Proxy Config**.

To set up a PrivX Carrier:

1. Ensure the machine has access to *docker* packages by enabling the required repositories:

   - **On Red Hat 7:**
     
     ```
     # subscription-manager repos --enable rhel-7-server-extras-rpms
     ```

   - **On AWS Red-Hat-7 instances:**
     
     ```
     # yum-config-manager --enable "Red Hat Enterprise Linux Server 7 Extra(RPMs)"
     ```

   - **On CentOS 7:**
     
     ```
     # yum install epel-release
     ```

   - **On Red Hat 8 and CentOS 8:**
     
     ```
     # yum-config-manager \
     --add-repo=https://download.docker.com/linux/centos/docker-ce.repo
     ```

2. Install the Carrier package in either of the following ways:

   - If the machine has internet access, we recommend setting up and installing packages from the PrivX repository:

     - **On Red Hat or CentOS 7:**
       
       ```
       # rpm --import https://product-repository.ssh.com/info.fi-ssh.com-pubkey.asc
       # curl https://product-repository.ssh.com/ssh-products.repo \n       -o /etc/yum.repos.d/ssh-products.repo
       ```

     - **On Red Hat or CentOS 8:**
# rpm --import https://product-repository.ssh.com/info.fi-ssh.com-pubkey.asc
# curl https://product-repository.ssh.com/rhel8/ssh-products.repo \
-o /etc/yum.repos.d/ssh-products.repo

Run:

```bash
# yum install PrivX-Carrier
```

- Otherwise, you can obtain the Carrier package from https://help.ssh.com, under the *Downloads* article. Then install the package with:

```bash
# yum install PrivX-Carrier-*.rpm
```

3. Copy your Carrier-configuration file to your Carrier machine, to the following path:

```
/opt/privx/etc/carrier-config.toml
```

4. To finalize setup and register the Carrier with PrivX, run:

```bash
# /opt/privx/scripts/carrier-postinstall.sh
```

**Note**

The Carrier machine must be able to connect to port 443 on the PrivX server.

To set up a PrivX Web Proxy:

1. If installing on Red Hat or CentOS 7 or 8, enable the EPEL repository:

   - On Red Hat 7:

     ```bash
     # yum install \n     https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
     ```

   - On Red Hat 8:

     ```bash
     # yum install \n     https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
     ```

   - On CentOS 7 and 8:

     ```bash
     # yum install epel-release
     ```

2. Install the Web-Proxy package in either of the following ways:

   - If the machine has internet access, we recommend setting up and installing packages from the PrivX repository:

     ```bash
     # rpm --import https://product-repository.ssh.com/info.fi-ssh.com-pubkey.asc
     # curl https://product-repository.ssh.com/ssh-products.repo \
     -o /etc/yum.repos.d/ssh-products.repo
     ```

   - On Red Hat or CentOS 8:
# rpm --import https://product-repository.ssh.com/info.fi-ssh.com-pubkey.asc
# curl https://product-repository.ssh.com/rhel8/ssh-products.repo -o /etc/yum.repos.d/ssh-products.repo

Run:

# yum install PrivX-Web-Proxy

- Otherwise, you can obtain the Web-Proxy package from https://help.ssh.com, under the Downloads article. Then install the package with:

  # yum install PrivX-Web-Proxy-*.rpm

3. Copy the Web-Proxy configuration file to the machine, to the following location:

   /opt/privx/etc/web-proxy-config.toml

4. Allow the Carrier host to access the Web-Proxy host (ports 18080 and 18443). You will need to add an access rule to the Squid configuration on the Web-Proxy host, at /etc/squid/squid.conf (replace `<carrier_address>` with the address of your Carrier host in CIDR format):

   ...
   acl localnet src fc00::/7
   acl localnet src fe80::/10
   acl localnet src <carrier_address>
   ...

5. To finalize setup and register the Web Proxy with PrivX, run:

   # /opt/privx/scripts/web-proxy-postinstall.sh

**Note**

Make sure the Web-Proxy host is able to connect to port 443 on the PrivX server. The host running the PrivX Carrier must also be able to connect to ports 18080 and 18443 on the Web-Proxy host.

After you have the required Carrier and Web-Proxy components, add target websites as known targets. To do this, go to Settings→Hosts and Add hosts with Web-type services. For more information about configuring Web-type services, see Section 8.1.4.

For advanced configuration of PrivX Carrier and Web Proxy components, see Section 5.4

### 3.2.3 PrivX Agent Setup

This section provides instructions for installing PrivX agents, which allow PrivX users to connect using SSH native clients.

**Unix Setup**

To set up native clients against PrivX for a user on a Unix workstation:

Save the Agents.zip package to the workstation.

2. Extract and install the PrivX agents to the workstation. For example (replace Agents.zip with the path to the agent-software package):

```bash
# unzip -j Agents.zip linux-amd64/privx-* -d /usr/local/bin/
```

The example command installs the PrivX agent, the agent utility, and the agent proxy to:

```
/usr/local/bin/privx-agent-unix
/usr/local/bin/privx-agent-ctl
/usr/local/bin/privx-nc
```

The following steps assume the PrivX agent and the agent utility to be in these locations.

3. Enable PrivX agent to start automatically when the user logs into the workstation. You can do this by appending the following to the users’ .profile file:

```bash
# Start PrivX Agent unless there is already an ssh-agent running
START_PRIVX_AGENT="yes"
PRIVX_AGENT_UNIX="/usr/local/bin/privx-agent-unix"
if [ -n "$START_PRIVX_AGENT" ] && [ -z "$SSH_AUTH_SOCK" ] && [ -x "$PRIVX_AGENT_UNIX" ]; then
    PRIVX_AGENT_DIR=`mktemp -d /tmp/ssh-XXXXXXXXXX 2>/dev/null || mktemp -d -t 'mytmpdir'`
    export PRIVX_AUTH_SOCK="$PRIVX_AGENT_DIR/agent.$$"
    export PRIVX_AGENT_PROXY="$PRIVX_AGENT_DIR/proxy.$$"
    nohup $PRIVX_AGENT_UNIX -a $PRIVX_AUTH_SOCK -x $PRIVX_AGENT_PROXY -config $HOME/.privx/agent.json > /dev/null 2>&1 &
    export PRIVX_AGENT_PID=$!
    export SSH_AUTH_SOCK="$PRIVX_AUTH_SOCK"
    export SSH_AGENT_PID="$PRIVX_AGENT_PID"
    trap 'test -n "$SSH_AGENT_PID" && kill $SSH_AGENT_PID' 0
fi
```

**Note**

The provided example commands for agent startup are not run if another application sets the SSH_AUTH_SOCK variable (such as a default ssh agent), nor if the PrivX agent is installed to a custom location. You may have to adjust the agent-startup conditions to suit your environment.

The temporary-directory path /tmp/ssh-XXXXXXXXXX is used by the agent. Do not change this path.

4. Add PrivX servers to the agent. To do this, log in to the workstation as the user who will be using native clients (in order to start the agent), then run the following (replace https://privx.example.com with the address of your PrivX server):

```bash
$ privx-agent-ctl server probe https://privx.example.com
```
Unless the PrivX server certificate is trusted on the workstation, you will be presented with additional information, such as the SHA checksums of the certificate.

PrivX server 'https://privx.example.com' certificate is untrusted
Server TLS certificate:

<table>
<thead>
<tr>
<th>Issuer</th>
<th>DC=com, DC=ssh, DC=fi, DC=hel, OU=QA, CN=MSCA-PKI-TEST2-ROOT-CA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject</td>
<td>OU=PrivX Web Server, CN=privx.hel.fi.ssh.com</td>
</tr>
<tr>
<td>Serial</td>
<td>2453081971937566630710942289116460769316175891</td>
</tr>
<tr>
<td>Not Before</td>
<td>2018-03-28 09:00:15 +0000 UTC</td>
</tr>
<tr>
<td>Not After</td>
<td>2028-03-28 09:10:15 +0000 UTC</td>
</tr>
<tr>
<td>Fingerprints</td>
<td>SHA1  B5 F3 70 E2 DB 61 56 D9 AC AE 03 4D AE DC 01 4C FB D2 AB C0</td>
</tr>
<tr>
<td></td>
<td>SHA256 17 22 63 69 D1 AA 77 34 96 BD B7 90 69 83 E7 30 97 A2 78 10 \</td>
</tr>
<tr>
<td></td>
<td>73 5C 93 D0 D6 01 99 5C 04 C5 55 B2</td>
</tr>
</tbody>
</table>

Would you like to add this server to trusted servers? Y/N:

You may ensure that these match to the checksums of the PrivX server certificate, which you can obtain by running the following on the PrivX server:

```bash
# openssl x509 -noout -fingerprint -sha1 -in /etc/nginx/ssl/nginx.crt
# openssl x509 -noout -fingerprint -sha256 -in /etc/nginx/ssl/nginx.crt
```

Once you have confirmed the server identity, enter Y to add the PrivX server.

5. You may verify setup by running:

```bash
$ privx-agent-ctl status
```

This should output similarly to the following:

<table>
<thead>
<tr>
<th>PrivX SSH Agent Status</th>
<th>PrivX Server</th>
<th><a href="https://privx.example.com">https://privx.example.com</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Login status</td>
<td>logged out</td>
<td></td>
</tr>
</tbody>
</table>

**MacOS Setup**

To set up native clients against PrivX for a user on a MacOS workstation:


   Save the Agents.zip package to the workstation.

2. Extract and install the PrivX agents to the workstation. For example (replace Agents.zip with the path to the agent-software package):

   ```bash
   # unzip agents.zip darwin-amd64/*
   # cp darwin-amd64/privx-agent-unix /usr/local/bin/
   # cp darwin-amd64/privx-agent-ctl /usr/local/bin/
   # cp darwin-amd64/privx-agent-start.sh /usr/local/bin/
   # cp darwin-amd64/privx-nc /usr/local/bin/
   ```

   The example commands install the PrivX agent, the agent utility, the agent-startup script, and the agent proxy to:
Chapter 3 Setting Up PrivX

The following steps assume the PrivX agent, the agent utility, and the agent-startup script to be in these locations.

3. Enable PrivX agent to start automatically when the user logs into the workstation. To do this, first install the agent launcher with:

```
# cp darwin-amd64/com.privxagent.plist ~/Library/LaunchAgents/
# launchctl load -w ~/Library/LaunchAgents/com.privxagent.plist
# launchctl start ~/Library/LaunchAgents/com.privxagent.plist
```

Also append the following to the users’ .bash_profile file:

```
# Register SSH_AUTH_SOCK & SSH_AGENT_PID to local environment
if sock=$(launchctl getenv SSH_AUTH_SOCK); then
    export SSH_AUTH_SOCK=$sock
fi

if pid=$(launchctl getenv SSH_AGENT_PID); then
    export SSH_AGENT_PID=$pid
fi
```

Alternatively, users may manually start the PrivX agent with:

```
$ ./privx-agent-unix bash
```

4. Add PrivX servers to the agent. To do this, log in to the workstation as the user who will be using native clients (in order to start the agent), then run the following (replace https://privx.example.com with the address of your PrivX server):

```
$ privx-agent-ctl server probe https://privx.example.com
```

Unless the PrivX server certificate is trusted on the workstation, you will be presented with additional information, such as the SHA checksums of the certificate.

```
PrivX server 'https://privx.example.com' certificate is untrusted
```

```
Server TLS certificate:
Issuer        DC=com, DC=ssh, DC=fi, DC=hel, OU=QA, CN=MSCA-PKI-TEST2-ROOT-CA
Subject       OU=PrivX Web Server, CN=privx.hel.fi.ssh.com
Serial        2453081971937566630710942289116460769316175891
Not Before    2018-03-28 09:00:15 +0000 UTC
Not After     2028-03-28 09:10:15 +0000 UTC
Fingerprints
SHA1        B5 F3 70 E2 DB 61 56 D9 AC AE 03 4D AE DC 01 4C FB D2 AB C0
SHA256      17 22 63 69 D1 AA 77 34 96 BD B7 90 69 83 E7 30 97 A2 78 10 \ 
            73 5C 93 D0 D6 01 99 5C 04 C5 55 B2

Would you like to add this server to trusted servers? Y/N:
```
You may ensure that these match to the checksums of the PrivX server certificate, which you can obtain by running the following on the PrivX server:

```
# openssl x509 -noout -fingerprint -sha1 -in /etc/nginx/ssl/nginx.crt
# openssl x509 -noout -fingerprint -sha256 -in /etc/nginx/ssl/nginx.crt
```

Enter Y to add the PrivX server.

5. You may verify setup by running:

```
$ privx-agent-ctl status
```

This should output similarly to the following:

<table>
<thead>
<tr>
<th>PrivX SSH Agent Status</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PrivX Server</td>
<td><a href="https://privx.example.com">https://privx.example.com</a></td>
</tr>
<tr>
<td>Login status</td>
<td>logged out</td>
</tr>
</tbody>
</table>

**Windows Setup**

To set up native clients against PrivX for a user on a Windows workstation:

1. Ensure that the native clients on the workstation are in the system PATH.


   Save the Agents.zip package to the workstation.

3. Extract the PrivX agents to the workstation. Navigate to the extracted folder, and in its windows-386 folder, run the privx-agent-windows.msi file to install the PrivX agent.

4. Start the PrivX agent in **Start Menu→SSH Communications Security→PrivX Windows Agent**.

   **Note**

   The PrivX agent may become blocked by other agent software (such as pageant) are running. In such cases, first close any blocking software, then try starting the PrivX agent.

   **Tip**

   For easier access to the agent, you can pin the PrivX Windows Agent to **Start, taskbar, or Quick Launch Menu**.

5. Configure the agent to connect to PrivX: In the system tray, right click the PrivX icon and then select **Settings**.

   Enter the PrivX server URL. To verify the connection, click **Test**.

   You may verify the PrivX-server certificate:

   i. Click **View** to open the Certificate Properties.
ii. Check the certificate details. You do not need to install the certificate to the system.

Click OK to save your settings.

**Note**

If native-client paths change after setup, the PrivX agent has to be restarted to apply these path changes.

### 3.3 PrivX License Management

This section describes the supported licensing methods, and license deactivation.

Ensure that you have a valid PrivX license code for taking PrivX into use. For license-subscription-related enquiries, please contact SSH Communications Security support at privx-sales@ssh.com

#### 3.3.1 Online Licensing

Online licensing is the recommended way to license PrivX. This method is easy to set up, and after setup the license is automatically updated according to your subscription.

For online licensing to work, your PrivX servers must have Internet connectivity for certain operations:

- Upon initial license activation: PrivX contacts license servers to fetch your license.
- Upon license renewal: PrivX contacts license servers to automatically fetch your new license.
- Upon license expiry: PrivX contacts license servers to fetch the grace period associated to your license.

To activate a PrivX license with the online method:

1. Access the PrivX GUI and navigate to the **Settings→License** page.
2. Under the **License code** section, enter your license code, and click **Update License**.

PrivX automatically installs your license, which enables PrivX functionality according to your license subscription.

In online licensing mode, PrivX checks for license updates every 12 hours and automatically installs any new licenses found for your subscription. To trigger an immediate license-update check, click 🔄 next to the *Last refreshed* section, then click **Refresh**.

**Note**

If you opt to not send product analytics, you may disable internet connectivity while no license updates are available. For more information about analytics, see Appendix C.
Note that if PrivX servers do not have Internet access when their license expires, the license expires immediately without entering grace period.

### 3.3.2 Deactivating PrivX Licenses

PrivX licenses can be activated only a limited number of times. To free up activations you must first deactivate a license.

To deactivate a PrivX license, go to the Settings→License page. Next to Status, click ⚒, then click Deactivate.

### 3.4 High-Availability Deployment

This section provides a high-level description about the components of a high-availability (HA) deployment, along with high-level instructions about setting up a HA deployment.

For fault tolerance and load balancing, you can set up multiple PrivX servers as part of a single PrivX deployment. HA deployments consist of:

- A **load balancer** for distributing user connections to PrivX servers. The load balancer should also be responsible for disconnecting dead PrivX servers and adjusting the number of PrivX servers based on system load.

- Multiple **PrivX servers** for serving user sessions. More PrivX servers results in better performance and greater fault tolerance. In cloud-based environments, we recommend setting up automatic scaling to add and remove PrivX servers according to system load.

- **System storage** consisting of:
  - A PostgreSQL database for storing system data. For greater fault tolerance we recommend setting up database replication, and configuring automatic switchover to secondary database(s).
  - A cache used for synchronizing microservice states. We recommend setting up cache replication for greater fault tolerance.
  - External trail storage (only needed for storing session replays).
To set up a PrivX HA deployment:

1. Set up system storage. You will at least need to set up a PostgreSQL database and a Redis cache. For production environments we recommend enabling replication and automatic failover for the database and the cache.

If needed, also set up a secure NAS share used for trail storage. For additional information about trail-storage requirements, see Section 9.3.

2. Set up the first PrivX server according to the instructions in Section 3.1, with the following in mind:
   
   • When postinstall prompts for FQDN and IP addresses, provide the public addresses of the PrivX deployment: typically you should provide the user-facing FQDN and IP addresses of the load balancer (not the addresses of the PrivX server itself).

   • Configure the PrivX server to use the external PostgreSQL database and Redis server.

   • If the PostgreSQL superuser is other than postgres, define the actual PostgreSQL-superuser name in the environment variable PRIVX_POSTGRES_USER for the post-install duration. You may set this (and additional environment variables) in /opt/privx/scripts/postinstall_env as described in Section 3.1.

3. Set up additional PrivX servers by duplicating the initial PrivX server:

   • If using virtual servers: Clone the initial PrivX server to create additional instances.

   **Note**

   Ensure that cloned machines obtain unique machine IDs. You can check the current machine ID of a virtual server from its /etc/machine-id file.
If needed, generate a new machine ID with:

```
# rm /etc/machine-id
# systemd-machine-id-setup
```

- **If using physical servers**: Create a backup of the initial PrivX server, then restore the backup to additional machines. Manual backup and restore is described in Section 3.5.1.

**Note**

Ensure that all PrivX server clocks are synchronized.

When creating additional servers via cloning or backups, all servers will have the same server and CA certificates. If you want to create unique server certificates for a PrivX server, run the following on the target server:

```
# /opt/privx/scripts/init_nginx.sh
```

4. Set up a load balancer for the deployment. The load balancer must distribute users to PrivX servers using *sticky sessions*. If you intend to enable native-client connections via PrivX Bastion, the load balancer must also support TCP load balancing.

An example configuration for an Nginx load balancer is provided in Appendix A.

5. Update the trust anchor for PrivX microservices. To do this, run the following command (replace `/path/to/load_balancer_ca_chain.crt` with the path to the CA chain of your load-balancer certificate):

```
# /opt/privx/scripts/init_nginx.sh update-trust /path/to/load_balancer_ca_chain.crt
```

### 3.5 Backing Up and Restoring PrivX Deployments

Back up your PrivX deployment in case of failures. You should at least back up your PrivX deployment before notable maintenance actions (such as system upgrades).

To allow system restore, back up at least the following components:

- The PrivX database.
- One PrivX server.

To create a backup of your PrivX Deployment:

1. If the PrivX database is external, back it up according to your database-vendor instructions.
   
   In single-server deployments the database is backed up along with your PrivX server.

2. Back up your PrivX servers.
   
   - In virtual environments, we recommend creating snapshots of your PrivX servers.
• In physical environments, manually create and download a PrivX-server backup as described in Section 3.5.1. The data on all PrivX servers should be identical, so you only need to do this on one PrivX server.

3. (Optional) In virtual environments, we recommend creating snapshots of the optional PrivX components as well, allowing you to restore them using snapshots instead of by reinstalling.

To restore a PrivX deployment:

1. If the PrivX database is external, restore it according to your database-vendor instructions.

2. Restore your PrivX servers.
   • If using snapshots, restore them now. Remember to ensure afterwards that the system times on your PrivX server(s) are correct.
   • If using manual backups, restore your backup to all your PrivX server(s), as described in Section 3.5.1.

3. Restore any other components in your deployment. If using snapshots, restore them now. Otherwise, reinstall them as described in Section 3.2. Remember to ensure afterwards that the system times on these components is correct.

3.5.1 Manual PrivX-Server Backup and Restore

To manually back up a PrivX server:

1. Create a backup by running:

   # /opt/privx/scripts/backup.sh

   This creates a backup directory to:

   /var/backups/privx/<hostname>_<date>

   Where <hostname> and <date> are the name of the host and the backup timestamp. A working example of a backup-directory path would be:

   /var/backups/privx/privx_2017-12-31-2350

   **Note**

   By default, PrivX servers create daily backups. Automatic backups are created under:

   /var/backups/privx/

   If you have configured the server to store its certificates in non-default locations, or to use a non-default local database name. Check and adjust the Default options in backup.sh before running it.
By default backup.sh backs up:

- PrivX keyvault at /opt/privx/keyvault
- PrivX configurations under /opt/privx/etc
- SSL certificate files located under /etc/pki/CA and /etc/nginx/ssl
- PrivX-CA trust anchor: files matching privx-*.pem under /etc/pki/ca-trust/course/anchors/
- Local PostgreSQL database named privx (only if using local databases).

2. Copy the backup to a secure, external backup location.

To restore a PrivX server from a manual backup, reinstall PrivX and apply the backup files as follows:

1. Uninstall PrivX-server software according to the instructions in Section 3.7.
2. Install the PrivX version from which your backups were taken.
3. Stop all PrivX services to prevent system changes during restore:
   
   ```
   # systemctl stop privx
   ```

4. Restore the PrivX local data. Do this by running (replace /path/to/backup with the path of the backup directory):
   
   ```
   # /opt/privx/scripts/restore.sh /path/to/backup
   ```

5. On all the PrivX servers, run the post-installation script to apply the restored configurations, and to restart the PrivX service:
   
   ```
   # /opt/privx/scripts/postinstall.sh
   ```

### 3.6 Upgrading PrivX Deployment

This section provides instructions for upgrading your PrivX deployment.

**Note**

Before upgrade, we strongly recommend you to create a backup of your PrivX server(s), and of the PrivX database. If upgrade fails, you can restore your old PrivX version from these backups. For more information about backing up and restoring PrivX servers, see Section 3.5.

#### 3.6.1 Single-Server-Deployment Upgrade

To upgrade the PrivX server, gain root-terminal access to your PrivX server and run:

```
# yum install PrivX
```
The system fetches and sets up the latest PrivX package. Upgrade is complete once the command completes.

### 3.6.2 High-Availability-Deployment Upgrade

This section describes the requirements for upgrading a high-availability (HA) PrivX deployment. This section also provides steps in which you may upgrade your HA deployment.

When upgrading a HA PrivX deployment, note the following requirements:

- PrivX servers must not service any users while their PrivX software is being upgraded.
- Ensure that PrivX servers never write to PrivX databases with different product versions.

**Note**

By default, upgrading the PrivX software also upgrades the connected PrivX database. If you need to postpone automatic database upgrade, set the environment variable `SKIP_POSTINSTALL` before upgrading the PrivX software package:

```bash
# export SKIP_POSTINSTALL=1
```

On PrivX servers upgraded like this, you will later need to run `postinstall` to finalize upgrade:

```bash
# /opt/privx/scripts/postinstall.sh
```

One way to upgrade HA deployments is by performing the operations on a duplicate database. This method allows un-upgraded portions of the deployment to run during the procedure. To upgrade a HA deployment in this way:

1. Duplicate the PrivX database.

   Upgrade shall be performed against the duplicate database, without modifying the original database.

2. Upgrade one PrivX server along with the duplicate database:

   i. Disconnect the PrivX server from the load balancer to prevent users from connecting to it.

   ii. To prevent database activity, stop the PrivX services:

      ```bash
      # systemctl stop privx
      ```

   iii. Connect to the duplicate database by providing its connection parameters. You only need to provide those database-connection parameters that differ between the original and the duplicate database.

      - The database-server address and port can be changed in `/opt/privx/etc/shared-config.toml`, under the `db` section.

      - To change the database name (replace `<db_name>` with the database name):

        ```bash
        # /opt/privx/bin/keyvault-tool -name db-name -value <db_name> set-passphrase
        ```
• To change the database-user name (replace <db_user> with the database-user name):

```
# /opt/privx/bin/keyvault-tool -name db-user
   -value <db_user> set-passphrase
```

• To change the password of the database user (replace <db_pwd> with the password):

```
# /opt/privx/bin/keyvault-tool -name db-password
   -value <db_pwd> set-passphrase
```

iv. Upgrade the PrivX software and the connected database:

```
# yum install PrivX
```

v. Reconnect the PrivX server to the load balancer.

3. Set up additional PrivX servers into your upgraded environment:

i. Duplicate the setup of the already-upgraded PrivX server. In virtual environments, you can clone the upgraded PrivX server.

   In physical environments, you can manually back up the upgraded PrivX server and apply that backup to other machines. For more information about manual PrivX-server backup and restore, see Section 3.5.1.

ii. Connect the additional PrivX server to the load balancer.

4. After all the PrivX servers have been upgraded successfully, you should replicate any new data accumulated during the upgrade from the original database to the duplicate database. This completes the upgrade.

   You may remove the original database and leftover PrivX servers after successful upgrade.

3.6.3 Upgrading Optional Components

PrivX Extenders, Carriers, and Web Proxies can be upgraded after PrivX-server and database upgrade. The typical workflow for upgrading any of these components is:

1. Install the latest software package.

2. Run postinstall to complete setup.

The specific steps for each optional component are provided in the following subsections.

Upgrading PrivX Extenders

To upgrade PrivX Extenders, run these steps on all your Extender machines:

1. Install the latest Extender package in either of the following ways:

   • If the PrivX repository is enabled on the machine. Install the package with:
# yum install PrivX-Extender

• Without the PrivX repository, go to Settings→Deployment→Deploy PrivX VPC/VPN Extenders and click Download PrivX Extender:

Then copy the rpm to the Extender host and install it with (replace /path/to/PrivX-Extender.*.rpm with the path to which you placed the Extender package):

```bash
# yum install /path/to/PrivX-Extender-*rpm
```

2. Run postinstall to complete the upgrade:

```bash
#/opt/privx/scripts/extender-postinstall.sh
```

## Upgrading PrivX Carriers and PrivX Web Proxies

PrivX Carriers and Web Proxies together enable web connections, and should be upgraded together.

To upgrade PrivX Carriers:

1. Install the latest Carrier package in either of the following ways:

• If the PrivX repository is enabled on the machine. Install the package with:

```bash
# yum install PrivX-Carrier
```

• Without the PrivX repository, go to Settings→Deployment→Deploy PrivX web access gateways and click Download PrivX Web Access Gateway Components:

Then copy the rpm to the Carrier host and install it with (replace /path/to/PrivX-Carrier.*.rpm with the path to which you placed the Carrier package):

```bash
# yum install /path/to/PrivX-Carrier-*rpm
```

2. Run postinstall to complete the upgrade:

```bash
#/opt/privx/scripts/carrier-postinstall.sh
```

To upgrade PrivX Web Proxies:

1. Install the latest Web-Proxy package in either of the following ways:

• If the PrivX repository is enabled on the machine. Install the package with:

```bash
# yum install PrivX-Web-Proxy
```

• Without the PrivX repository, go to Settings→Deployment→Deploy PrivX web access gateways and click Download PrivX Web Access Gateway Components:

Then copy the rpm to the Web-Proxy host and install it with (replace /path/to/PrivX-Web-Proxy.*.rpm with the path to which you placed the Web-Proxy package):

```bash
# yum install /path/to/PrivX-Web-Proxy-*rpm
```

2. Run postinstall to complete the upgrade:
### 3.7 Uninstalling PrivX

This section provides steps for completely removing PrivX from a machine. These steps may be used in preparation for PrivX reinstallation.

**Note**

Before uninstalling PrivX, make sure to deactivate the license to free up activations. For more information about deactivating licenses, see Section 3.3.2.

To delete all PrivX related data from a PrivX server:

1. Uninstall the PrivX package:
   ```
   # yum erase PrivX
   ```

2. Remove the leftover PrivX installation directory (includes configurations and keyvault data):
   ```
   # rm -rf /opt/privx
   ```

3. If using a local database, remove PostgreSQL and its data directory:
   ```
   # yum erase postgresql
   # rm -rf /var/lib/pgsql/
   ```

4. Remove Nginx:
   ```
   # yum erase nginx
   ```
   Also remove the leftover Nginx server configuration and SSL-certificate files:
   ```
   # rm -rf /etc/nginx
   ```

5. Finally remove the leftover `privx-cert.json` file:
   ```
   # rm /tmp/privx-cert.json
   ```

### 3.8 Manual PrivX Package Installation

Instead of installing PrivX packages from the SSH product repository, you may install the PrivX package manually as follows:

1. Obtain the PrivX RPM package. The PrivX RPM package is a file with a name like the following:
   ```
   PrivX-*.x86_64.rpm
   ```
   Where `*` is to be replaced with the product version.

2. Copy the PrivX RPM to your target server.
3. Install the PrivX RPM (replace `PrivX-*.x86_64.rpm` with the path of the PrivX RPM):

```
# yum install PrivX-*.x86_64.rpm
```
Chapter 4 Getting Started with PrivX

This chapter describes a simple test setup for testing access from roles to hosts:

1. Create local users and roles (Section 4.1).
2. Add hosts, and specify who has access to them (Section 4.2).
3. Access the host via the PrivX GUI (Section 4.3).

This chapter also provides examples for setting up some additional features:

- Enabling certificate-based SSH authentication (Section 4.4).
- Providing access for Active-Directory users (Section 4.5).
- Adding hosts from Amazon Web Services (Section 4.6).

The sections describing the simple test setup should be performed in order. The sections describing additional features may be attempted after the simple test setup.

4.1 Creating Local Users and Roles

PrivX users can log into PrivX to use features (such as connecting to target hosts).

To create a local PrivX user:

1. Navigate to the address of the PrivX server and log in as superuser.
2. In the PrivX GUI, navigate to the Settings→Users page and click Add User.

You will be presented with the New User view.

3. In the New User view, provide the required information about the user. Click Save to save the user.

You should now be able to see your new local user back on the Settings→Users page.

PrivX provides access in a role-based manner. To create a role and assign members:

1. In the PrivX GUI, navigate to the Settings→Roles page, and click Add Role.

You will be presented with a form for providing information about the new role.
2. Provide a name for the new role. Also add users to the role by defining rules. To define a new rule for the role, click Add Rule. In this example, we add a rule to include all local users who have the user name (cn) alice into the role:

Tip

The number of role members is indicated by Matching users. The count is updated when you unfocus from the Search String field (such as by pressing enter, or by clicking somewhere else in the GUI).

Leave the other role settings as they are.

3. Click Save to finalize role creation. Your new role should be visible back on the Settings→Roles page.

To verify the users belonging to the role, click List Members:
4.2 Adding Target Hosts

Make hosts accessible via PrivX:

On the Settings→Hosts page, click Add Host. Provide at least:

- The Name and the network Addresses of the host. This optional data helps users identify the target host.
- The Services (SSH and/or RDP servers) available on the host. In this example we add the SSH server by providing its FQDN address and port number.
- The Accounts to which roles are mapped on the target host. Leave the Password empty to require password authentication upon connecting. In this example we allow the previously-created example role to access the host as target user root.

**Note**

If you are adding an RDP host which uses domain accounts, you must use the `username@domain` syntax. For example, for domain account `Domain\Administrator`, the correct syntax is `Administrator@Domain`.

- Enable SSH - Trust on first use to allow users to accept the SSH host key upon login.
Connecting to Target Hosts

4.3 Connecting to Target Hosts

After you have set up roles to access hosts, you may test connections as follows:

1. Log into the PrivX GUI as the test user we created previously (in Section 4.1).

2. Navigate to the Connections→Available Hosts page. The hosts you can connect to are listed under Available hosts.

   Expand a connection entry to display its available services. In this example, we click the SSH-server service to connect to our test host.

Click Save to save the host.

You may verify that the host is listed back on the Settings→Hosts page.
3. Accept the SSH host key if prompted. Authenticate to the host by providing the password of the target account (not the PrivX account password). You should now be successfully connected to the host.

4.4 Enabling Certificate Authentication for OpenSSH Connections

To enable certificate authentication for OpenSSH connections, run the PrivX host-deployment script on the target host.

The host-deployment script is a Python script that configures the OpenSSH server on the target host to accept certificates issued by PrivX. The script also sets up allowed principals for target users.

To obtain and run the host-provisioning script:

1. Create a host-deployment script. To do this, access the PrivX GUI as superuser, then go to the Settings→Deployment→Deploy and Configure SSH Target Hosts page.
   
   Select Configure using a deployment script, provide a name for the script, then click Add Script. Download the deploy.py script when prompted to.

2. Upload the host-deployment script to the target host. You may do this via the PrivX GUI by connecting to the target host (similarly as in Section 4.3), and then by navigating to the File Transfers tab.

3. Execute the host-deployment script as root on the target host.

   In the command, use --principals to specify the target accounts and the roles that are allowed to access them. Also add the --standalone option if target host was not added previously using a scan script.

   For example, allowing both the target accounts root and johndoe to be accessed by members of Example Role and privx-admin (replace /path/to/deploy.py with the path of the host-deployment script, note that role names with spaces need to be quoted):
SSH connections to the target accounts from the specified roles are now authorized using certificates, without prompting users for passwords.

You may also verify that certificate authorization is used by checking the OpenSSH-server logs on the target server. Upon successful certificate authorization there should be a log message like the following:

```
Accepted publickey for root from 192.0.2.26 port 50930 ssh2: RSA-CERT \
ID alice@127.0.0.1:53188 serial 4920619392583124720 (serial 4920619392583124720) \
```

### 4.5 Importing Users from Directories

You can set up PrivX to automatically add users from user directories. Such users can later be given SSH/RDP access to hosts via PrivX.

For example, to add Active Directory (AD) users:

1. Configure PrivX to scan the AD server for users. To do this, log into PrivX as superuser (or other privx-admin user). Then on the **Settings→Directories** page, click **Add Directory**.

2. Provide the required AD settings.

```
# python /path/to/deploy.py --standalone --principals \
root=Example Role,privx-admin:johndoe=Example Role,privx-admin
```

#### Note

The `%s` in the **User DN pattern** stands for the user name by which AD users may log into PrivX.

For example assume there is an AD user with the following fields:

- `sAMAccountName: alice`
userPrincipalName: alice@ad.example.com

In this case, if User DN pattern were set to (sAMAccountName=%s), the user can log in with the user name 
alice. If User DN pattern were set to (userPrincipalName=%s), the user can log in with the user name alice@ad.example.com.

**Save** the directory settings. PrivX automatically connects to the AD server to add any users found with the given settings.

3. You may verify the AD status back on the Settings→Directories page. After PrivX finishes adding users from the AD, the connection status should display **OK**, along with the number of users added.

To list the users added from the AD, perform a **List Users** action on the AD entry.

To grant AD users access to hosts and services, add them to roles (similarly as in Section 4.1). For example, you can add an additional rule to the Example Role that was created earlier.

AD users may then log into the PrivX GUI and establish SSH/RDP connections. The allowed connection targets are determined by the role(s) assigned to the AD users.
4.6 Importing Hosts from Directories

You can set up PrivX to automatically add existing hosts on cloud platforms. Such hosts can later be connected to via PrivX.

For example, to add hosts from Amazon Web Services (AWS):

1. In your AWS, add a policy to allow host scans. To do this, access your AWS and navigate to IAM→Policies, then create a policy with the following JSON:

   ```json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
   "Action": "ec2:Describe*",
   "Resource": "*"
   }
   ]
   }
   ```

2. Create an IAM user with permissions to use the host-scan policy. This can be done on the IAM→Users page.

   The IAM user must have Programmatic access, and be attached with the host-scan policy.

   Note the Access key ID and the Secret access key of the user. These are required later for configuring PrivX against AWS.
3. Configure PrivX to scan and add the AWS hosts.

Log into PrivX as superuser (or other privx-admin user). On the Settings→Directories page, click Add Directory.

Fill in the basic information of the directory. To allow PrivX to detect AWS hosts, add the Access key ID and the Secret access key of the IAM user.

You can selectively filter hosts using the Fetch hosts with tag option found under the Advanced directory settings.

Save the directory settings. PrivX begins importing hosts from AWS.

After a moment, you may verify the directory status back on the Settings→Directories page. The Connection should be in the OK state, and list the number of instances found on AWS.

To list the imported hosts, click List Hosts.

You may then Edit hosts to add services and account mappings to them, similarly as in Section 4.2.
Chapter 5 Advanced Configuration

This chapter provides instructions for customizing PrivX components to suit the needs of your environment.

5.1 SSL/TLS Security

This section describes the configuration topics related to SSL/TLS security.

5.1.1 Setting Up a Trusted Server Certificate

Freshly set-up PrivX servers are equipped with self-signed server certificates. For production deployments we recommend replacing the self-signed server certificate with a server certificate issued by a trusted Certificate Authority (CA).

Note

The self-signed server certificate should only be trusted in evaluation deployments. In production environments, we strongly recommend that you acquire and use a trusted server certificate. You can obtain a trusted server certificate by enrolling the certificate-signing request with your corporate CA.

1. The initial setup has generated a certificate-signing request (CSR), which can be enrolled with a trusted CA for a trusted server certificate. The CSR is located at the following path on the PrivX server:

/etc/nginx/ssl/nginx.csr

Enroll this CSR with your CA. In response, the CA should provide you with the following:

- The server certificate.
- The CA-certificate chain of the CA itself.

Note

Note to certificate-signing authorities: The PrivX CSR contains subjectAltName definitions for DNS and IP addresses. These are critical to PrivX operation and must be preserved in the signed server certificate.
2. Copy the PEM (Base64) encoded server-certificate file to the `ssl_certificate` location on the PrivX instance. By default, the location is:

```
/etc/nginx/ssl/nginx.crt
```

Ensure that the server-certificate file has correct ownership, permissions, and SELinux context:

```
# chown root:nginx /etc/nginx/ssl/nginx.crt
# chmod 0640 /etc/nginx/ssl/nginx.crt
# restorecon /etc/nginx/ssl/nginx.crt
```

3. Update the trust anchor for PrivX microservices. To do this, run the following command (replace `/path/to/ca_chain.crt` with the path to the CA-certificate-chain file):

```
# /opt/privx/scripts/init_nginx.sh update-trust /path/to/ca_chain.crt
```

**Note**

In single-server deployments, provide the CA chain of the PrivX-server certificate. In HA deployments, provide the CA chain of the load-balancer certificate.

`init_nginx.sh` requires PEM-encoded certificate files to have Unix line endings. If the command fails, ensure correct line endings in the CA-certificate-chain file, then rerun the command.

4. Finally, restart the Nginx and PrivX services to start using the new server certificate:

```
# systemctl restart nginx
# systemctl restart privx
```

### 5.1.2 Setting Up a Trusted CA Certificate

PrivX Authorizer issues access certificates using its own CA, which is self-signed by default. For production deployments we recommend replacing the Authorizer CA with one signed by a trusted certificate authority.

**Note**

The following procedures use `<key_id>` and `<cert_id>` to substitute for PrivX-CA-key ID and the Authorizer-certificate ID respectively. Be careful not to mix them up.

To set up PrivX Authorizer with a trusted CA certificate:

1. Gain root-terminal access to a PrivX server, and display the PrivX-CA-key ID by running:

```
# /opt/privx/bin/keyvault-tool -name "PrivX CA Key" list-asymmetric
```

2. Generate a Certificate-Signing Request (CSR) for the CA private key (replace `<key_id>` with the PrivX-CA-key ID):

```
# /opt/privx/bin/cert-tool -command create -type authorizer-ca \
-subject "OU=PrivX Authorizer CA/CN=PrivX CA" \  
-key <key_id> -csr -csrout privx-im-ca.csr
```
This creates the CSR to the file `privx-im-ca.csr` in your current working directory.

3. Request a trusted CA to sign the CSR with the following constraints:

- **keyUsage** must include: Digital Signature, Key Agreement, Certificate Sign, CRL Sign
- **basicConstraints** must be: CA:TRUE, pathlen:0

After the trusted CA provides you with the signed certificate, you may verify its constraints with (replace `privx-im-ca.crt` with the path to the signed certificate):

```
$ openssl x509 -text -noout -in privx-im-ca.crt
```

4. Replace the Authorizer certificate. First, obtain the ID of the current Authorizer certificate:

```bash
# /opt/privx/bin/cert-tool -command list -type authorizer-ca
```

Then delete the current Authorizer certificate (replace `<cert_id>` with the Authorizer-certificate ID):

```bash
# /opt/privx/bin/cert-tool -command delete -id <cert_id>
```

Finally, import the signed certificate (replace `<key_id>` with the PrivX-CA-key ID, replace `privx-im-ca.crt` with the path to the signed certificate):

```bash
# /opt/privx/bin/cert-tool -command import -type authorizer-ca -key <key_id> -in privx-im-ca.crt
```

5. On all PrivX servers, restart the PrivX services to apply the changes:

```bash
# systemctl restart privx
```

Subsequent certificates issued by PrivX are signed using the new trusted Authorizer CA certificate.

### 5.1.3 Allowed SSL Protocols and Ciphers for GUI Connections

Connections to the PrivX GUI are secured using TLS. The allowed SSL protocols and SSL ciphers may be adjusted if some browsers cannot establish connections to the PrivX GUI, or if you want to harden the PrivX instance.

The allowed SSL protocols and SSL ciphers are defined in the Nginx configuration file `/etc/nginx/conf.d/privx.conf`, by the parameters `ssl_protocols` and `ssl_ciphers` respectively, similarly to the following:

```nginx
http {
    sendfile on;
    server_tokens off;

    ssl_protocols  TLSv1.2;
    ssl_ciphers 'AESC128+EECDH:AESC256+EECDH:AESGCM+EECDH:EDH:AES+EECDH: ... ';
    tcp_nodelay on;
...```
After any adjustments to Nginx settings, restart the Nginx web server to apply the changes:

```
# systemctl restart nginx
```

### 5.1.4 Rotating Encryption Keys

This section provides instructions for rotating PrivX cryptographic keys, which should be done regularly for improved security. You can rotate the following keys:

- PrivX CA keys, which issue certificates for certificate-based access.
- Authentication keys, which issue PrivX-user access and session tokens.

To rotate PrivX CA keys:

1. Run the rotation script from a PrivX server in the deployment:

```
# /opt/privx/scripts/rotate_privx_ca.sh
```

   The old PrivX CA certificates are deleted and new ones are generated. The script will then restart PrivX.

2. On hosts with certificate authentication enabled, configure trust for the new PrivX CA. To do this:
   - **On target hosts with certificate-based SSH authentication:** run the host-deployment script with the `--rotate-ca` option:
     ```
     # deploy.py --rotate-ca
     ```
   
   Alternatively, manually configure the host to trust the new PrivX CA as described in the section called “Trusting PrivX Certificates”.
   
   - **If you use certificate-based RDP authentication:** Set up trust for the new PrivX CA as described in Section 7.5.2.

To rotate PrivX authentication keys:

1. Run the rotation command on a PrivX server in the deployment:

```
# /opt/privx/bin/keyvault-tool rotate-auth-keys
```

   This generates new authentication keys.

2. On all PrivX servers, restart the PrivX services:

```
# systemctl restart privx
```

**Note**

PrivX-user logins may fail during the time between authentication-key generation and restarting PrivX services.

### 5.2 PrivX Log Settings
By default, regular and audit events are logged to syslog. Audit events use the namespace `SSH-PRIVX-AUDIT`, and other (info/debug/error) messages use the namespace `SSH_PRIVX`.

You can adjust the locations to which PrivX writes logs, and the log level of each microservice.

The log locations are configured by settings defined in `/opt/privx/etc/shared-config.toml`

```
# Should we send PrivX system events to logs under /var/log/privx
# send_regular_events_to_stdout = false
send_regular_events_to_stdout = false

# Should we send PrivX audit events to syslog
# send_regular_events_to_syslog = true
send_audit_events_to_syslog = true

# Should we send PrivX system events to syslog
# send_regular_events_to_syslog = true
send_regular_events_to_syslog = true
```

To enable debug output for a microservice:

1. Edit `/opt/privx/scripts/<microservice_name>.sh` (replace `<microservice_name>` with the name of the microservice) and set the `SSH_TRACE` parameter:

   ```
dir="/opt/privx/bin"
cmd="SSH_LOG_LEVEL=DEBUG SSH_TRACE=6 /opt/privx/bin/auth"
user="privx"
```

2. By default, the system-logging service `rsyslog` is configured to show INFO-level messages only, which blocks DEBUG messages. To configure `rsyslog` to show DEBUG-level messages, find the following line in the `rsyslog` configuration file, `/etc/rsyslog.conf`:

   ```
*.info;mail.none;authpriv.none;cron.none /var/log/messages
```

   And change it to, for example:

   ```
*.debug;mail.none;authpriv.none;cron.none /var/log/messages
```

   Restart `rsyslog` to apply the changes:

   ```
systemctl restart rsyslog
```

3. Restart the `privx` service to apply any changes to configuration files:

   ```
# systemctl restart privx
```

### 5.3 PrivX-Extender Configuration

This section describes configuration topics for PrivX Extenders.

#### 5.3.1 Configuring Extender Log Location

By default PrivX Extender logs info and errors to `/var/log/privx/privx-extender.log`
If you want to enable logging to syslog, specify the rsyslog address and protocol in `/opt/privx/etc/extender-config.toml`, similar to the following:

```toml
[syslog]
syslog_protocol="tcp"
syslog_address="localhost:514"
```

Restart PrivX Extender to apply the changes. In addition make sure rsyslog is enabled on the extender host:

```bash
systemctl restart privx-extender
systemctl restart rsyslog
```

### 5.3.2 Proxying Native-Client Connections

To allow proxying native-client connections via PrivX Extenders:

1. On all your PrivX servers, enable the `forwarder_enabled` setting in `/opt/privx/etc/ssh-proxy.toml`

And restart PrivX services to apply the changes:

```bash
systemctl restart privx
```

⚠️ **Caution**

The forwarder relays all the data it receives (not just the native-client connections), and should not be enabled in high-security networks.

2. Session recording must be disabled on hosts that are to be accessed using proxied native-client connections. For more detailed instructions about toggling session recording, see Section 9.3.

3. *(Optional)* To simplify native-client commands, specify the required connection parameters in the users' client configuration (typically at `/etc/ssh/ssh_config` or `~/.ssh/config`). You can do this using `Host` blocks that at least specify:

- The `HostName` in `extender-name/target-host-address` format.
- The `ProxyCommand`: `privx-nc -x $PRIVX_AGENT_PROXY %h %p`

For example:

```bash
Host bilberry
  HostName example-extender/bilberry.example.com
  ProxyCommand privx-nc -x $PRIVX_AGENT_PROXY %h %p
```

After setup, you can connect to target hosts as follows:

1. As the native-client user, start the PrivX agent (if not already started) and use it to log into PrivX.

2. If you have specified the required parameters in your SSH-client configuration, you can connect simply using the appropriate `Host` block. For example:

```bash
$ ssh target-user@bilberry
```
Otherwise, you must additionally provide the ProxyCommand and the name of the PrivX Extender, similar to the following:

```
$ ssh -o "ProxyCommand privx-nc -x $PRIVX_AGENT_PROXY %h %p" target-user@example-extender/bilberry.example.com
```

```
$ sftp -o "ProxyCommand privx-nc -x $PRIVX_AGENT_PROXY %h %p" target-user@example-extender/bilberry.example.com
```

```
$ scp -o "ProxyCommand privx-nc -x $PRIVX_AGENT_PROXY example-extender/%h %p" source/file/path target-user@bilberry.example.com:/target/file/path
```

### 5.4 PrivX-Carrier and Web Proxy Configuration

This section describes configuration topics for the PrivX Carrier and PrivX Web Proxy.

#### 5.4.1 Allowing HTTP/HTTPS Targets at Unusual Ports

This section describes modifying the allowed target ports for PrivX Web Proxies. These instructions assume you to have some familiarity with Squid configuration.

By default PrivX only allows access to HTTP/HTTPS targets running on certain ports. The allowed ports are available toward the start of your Squid configuration file.

To allow access to targets at nonstandard ports:

1. Verify and set the allowed target ports on your Web Proxy machines in `/etc/squid/squid.conf`

   - For HTTP and HTTPS targets, ensure the target port is included in the `Safe_ports` list. You can add the target port by adding a similar line immediately after other `acl Safe_ports` directives (replace `<target_port>` with the port of your HTTP/HTTPS service):

     ```
     acl safe_ports port <target_port>
     ```

   - Additionally for HTTPS targets, ensure the target port is included in the `SSL_ports` list. You can add the target port by adding a similar line immediately after other `acl SSL_ports` directives (replace `<target_port>` with the port of your HTTPS service):

     ```
     acl SSL_ports port <target_port>
     ```

   For example, to allow access to HTTPS target running on port 9443, you would need to add the port to `SSL_ports`:

   ```
   acl SSL_ports port 443
   acl SSL_ports port 9443
   acl Safe_ports port 80  # http
   acl Safe_ports port 21  # ftp
   ```
acl Safe_ports port 443 # https
acl Safe_ports port 1025-65535 # unregistered ports
acl CONNECT method CONNECT

Port 9443 is already included in Safe_ports by default.

5.4.2 Changing Web Proxy Port Numbers

To change the Web Proxy ports, you need to edit the following:

ℹ️ Note

In the following examples, replace the port numbers 18080 and 18443 with the ports you want to use.

- On the Carrier host, change the ports in the /opt/privx/etc/carrier-container.toml file:

```toml
[web_container]
# Web proxy ports for HTTP and HTTPS. The port numbers and Carrier host IP address should match the Squid configuration file in /etc/squid/squid.conf
# Proxy server IP address is configured via PrivX UI.
http_proxy_port = 18080
https_proxy_port = 18443
```

After editing the file, restart the Carrier service:

```
# systemctl restart privx-carrier
```

- On the Web Proxy host, edit the Squid configuration file in /etc/squid/squid.conf to match the port numbers you entered previously:

```conf
http_port 18080
http_port 18443 ssl-bump cert=/opt/privx/squid_cert/squid.crt 
key=/opt/privx/squid_cert/squid.key generate-host-certificates=on 
dynamic_cert_mem_cache_size=4MB
```

After editing the file, restart the Web Proxy and Squid services:

```
# systemctl restart privx-web-proxy squid
```

- The Web Proxy host's firewall should allow access to the ports you have specified. Run:

```
# firewall-cmd --permanent --add-port=18443/tcp
# firewall-cmd --permanent --add-port=18080/tcp
# firewall-cmd --reload
```

5.5 Setting Custom Instance Names

You can set a custom instance name to better distinguish between multiple PrivX servers. Custom instance names are displayed in the PrivX GUI, and added to the audit events generated from the server.
To set a custom instance name for a PrivX server:

1. Gain root-terminal access to the PrivX server.

2. Edit the shared configuration at:

   ```bash
   /opt/privx/etc/shared-config.toml
   ```

   In this file, set the custom instance name with `privx_instance_name`. For example:

   ```toml
   # A custom PrivX instance name to be shown in the header
   privx_instance_name = "PrivX Example Deployment 01"
   ```

   Save your changes to the file.

3. Restart the PrivX services to apply the changes:

   ```bash
   # systemctl restart privx
   ```

### 5.6 Resetting the Superuser Password

If you have forgotten the PrivX superuser's password, you can reset it as follows:

1. On a PrivX server, obtain an encrypted version of your new password (replace `example_password` with your new password):

   ```bash
   # /opt/privx/bin/keyvault-tool bcrypt <example_password>
   ```

2. Access the PrivX database using `psql` with write permissions. Change the superuser password (Replace `encrypted_password` and `superuser` with the encrypted password and your superuser-account name respectively):
# UPDATE localuser SET password='encrypted_password' WHERE username='superuser'

You can now log into PrivX as superuser using the new password.
Chapter 6 PrivX Users and Permissions

This chapter describes user and permission management:

- Adding PrivX users.
- Providing role-based permissions for PrivX users.
- Workflows for requesting/revoking roles for PrivX users.

6.1 Adding PrivX Users

You can add PrivX users in the following ways:

Add local users

On the Settings→Users page click Add User, and then provide their information, such as credentials and contact information.

Local users are an easy way to add PrivX users.

Note

A local user must have a password that is at least eight characters long.

Import existing users from LDAP, AD, and/or OpenID Connect providers

On the Settings→Directories page click Add Directory. Select a Type from one of the supported User Directories. Provide other required information about the LDAP/AD directory, such as connection information and the query for obtaining user entries.

For additional information about LDAP/AD configuration, see Section 6.5.

For additional information about OpenID Connect (OIDC) configuration, see Section 6.6.

6.2 Granting Permissions for PrivX Users
PrivX users gain permissions from roles. Roles may allow users to:

- Access target hosts.
- Approve/deny requests.
- Manage connections.
- Perform PrivX administration.

Members of a role automatically receive the permissions from their roles. In other words, users gain permissions by becoming members of roles. Users may become role members in either of the following ways:

- The user is included in the role via rules *(mapped users)*. For more information about configuring rules for roles, see Section 6.2.1.

- The user has been approved as a member of the role *(approved users)*. For more information about approval mechanisms, see Section 6.2.2.

All users automatically start as members of the privx-user role.

### 6.2.1 Managing Roles

You can create, edit, and remove roles from the Settings→Roles page. On this page you can also list the members belonging to the selected role and where the role grants access to.

A role consists of:

- **Rules**: LDAP filters for specifying the members of the role.
- **Permissions**: Allow specific management and viewing options.
- **SSH Options**: Allow SSH options.

For more information about granting access to target hosts, see Section 6.2.3.

**Note**

Role changes take effect within 1 - 5 minutes.

### 6.2.2 Requesting and Approving Memberships

For active PrivX users, permission changes take effect when their access token is refreshed. The interval is specified in `/opt/privx/etc/oauth-shared-config.toml`, by the setting `access_token_validity`.
PrivX users can submit requests to grant or remove membership from roles, either for themselves or on behalf of others. To submit a request, go to the Requests→My Requests page and click New Request.

In a request, you need to specify (at least):

- The user whose membership is to be granted or revoked.
- The role to be granted or revoked.

You can review the status of your requests on the Requests→My requests page.

PrivX users can approve and deny requests on the Requests→Approvals page. Click a request to submit a decision.

You need to have an existing workflow for a role before users can request/revoke memberships from the role. For more information about workflows, see Section 6.3.

**Note**

PrivX users may not approve their own requests.

A request must gather all the approvals to be approved; One denial is enough to deny an entire request.

Only the mapped users of a role can submit decisions required from the role.

Users granted membership by the rules of the role can only be removed by adjusting the rules.

Role changes take effect within 1 - 5 minutes from being applied.

OpenID Connect users can not request memberships for roles.

### 6.2.3 Granting Access to Target Hosts

PrivX users may access a target host if one or more of their roles is mapped to a target user. To allow the members of a role to connect to a target host:

1. Navigate to the Settings→Hosts page, and Edit the target host.

2. Under the Accounts section, specify mappings from roles to target users. Multiple roles may be mapped to one target user. If a role is mapped to multiple target users, the members of that role may choose which user to connect as.

Save your changes to the host. PrivX users belonging to the specified role(s) can now access the target host as the target user(s) they are mapped to. PrivX users can see accessible target hosts on the Connections→Available Hosts page.

For more information about connecting to target hosts see Chapter 8.

### 6.2.4 Granting Administrator Permissions
You can grant PrivX administrator permissions for a user by giving them the `privx-admin` role. Similarly, you can remove administrator permissions by removing the user from the `privx-admin` role.

PrivX administrator permissions allow users to:

- Manage directories, hosts, roles, users, sessions, and workflows.
- Connect to any target host.

### 6.3 Managing Workflows

Workflows allow PrivX users to grant or revoke role memberships for themselves and for others. You can create, edit, and remove workflows from the Settings→Workflows page.

A workflow contains one to three steps, each defining some approval requirements. Once a request gets all the approvals from a step, it proceeds to the next step. Membership is granted to the target user once the approvals from each step have been gathered.

For more information about requests, see Section 6.2.2.

#### 6.3.1 Enabling Email Notifications

PrivX can automatically notify approvers when a new request requires their attention. This can happen when a new request is created, or when a request proceeds to a new step.

1. To enable automatic email notifications, configure PrivX against an existing SMTP server. You can set the relevant email settings from the PrivX GUI, on the Settings→Workflows page.

2. If the email-server connection uses TLS, the email-server root-CA certificate must be added to the system-wide trust anchors on your PrivX servers. To do this, copy the root-CA certificate under:

   `/etc/pki/ca-trust/source/anchors/`

   Then run:

   ```bash
   # update-ca-trust extract
   ```

### 6.4 User Configuration

This section describes the configuration topics related to PrivX users.

#### 6.4.1 Automatic Logout

Settings for automatically logging out PrivX users are in `/opt/privx/etc/oauth-shared-config.toml`. You can set PrivX to automatically log out users for one or more of the following reasons:
• Inactivity - Set with `refresh_token_valid`.

• Time since login (optional) - Set with `session_valid`. If this is not set, `refresh_token_valid` effectively specifies the maximum session validity.

For example, to automatically log out PrivX users after 30 minutes of inactivity, or after 8 hours from initial login:

```
refresh_token_valid="30m"
session_valid="8h"
```

To automatically log out PrivX users after a certain duration regardless of activity, uncomment `session_valid` and set the duration in `refresh_token_valid`:

```
refresh_token_valid="30m"
# session_valid="8h"
```

### 6.4.2 Require Password Change

You can require a PrivX local user to change their password on next login. This may be useful in scenarios where PrivX administrators hand out PrivX accounts to end users.

To require a target user to change their password:

1. On the Settings → Users page, Edit the target user to display their settings.

2. Under Additional options, select Require password change.

   Click Save to apply your changes. Next time the target user logs in they will be prompted to change their password, before being allowed to do anything else.

### 6.4.3 Limiting Login Rate

After a user and/or IP address repeatedly fails to log into PrivX, you can temporarily prevent them from making further login attempts to PrivX. This can help secure your PrivX against brute-force-access attempts.

To set limits for failed logins, perform the following on each PrivX server:

1. Modify the settings under the `[loginratelimit]` section in the configuration file:

   `/opt/privx/etc/auth-default-config.toml`

2. Restart the PrivX services to apply:

   ```
   # systemctl restart privx
   ```

### 6.5 AD/LDAP Directory Configuration

This section describes configuration topics for AD and LDAP directories.
Users from added AD/LDAP directories can log into PrivX, and furthermore to target hosts (as permitted by their roles). The general workflow for providing access to AD/LDAP users is as follows:

1. To add an AD/LDAP directory, go to the Settings→Directories page and click Add Directory.

2. Set the directory Type to Active Directory or LDAP. Provide any required directory settings, such as connection and bind parameters.

   **Note**

   By default, PrivX allows directory users to log in with userPrincipalName (typically given in username@domain format) as username.

   You may set User DN pattern to use another field as the user name. For example, to allow directory users to log in using their uid as username, change the setting to:

   \[(uid=\%s)\]

   After you have provided the necessary settings, click Save to apply your changes.

3. **Optional:** Back on the Settings→Directories page, you may verify that PrivX is able to import users from the directory: the directory Status should be OK, and it should display the correct number of users.

Any imported PrivX users can login to the PrivX GUI using their AD/LDAP credentials. To further allow such users to log into target hosts, add them to roles as described in Section 6.2.

The following subsections provide instructions for some of the more advanced AD/LDAP setup scenarios.

### 6.5.1 Secure-Connection Setup

To allow TLS-based secure connections (STARTTLS or LDAPS) to directory servers, the following requirements must be met:

- The directory-server certificate must specify its DNS and IP addresses in the Subject Alternative Name.

- The server-certificate CA must be trusted on the PrivX server. This can be satisfied, for example, by adding the CA certificate to the trust store of the PrivX server.

To enable secure connections to user directories, perform the following steps on the PrivX server:

1. Ensure the directory-server certificate specifies the server DNS and/or IP addresses in the SubjectAltName field. You can do this, for example, using openssl (replace directory.example.com and 636 with the address and the port of the directory service):

   ```bash
   $ echo "Q" | \
   openssl s_client -connect directory.example.com:636 | \
   openssl x509 -noout -text
   ```

2. On the PrivX server, place the directory-server CA certificate under the trust-anchors directory, located at:
Then to add the certificate to the system trust store, run:

```bash
# update-ca-trust extract
```

3. You may verify that the PrivX server correctly authenticates the directory server (replace `directory.example.com` and 636 with the address and the port of the directory service):

```bash
$ openssl s_client -connect directory.example.com:636
```

With the correct certificate setup, you should get:

```
Verify return code: 0 (ok).
```

4. Restart PrivX to apply the changes:

```bash
# systemctl restart privx
```

### 6.5.2 Adjusting User Matching

PrivX applies a default pre-filter for matching user records. You may adjust the pre-filter to affect what records are imported.

To override the pre-filter for a user directory, specify your custom `User filter` in the directory settings:

1. On the **Settings→Directories** page, **Edit** your user directory.

2. Under the **Active directory settings** section, specify a **User filter** for matching users on the user directory.

3. Click **Save** to apply your changes. Your new **User filter** overrides the default pre-filter.

You can also change the default pre-filter globally from the Role Store configuration file, located under the following path on the PrivX machine:

```
/opt/privx/etc/rolestore.toml
```

Under the `[ldap]` section of the file, look for the settings `default_user_filter` and `attributes`. These settings respectively specify what objects are recognized as users, and what fields are fetched from matching records.

```
default_user_filter = "(|(objectClass=user)(objectClass=person)(objectClass/inetOrgPerson))"
attributes = "objectClass cn dn distinguishedName ... "
```

For default OpenLDAP setup you could specify, for example, the following line with `inetOrgPerson` and `posixAccount`:

```
default_user_filter = "(&(objectClass=inetOrgPerson)(objectClass=posixAccount))"
```

If you are using two directories with different schemas, you can combine them for example as follows:
default_user_filter = "(|(&(objectClass=user)(objectClass=person))(&(objectClass/inetOrgPerson)
(objectClass=posixAccount)))"

To allow rules in PrivX roles to match nested groups, you must enable enable_nested_groups in the Role Store configuration file /opt/privx/etc/rolestore.toml

Restart PrivX services to apply any changes.

# systemctl restart privx

Note

User filters may also be specified in directory settings. To modify directory settings, go to Settings→Directories and Edit your user directory.

When a User filter is specified in directory settings, only those records matching both the default_user_filter and User filter are imported as PrivX users.

6.5.3 Refreshing Directory Data

User-directory changes are updated to PrivX in the following ways:

- By default, PrivX refreshes directory data every 15 minutes. To adjust the refresh interval of a directory, go to Settings→Directories and Edit the target directory. Shortening the interval allows PrivX to detect user changes faster, while lengthening the interval reduces system load.

- To immediately refresh directory data, go to Settings→Directories and perform a Refresh action on the target directory.

6.6 OpenID-Connect Configuration

This section describes the configurations for allowing users from an OpenID Connect (OIDC) provider to log into PrivX.

Users from added OIDC providers can log into PrivX via the PrivX GUI, and furthermore to target hosts (as permitted by their roles). Such users are authenticated against the OIDC provider.

To provide access for OIDC users, set up your OIDC and PrivX as follows:

1. Configure your OIDC provider to provide role information in End-User Claims.

   End-User Claims from the OIDC provider must include tags that can be used to associate the End User to PrivX Roles. For example, in the following ID Token/Userinfo response, the groups Claim provides the required tags:

   ```
   {
       "sub": "248289761001",
       "name": "Jane Doe",
       "given_name": "Jane",
       "family_name": "Doe",
   }
   ```
"preferred_username": "j.doe",
"email": "janedoe@example.com",
"groups": ["Example Role 01", "Example Role 02"]
}

2. PrivX requires continued access to the OIDC provider for updated group and role information. Ensure your OIDC provider is configured to issue access and refresh tokens for PrivX. Provide the following redirect URI to the OIDC provider:

https://privx.example.com/auth/api/v1/oidc-cb

3. Add the OIDC provider to PrivX. To do this, go to the Settings→Directories page and click Add Directory.

4. Set the directory Type to OpenID Connect. Provide any required settings, such as the OIDC issuer URL and client credentials.

**Note**

To allow OIDC users to log into personal accounts, specify the claim(s) containing the users' personal-account names. Then in Source=Target attribute pairs, map these claims to unix_account or windows_account for SSH and RDP connections respectively.

For example, if the users' Windows logon names are in their email claim, specify the following:

email=windows_account

As another example, if the users' preferred_username claim matches both their Unix- and Windows-account names, specify the following:

preferred_username=unix_account,windows_account

After you have provided the necessary settings, click Save to apply your changes.

5. **Optional:** You may test OIDC logins from the PrivX login page. You can log in using either of the following methods:

   • Via the PrivX login page, click the OIDC authentication method listed below the Login button.

   • Go directly to the OIDC login page. To display the link to the OIDC login page, go to Settings→Directories and click your OIDC directory. The link is in The Direct Login URL, under the OpenID connect settings section.

Any imported PrivX users can login to the PrivX GUI using their OIDC credentials. To further allow such users to log into target hosts, add them to roles as described in Section 6.2.

### 6.7 Advanced Authentication for PrivX Users

This section describes additional authentication configurations that allow greater ease and security for PrivX users logging in.
6.7.1 Kerberos Authentication for PrivX Users

In PrivX you may enable Kerberos Single Sign On (SSO) for PrivX users belonging to one AD/LDAP user directory. Users from that directory can, after initial login to a Kerberos client, log into the PrivX GUI without having to retype their credentials.

Prerequisites

Ensure and perform the following before setting up Kerberos authentication for PrivX:

• You need a functioning Kerberos Key Distribution Center (KDC) and admin server. These services may be run on the same server.

• Kerberos servers must be able to resolve the host names of your PrivX servers.

• Users must be part of a AD/LDAP user directory. The user directory must have been added to PrivX. Furthermore, users from that directory must have access to some Kerberos client for obtaining their Kerberos tickets (such as Windows login or \texttt{kinit}). Note that you may only enable Kerberos authentication for one user directory in PrivX at any time.

For more information about adding user directories to PrivX, see Section 6.1.

PrivX Integration to Kerberos

To enable Kerberos authentication for PrivX users belonging to a user directory, repeat the following steps for each of your PrivX servers:

1. To add PrivX as a service to Kerberos, create a unique service principal for the PrivX server. The principal name must follow the syntax:

   \texttt{HTTP/privx.example.com@EXAMPLE.COM}

   Replace the example values as follows:

   • \texttt{privx.example.com} - the PrivX server name.

   • \texttt{EXAMPLE.COM} - the name of your Kerberos realm.

2. To allow the PrivX to authenticate against the KDC, obtain a keytab of the PrivX service principal. The keytab entries should be encrypted using algorithms supported by your Kerberos environment. If using Windows Kerberos, also ensure the keytab is mapped to the user account associated with the PrivX service principal.

   \textbf{Caution}

   Ensure that the keytab is never exposed to unauthorized personnel. Malicious users with access to the keytab may use it to impersonate services.

3. On your PrivX server, add the keytab to \texttt{/etc/krb5.keytab}.

   Also ensure the keytab-file is readable by the local user \texttt{privx}. For example:
4. Configure your PrivX server to connect to your KDC. To do this, add a similar configuration to `/etc/krb5.conf` on the PrivX server:

```plaintext
[libdefaults]
default_realm = EXAMPLE.COM

# The following krb5.conf variables are only for MIT Kerberos.
krb4_config = /etc/krb.conf
krb4_realms = /etc/krb.realms
kdc_timesync = 1
cache_type = 4
forwardable = true
proxiable = true

[realms]
EXAMPLE.COM = {
  kdc = dc.example.com
  admin_server = dc.example.com
}

[login]
krb4_convert = true
krb4_get_tickets = false
```

Adjust the configuration according to your Kerberos environment. You will at least have to change `default_realm` to the actual name of your Kerberos realm, change the `[realms]` section to include the actual name of your Kerberos realm, and to specify the addresses of its KDC and Kerberos admin server.

Save your changes to the file.

5. On the PrivX server, adjust the PrivX settings in `/opt/privx/etc/auth-default-config.toml`. You will have to set the following:

- `kerberos_enabled` - set to `true`
- `kerberos_service_name` - set this equal to the PrivX service principal.
- `kerberos_directory_name` - the name of the user directory for which Kerberos is enabled. For a list of directories and their names, see the Settings→Directories page in the PrivX GUI.
- `kerberos_realm_name` - the name of your Kerberos realm.

Example values for these settings:

```plaintext
kerberos_enabled=true
kerberos_service_name="HTTP/privx.example.com@example.COM"
kerberos_directory_name="Example AD 01"
kerberos_realm_name="EXAMPLE.COM"
```
Save your changes to the file.

6. On the PrivX server, restart PrivX services to apply the changes:

```
# systemctl restart privx
```

Once you have completed these steps on each PrivX server, Kerberos SSO is enabled.

**Logging in with Kerberos**

After you have set up kerberos SSO, PrivX users belonging to the configured user directory can log in as follows:

1. Obtain your Kerberos ticket, for example, by logging into a machine with Kerberos authentication or by running `kinit`.
2. Access the PrivX login page. You will be automatically logged into PrivX, without having to re-enter your credentials.

**6.7.2 Multi-Factor Authentication for PrivX Users**

When multi-factor authentication (MFA) is enabled for PrivX users, they are required to input a time-based PIN code (in addition to their account password) to log into PrivX.

![Note]

PrivX users must set up and use the *Google Authenticator app* to successfully log in with multi-factor authentication.

**Enabling MFA for Directories**

To enable/disable MFA for a user directory:

1. On the **Settings**→**Directories** page, **Edit** a directory entry.
2. Expand **Advanced directory settings**. Under the **Multi-factor authentication settings** section, set the **MFA type**. Set it to **TOTP MFA (time-based)** to enable MFA; Set it to **Disabled** to disable MFA.
3. Click **Save** to apply the changes.

**Logging in with MFA**

When MFA is enabled, PrivX users can obtain their MFA code and log in as follows:

1. Log into PrivX normally, using your user name and password.
2. If you have not done so before, you will be asked to import your MFA code. This allows you to obtain PIN codes for MFA login.
Scan or enter the code into your authenticator app (such as Authy or Google Authenticator). After this, your authenticator app should display 6-digit time-based PIN codes.

Click Next to proceed.

3. Enter the PIN code displayed in your authenticator app. After this you should be logged into the PrivX GUI.

Reobtaining MFA Codes

If a PrivX user has lost their MFA code, you can set PrivX to offer a new MFA code upon their next login:

1. On the Settings→Users page, click the user who needs a new MFA code.

2. Under Multi-factor authentication, click and select Reset MFA Pairing. Verify that the Multi-factor authentication status is updated to Enabled, not activated.

The user is prompted to import a new MFA code upon their next PrivX login.

6.7.3 Client-Certificate Authentication for PrivX Users

Users imported from AD and LDAP directories can log in to PrivX using SSL certificates. Such certificates may be for example from users' smart cards, or from client certificates imported to browsers.

Enabling Certificate Authentication

To enable certificate authentication for users from a certain directory:

1. In the PrivX GUI on the Settings→Directories page, Edit a AD or LDAP directory to display its settings.

2. Expand Advanced directory settings. Then under Multi-Factor authentication settings, enable Client certificate authentication. You must also provide Trust anchors: the certificates of those CAs who issued the users' certificates.
Click **Save** to apply the new settings. Certificate authentication is enabled for users belonging to the directory.

**Logging In with Certificate Authentication**

After certificate authentication is enabled, users can log into PrivX as follows:

1. Ensure that your client certificate is available. For example, that your smart card is inserted and read properly, or your certificate is imported to the browser.

2. Access the PrivX-login page. If prompted by your browser, provide your client certificate.

   For successful login, the *Subject Alternative Name* in your client certificate must specify your *User Principal Name*.

After providing a valid client certificate you will be logged into PrivX.
Chapter 7 Authentication Methods for Host Connections

PrivX allows password authentication by default. This chapter describes how you may allow other authentication methods (such as certificate or public-key) for connecting to target hosts.

7.1 Supported Authentication Methods

PrivX users can authenticate to target hosts using the following methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Supported by</th>
<th>Notes</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certificate (for SSH connections)</td>
<td>SSH</td>
<td>Only for supported OpenSSH versions described in Section 7.2.1.</td>
<td></td>
</tr>
<tr>
<td>Public key</td>
<td>SSH</td>
<td>Not supported with agent-based SSH connections.</td>
<td></td>
</tr>
<tr>
<td>Stored password</td>
<td>SSH, RDP, Web</td>
<td>Not supported with agent-based SSH connections.</td>
<td></td>
</tr>
<tr>
<td>Certificate (for RDP connections)</td>
<td>RDP</td>
<td>Target hosts must satisfy the prerequisites from Section 7.5.1.</td>
<td>Not supported with native-client connections.</td>
</tr>
<tr>
<td>User-provided password</td>
<td>SSH, RDP, Web</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.1. Supported authentication methods

Certificate

Users are authenticated using just-in-time certificates. PrivX automatically issues certificates as needed; users do not need to input credentials when connecting.

Advantages: Automatically expiring credentials that are never exposed to users, making this method the most secure option.
Disadvantages: Allowed principals must be configured for target users, which is unwieldy in environments without automated configuration management. System times must be synchronized for certificate-based authentication to work correctly.

Public key

Users are authenticated using public keys provided by PrivX. Users do not need to input credentials when connecting.

Advantages: Public-key authentication is largely supported even on older and non-mainstream SSH servers.
Disadvantages: Public keys must be manually provisioned to target users on target servers. Public keys never expire, so they need to be manually renewed.

Stored password

Users are authenticated using passwords stored in PrivX. Users do not need to input credentials when connecting.

Advantages: Easy to set up.
Disadvantages: Weak passwords may compromise security.

User-provided password

Users are authenticated using passwords stored in PrivX.

Advantages: No additional target-host configuration required.
Disadvantages: Users must provide the target-user password when connecting. Weak passwords may compromise security.

Note

If a target host/target user supports multiple methods, the topmost supported method is used. User-provided password is enabled on all target hosts by default.

7.2 Enabling Certificate-Based Authentication for SSH Connections

You can enable certificate-based authentication on a target host with one of the following methods:

- Run the PrivX host-deployment script on the target host (Section 7.2.2)
- Manually set up OpenSSH server and allowed principals on the target host (Section 7.2.4)

7.2.1 Requirements for Certificate Authentication Over SSH

To use certificate-based authentication for SSH connections, hosts serving as connection endpoints (target hosts) must use an SSH server that supports OpenSSH certificates. The exact required version depend on the chosen authorization method. For additional information about the supported authorization methods, see Section 7.1.
<table>
<thead>
<tr>
<th>Authorization method</th>
<th>Required version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role-based access</td>
<td>OpenSSH 5.6 or later</td>
</tr>
<tr>
<td>Login as self</td>
<td>OpenSSH 6.9 or later</td>
</tr>
</tbody>
</table>

Table 7.2. SSH version requirements for certificate-based authentication

Determine the PrivX roles that can access the host, and the target users as whom they are granted login.

RDP certificate authentication is only supported through the PrivX GUI.

### 7.2.2 Script-Based Certificate-Authentication Setup

PrivX provides a host-deployment script. It is a Python script that can be executed on target hosts to automatically set up certificate authentication.

The host-deployment script has the following system requirements:

- **Supported operating systems**: CentOS, Fedora, Debian, Red Hat, Ubuntu.
- **Required Python version**: 2.7.9+ or 3.6.5+
- **Required OpenSSH version**: 6.9 or later.
- **Network**: Connectivity to a PrivX server.

**Note**

On target hosts that do not support the host-deployment script, and on hosts that cannot connect to PrivX, you can set up certificate-authentication manually as per Section 7.2.4.

**Note**

For hosts that cannot be accessed directly by PrivX, you need to first set up a PrivX Extender for proxying host connections. For more information about PrivX Extender setup, see Section 8.1.1.

To use the host-deployment script to set up certificate authentication on a target host, for connections between certain target accounts and PrivX users from certain roles:

1. To create a host-deployment script, navigate to the **Settings→Deployment→Deploy and configure SSH target hosts** page, select **Configure using a deployment script**, then click **Add Script**. Download the deploy.py script when prompted to.

2. Execute the deployment script on the target host as root. You will at least need to provide the host type, and the principals (specifying which target accounts can be accessed, and by whom).

To specify the host type, use one of the following options:

- **--aws**: The host runs on AWS.
- **--azure**: The host runs on Microsoft Azure.
- **--google-cloud**: The host runs on Google Cloud.
- **--openstack**: The host runs on OpenStack.
--standalone: For non-cloud hosts.

To specify principals, use any of the following options:

--personal-account-roles: Allow members of certain roles to log in with their personal accounts (to target accounts whose names match their Windows/Unix username).

--principals: Manually define target accounts and the roles that may access them.

For example, using --personal-account-roles to allow members of the roles Example Role 01 and Example Role 02 to log in with their personal accounts, add:

```
--personal-account-roles "Example Role 01,Example Role 02"
```

You may also specify --principals to define target accounts and the roles that may access them. The general format is:

```
--principals target_account1=role1,role2,...:target_account2=role3,role4,...:
```

For example, to allow target account alice to be accessed by PrivX users belonging to the privx-admin role, set this option to:

```
--principals alice=privx-admin
```

As another example, to additionally allow accounts alice and bob to be accessed by members of Example Role:

```
--principals alice=privx-admin,"Example Role":bob="Example Role"
```

Full syntax example of running the host-deployment script:

```
# /path/to/deploy.py --standalone \
--personal-account-roles "Example Role 01, Example Role 02" \
--principals alice="Example Role",privx-admin:bob=privx-admin
```

For a full list of options supported by the host-deployment script, run:

```
$ /path/to/deploy.py help
```

**Note**

If connections to the target host are relayed via a PrivX Extender, use the --api-hostname option to specify the Extender's address and port, similarly to the following:

```
# /path/to/deploy.py --standalone \
--personal-account-roles "Example Role 01" \
--api-hostname "extender.example.com:8443"
```

You may verify the addresses (subject-alternative names) of your PrivX Extender from its TLS certificate:

```
# openssl x509 -text -noout -in /opt/privx/extender/extender.crt \n  | grep -A 1 "Subject Alternative Name"
```
The PrivX Extender port is 8443 by default. You may verify the port from the Extender's configuration file, where it is specified by the `host_deployment_listen_address` variable.

3. Certificates issued by PrivX are very time-sensitive. Even a clock skew of few minutes may prevent certificates from working correctly.

Verify that the target-host time is correct. Adjust as necessary.

PrivX users belonging to one of the allowed roles should now see the target host on the Connections→Available Hosts page.

You may also verify that certificate-based authentication is used by checking the OpenSSH-server logs on the target server. Upon successful certificate-based authentication there should be a log message like the following:

```
Accepted publickey for alice from 192.0.2.26 port 50930 ssh2: RSA-CERT
ID alice@127.0.0.1:53188 serial 4920619392583124720
(serial 4920619392583124720)
```

**Configuring Access Using Host Tags**

For cloud-based hosts, PrivX can be configured to import allowed principals from host tags. This method simplifies host deployment as principals do not need to be provided for the host-deployment script.

**Tip**

Configuring access with host tags can be used for setting up hosts that mostly lack connection to PrivX: An initial instance must be able to connect to PrivX, but subsequent clones of this instance do not need PrivX connectivity.

To configure allowed principals for a cloud-based host using host tags:

1. Add host tags to the host via your cloud-host-management interface (such as EC2 for AWS instances). Host tags must specify the access rules and SSH/RDP services related to the host.

   The list of supported host tags is provided later in this section.

**Note**

When specifying tags for cloud hosts, do not include double-quotiation marks in the value.

2. Configure PrivX to import host tags. To do this, navigate to Settings→Directories in the PrivX GUI and Edit the directory to which the target host belongs to. Enable the `Import host instance tags from the directory` setting. Click Save to apply your changes.

3. Deploy the host using the host-deployment script, or manually.

   Users from PrivX roles are now able to access the host according to the principals defined in the host tags, with certificate-based authentication.
PrivX supports the following host tags:

**privx-ssh-principals=<target1>=<roles1>:<target2>=<roles2>:...**

Allow target accounts to be accessed by specified PrivX roles using SSH connections.

Access to each target account is specified with `<target1>=<roles1>` syntax, where `<target1>` is the account name and `<roles1>` is the comma-separated names of PrivX roles allowed to access the account. Each such mapping must be separated with a colon (`:`).

For example, to allow target user alice to be accessed by PrivX roles privx-admin and Role 01, and to allow target user bob to be accessed by Role 02:

```
privx-ssh-principals=alice=Role 01,privx-admin:bob=Role 02
```

**privx-rdp-principals=<target1>=<roles1>:<target2>=<roles2>:...**

Allow target accounts to be accessed by specified PrivX roles using RDP connections.

Syntax is the same as with the `privx-ssh-principals` tag.

**privx-ssh-personal-account-roles=<roles>**

Allow members of certain role(s) to log in with their personal accounts (to target accounts whose names match their Windows/Unix username) using SSH connections. Any comma-separated PrivX-role names in `<roles>` are allowed login to personal accounts.

For example, to allow members of Role 01 and Role 02 to login to personal accounts:

```
privx-ssh-personal-account-roles=Role 01,Role 02
```

**privx-rdp-personal-account-roles=<roles>**

Allow members of certain role(s) to log in with their personal accounts (to target accounts whose names match their Windows/Unix username) using RDP connections. Syntax is similar to the `privx-ssh-personal-account-roles` tag.

**privx-ssh-service-port=<port>**

Specify the SSH-server port of the target host. By default SSH servers run on port 22.

Example:

```
privx-ssh-service-port=22
```

**privx-rdp-service-port=<port>**

Specify the RDP-server port of the target host. By default RDP servers run on port 3389.

Example:

```
privx-rdp-service-port=3389
```

**privx-extender=<extendername>**

If connections to the host are relayed through a PrivX Extender, use this tag to specify the name of the Extender.
Example:

```
privx-extender=exampleExtender
```

For more information about PrivX Extender, see Section 8.1.1.

**privx-carrier=<carriername>**

Specify the name of the carrier that should be used as the default carrier for this host's web connections.

Example:

```
privx-carrier=exampleCarrier
```

**privx-enable-auditing=<yes|no>**

Toggle auditing for the host. Defaults to no.

Example:

```
privx-enable-auditing=yes
```

**privx-use-private-ips-only=<yes|no>**

Enabling this prevents using scanned public IP addresses as connection addresses for cloud hosts. This can be used to enforce connecting to hosts via private IP addresses only.

Example:

```
privx-use-private-ips-only=no
```

**Note**

To obtain the tag in key-value format, split the syntax at the first `=` sign. For example, the previously-provided examples are as follows in key-value format:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>privx-ssh-principals</td>
<td>alice=Role 01,privx-admin:bob=Role 02</td>
</tr>
<tr>
<td>privx-ssh-personal-account-roles</td>
<td>Role 01,Role 02</td>
</tr>
<tr>
<td>privx-ssh-service-port</td>
<td>22</td>
</tr>
<tr>
<td>privx-rdp-service-port</td>
<td>3389</td>
</tr>
<tr>
<td>privx-extender</td>
<td>example-extender</td>
</tr>
<tr>
<td>privx-enable-auditing</td>
<td>yes</td>
</tr>
<tr>
<td>privx-use-private-ips-only</td>
<td>no</td>
</tr>
</tbody>
</table>

**7.2.3 Certificate-Authentication Setup via Chef**

SSH Communications Security provides a public Chef cookbook `privx-chef-cookbook`. You can use it to set up nodes to trust PrivX certificates.

To obtain this cookbook, along with more information about its usage, visit:
7.2.4 Manual Certificate-Authentication Setup

To manually enable certificate-based authentication, you must configure your target hosts as follows:

1. Ensure the system clocks are synchronized (for example, using NTP). The certificates issued by PrivX have very short validity periods: clock skew may cause authentication to fail.

2. The SSH server on the target host must be configured to trust certificates issued by PrivX (the section called “Trusting PrivX Certificates”).

3. For target user accounts, specify the principals that are allowed to connect. Depending on the target-host OpenSSH version, you can do this in either of the following methods:
   
   - **For OpenSSH 6.9 or later**: Set up the `principals-command` script provided with PrivX. Allows PrivX users of selected roles to login with their personal accounts. Additionally, you can define the target accounts and the roles that may access them (the section called “Allowing Login as Self and Role-Based Access”).
   
   - **For OpenSSH 5.6 or later**: Configure allowed principals per target account. You can define the target accounts and the roles that may access them (the section called “Allowing Role-Based Access”).

4. To display the target hosts and accounts under known targets in PrivX, add the host to PrivX (Section 8.1).

Principals in PrivX-Issued Certificates

This section describes *principals* in the context of certificate-based authentication.

OpenSSH certificates contain principals indicating who is requesting access. OpenSSH certificates issued by PrivX contain at least the user name as a principal, and may have any number of other principals. For example, assume a PrivX user named *alice* belongs to roles having the principals:

```
bd5eab79-fe35-4f52-839a-05737075b152
b8b72199-689d-5a9b-782f-8e1069f7db19
```

When PrivX user *alice* connects to a target host using certificate authentication, PrivX issues them a certificate with the following principals:

```
bd5eab79-fe35-4f52-839a-05737075b152
b8b72199-689d-5a9b-782f-8e1069f7db19
alice
```

Authentication succeeds if the target account allows at least one of the principals.

Trusting PrivX Certificates

Target hosts using certificate-based authentication must be configured to trust the certificates issued by PrivX.
1. In the PrivX GUI, go to the **Settings→Deployment→Deploy and Configure SSH target hosts** page. Select **Configure manually**. Acquire the PrivX CA key.

![PrivX CA key](image)

**Figure 7.1. PrivX CA key**

Copy the PrivX CA key. This key is later required for configuring the target host.

2. On the target host, create the following file:

```
/etc/ssh/privx_ca.pub
```

Copy and save the PrivX CA key into this file.

Ensure that the CA-key file is owned and readable by the user running the SSH server (typically root):

```
# chown root:root /etc/ssh/privx_ca.pub
# chmod 0400 /etc/ssh/privx_ca.pub
```

3. In the SSH server configuration file (located at `/etc/ssh/sshd_config` by default), specify `TrustedUserCAKeys` to point to the PrivX CA key. To do this, add a line similar to the following into the SSH server configuration:

```
TrustedUserCAKeys /etc/ssh/privx_ca.pub
```

Restart the SSH server to apply the new settings.

4. Certificates issued by PrivX are very time-sensitive. Even a clock skew of few minutes may prevent certificates from working correctly.

Verify that the target-host time is correct. Adjust as necessary.

### Allowing Login as Self and Role-Based Access

PrivX provides a principals-command script that enables *login with personal accounts*: PrivX users can log into the target accounts, the names of which match the PrivX user’s Windows/Linux username.

**Note**

The principals-command script is only supported on target hosts running OpenSSH 6.9 or later.

The provided principals-command script is provided as an example. You may modify the script to suit your own needs.

To set up the principals-command script, enable login with personal accounts, and to configure role-based access:
1. PrivX users who are to access the target host must belong to at least one PrivX role. If you have not done so already, assign users to roles as described in Section 6.2.1.

   Also ensure that PrivX users are set with correct Windows username and Linux username. By default, AD users’ Windows username and Linux username are set to match their User-Principal Name and sAMAccountName respectively.

2. For security reasons we recommend creating a user specifically for running the principal-command script on the target host. This AuthorizedPrincipalsCommandUser should be an unprivileged user who only has permissions for performing the following:

   • Running the principals-command script.
   • Reading the principals that are allowed to log in with personal accounts, at /etc/ssh/personal_account_roles
   • Reading any user-specific authorization-principals files located at /etc/ssh/auth_principals/%u (replace %u with the name of the authenticated user).

   Some Unix distributions have a default nobody account, which can be used as the AuthorizedPrincipalsCommandUser.

3. Obtain the AuthorizedPrincipalsCommand script from your PrivX server, at the following path:

   /opt/privx/authorizer/principals_command.sh

   Copy the script to the target host. In this example, we assume the script is copied to

   /etc/ssh/principals_command.sh

   Both the script and the directory it is in must be owned by root. The script must also be readable and executable by the AuthorizedPrincipalsCommandUser (replace /etc/ssh/principals_command.sh with the path to which you copied the script):

   ```
   # chown root:root /etc/ssh
   # chown root:root /etc/ssh/principals_command.sh
   # chmod 0755 /etc/ssh/principals_command.sh
   ```

4. Configure SSH server to run the principals-command script as the AuthorizedPrincipalsCommandUser. This is done by adding lines similar to the following into the SSH configuration (by default at /etc/ssh/sshd_config, replace authprincipalsuser with the name of your designated AuthorizedPrincipalsCommandUser, replace /etc/ssh/principals_command.sh with the path where you copied the script):

   ```
   AuthorizedPrincipalsCommand /etc/ssh/principals_command.sh %u
   AuthorizedPrincipalsCommandUser authprincipalsuser
   ```

   Save your changes to the SSH server configuration file. Restart the SSH server to apply the changes.

5. Next, specify who is allowed to access target accounts. Note that all access must be specified in a role-based fashion.
To allow login with personal accounts for members of certain roles: On the target host, specify in `/etc/ssh/personal_account_roles` which PrivX-role members are allowed to log in with personal accounts. The file should specify one UUID per line, similar to the following:

```
269eae0f2-482e-5b44-5d0f-2ee4c37e2877 # Example Admins
31f070c6-33ff-523f-4cea-d586ab25071a # Example Role
8a0c0cbb-1417-4de2-8532-3eac1f6c30b # Example Users
...
```

To determine the UUIDs of PrivX roles. Go to **Settings→Deployment→Deploy and Configure SSH target hosts** page, then select *Configure manually*.

Also ensure the file is readable by root and the `AuthorizedPrincipalsCommandUser`:

```
# chown root:root /etc/ssh/personal_account_roles
# chmod 0604 /etc/ssh/personal_account_roles
```

PrivX users belonging to the specified roles may now log into the host as themselves. For example, a PrivX user belonging to the `Example Admins` role (with role UUID `269eae0f2-482e-5b44-5d0f-2ee4c37e2877`) with the *Linux username* `alice` may now log in to the target-account named `alice` (assuming the host is Unix/Linux-based).

ii. **To grant custom role-based access to certain target accounts**: Create the directory `/etc/ssh/auth_principals/` for holding target-account-specific principals files (if not already done):

```
# mkdir /etc/ssh/auth_principals
# chown root:root /etc/ssh/auth_principals/
# chmod 0755 /etc/ssh/auth_principals/
```

For each target account requiring custom role-based access, create a principals file under `/etc/ssh/auth_principals/`. The principals file must have the same name as the target account. For example, when setting up custom role-based access for a target account named `alice`, the principals file is `/etc/ssh/auth_principals/alice`

In the principals file, specify the roles that may access the account. The file should specify one UUID per line, similar to the following:

```
269eae0f2-482e-5b44-5d0f-2ee4c37e2877 # Example Admins
31f070c6-33ff-523f-4cea-d586ab25071a # Example Role
8a0c0cbb-1417-4de2-8532-3eac1f6c30b # Example Users
...
```
After saving your changes to the file, ensure it is readable and executable by both root and the AuthorizedPrincipalsCommandUser (replace /etc/ssh/auth_principals/alice with the path to the user's principals file):

# chown root:root /etc/ssh/auth_principals/alice
# chmod 0705 /etc/ssh/auth_principals/alice

6. Certificates issued by PrivX are very time-sensitive. Even a clock skew of few minutes may prevent certificates from working correctly.

Verify that the target-host time is correct. Adjust as necessary.

You may verify that certificate-authentication is used by checking the OpenSSH-server logs on the target server. Upon successful certificate authentication there should be a log message like the following:

Accepted publickey for alice from 192.0.2.26 port 50930 ssh2: RSA-CERT \nID alice8127:0.0.1:53188 serial 4920619392583124720 \n{serial 4920619392583124720} \nCA RSA 9b:16:36:bf:6e:c6:3f:e5:a1:5e:31:61:c1:37:ef:d8

Allowing Role-Based Access

PrivX roles have unique principals. When a target user allows a role principal, members of corresponding roles can connect to the target user.

To allow the members of a PrivX role to access a target user:

1. On the target host, create a directory for storing principals files (if not already done).

   # mkdir /etc/ssh/auth_principals

   The permissions of the directory must be set so that individual target users are able to read their authorization-principals files, for example:

   # chown root:root /etc/ssh/auth_principals
   # chmod 0755 /etc/ssh/auth_principals

2. On the target host, create a file for storing the user-specific allowed principals (replace %u with the account name of the target user on the target host):

   /etc/ssh/auth_principals/%u

   For example, if the target-user name is alice, their principals file is:

   /etc/ssh/auth_principals/alice

   Within the file, specify the UUIDs of the roles that are allowed access to the target-user account. Specify each UUID on its own line, similar to the following:

   269eaf2-482e-5b44-5d0f-2ee4c37e2877 # Example Admins
   31f070c6-33ff-523f-4ce-a5b86ab25071a # Example Role
   8a0c0cbb-1417-4de2-8532-3eac1fb4c30b # Example Users
   ...
To determine the UUIDs of PrivX roles, go to Settings→Deployment→Deploy and Configure SSH target hosts page, then select Configure manually.

3. Ensure that the authorization-principals file is owned and readable by the target-user account (replace \%u with the name of the target-user account):

```
# chown \%u:%u /etc/ssh/auth_principals/%u
# chmod 0400 /etc/ssh/auth_principals/%u
```

For example, if the target-user name is alice:

```
# chown alice:alice /etc/ssh/auth_principals/alice
# chmod 0400 /etc/ssh/auth_principals/alice
```

4. If not already done, instruct the SSH server about where the authorization-principals files can be found from. This can be done by specifying the AuthorizedPrincipalsFile setting in the OpenSSH server configuration (by default at /etc/ssh/sshd_config):

```
AuthorizedPrincipalsFile /etc/ssh/auth_principals/%u
```

Save your changes to the OpenSSH server configuration. Restart the OpenSSH server to apply the changes. PrivX users belonging to one or more allowed principals (roles) are now authenticated to the target-user account using certificates.

5. Certificates issued by PrivX are very time-sensitive. Even a clock skew of few minutes may prevent certificates from working correctly.

Verify that the target-host time is correct. Adjust as necessary.

You may verify that certificate-authentication is used by checking the OpenSSH-server logs on the target server. Upon successful certificate authentication there should be a log message like the following:

```
Accepted publickey for alice from 192.0.2.26 port 50930 ssh2: RSA-CERT \nID alice@127.0.0.1:53188 serial 4920619392583124720 \serial 4920619392583124720) \CA RSA 98:16:36:bf:6e:c6:3f:e5:a1:5e:31:61:c1:37:ef:d8
```

### 7.3 Enabling Role-Based Public-Key Authentication

1. Obtain the public key of the role you want to authorize. You can do this on the Settings→Deployment→Deploy and Configure SSH target hosts page.
2. Authorize the public key to the target user on the target host. In OpenSSH for example, this is done by appending the public key to the authorized_keys file, typically located at:

```
~/.ssh/authorized_keys
```

All the members of the role should now be able to connect to the target user with public-key authentication.

### 7.4 Enabling Authentication with Stored Passwords

You can store target-user passwords in PrivX, so PrivX users can connect to those target users without having to provide the password. The steps for enabling this authentication method are:

1. **For SSH only**: Configure the target-host SSH server to allow password authentication. Note that allowing keyboard-interactive authentication does not enable using stored passwords in PrivX.

2. Store the target-user passwords in PrivX:
   i. On the Settings→Hosts page, Edit the target host on which the target user is.
   ii. Under the Accounts section, provide the Password for the target user(s). If not done already, also specify the Roles that may access the target user(s)
   iii. Click Save to apply the changes.

Connections to the target user should no longer prompt for password.

### 7.5 Enabling Certificate Authentication for RDP Connections

This section describes the procedures for enabling certificate authentication for RDP connections.

Before setting up certificate authentication for RDP connections, ensure the prerequisites described in Section 7.5.1.

#### 7.5.1 Requirements for Certificate-Based Authentication Over RDP

Before enabling certificate authentication for RDP, check and execute the following:
• Target hosts must belong to a Windows domain. The domain must include:

• A domain controller with the server role *Active Directory Domain Services*, for handling authentication requests.

• A Certificate Authority Server (CA server), with the server role *Active Directory Certificate Services*, including the role service *Certificate authority*.

The CA server must also have a service for certificate-revocation-status checks, for example, HTTP CRL with *Certificate Enrollment Web Service* or *Web Server IIS* role, or alternatively OCSP with the *Online Responder* role service.

• Ensure that users' group policy allows RDP login. When enabling login with personal accounts, also ensure target-host local policy allows users to log on locally.

• Both the domain controller and the CA server must run on one of these supported platforms:

  • Windows Server 2008 R2 (with the latest service pack and updates)
  • Windows Server 2012 R2 (with the latest service pack and updates)
  • Windows Server 2016 (with the latest service pack and updates)

• The domain policy must enable *server certificate auto-enrollment*. For instructions about enabling this, please refer to Microsoft documentation at [https://docs.microsoft.com](https://docs.microsoft.com) (search title: configure server certificate autoenrollment).

• Firewalls for the domain must allow HTTP access to PrivX server port 80, for obtaining the Certificate Revocation List.

• PrivX server's IPs and FQDNs should be recorded in the *shared-config.toml* file. All listed IPs and FQDNs will be used as Certificate Revocation List Distribution Points.

• Hosts in the target domain must be able to resolve PrivX server FQDNs.

**7.5.2 Steps for RDP Certificate-Authentication-Setup**

After ensuring the prerequisites, enable certificate authentication for RDP by performing the following:

1. For target hosts to trust PrivX certificates, you must publish the PrivX CA certificate in the Windows domain.

   To obtain the PrivX CA certificate, go to the PrivX GUI. On the *Settings→Deployment→Configure a Windows Domain for RDP Access* page, click *Download Certificate*.

2. Add the PrivX CA certificate to the Trusted Root Certification Authorities for the domain.

   For improved security, also restrict the purposes of the PrivX CA: In the general properties of the PrivX CA certificate, select *Enable only the following purposes*, then select the following purposes:

   • Smart Card Logon
• Client Authentication

Save your changes to the certificate.

For more information, please refer to Microsoft documentation at https://docs.microsoft.com (search title: distribute certificates to client computers by using group policy).

3. Publish the PrivX CA certificate to the domain (replace privx_ca.crt with the path of the PrivX CA-certificate file):

   $ certutil -dpublish -f privx_ca.pem NTAuthCA

   Also ensure that the registry is updated by running:

   certutil -addstore -enterprise NTAuth privx_ca.pem

4. On all target hosts, ensure that the host allows remote connections without Network Level Authentication.

5. Define which roles are allowed to access the target hosts, and as which target accounts. For more information about mapping roles to target accounts, see Section 8.1.

6. Certificates issued by PrivX are very time-sensitive. Even a clock skew of few minutes may prevent certificates from working correctly.

   Verify that the system times on the target hosts is correct. Adjust as necessary.

RDP connections to target hosts are now be authenticated by just-in-time certificates provided by PrivX, without needing to provide target-user passwords.

### 7.6 Trusting Target-Host Identities

This section describes target-host authentication.

#### 7.6.1 SSH Target-Host Authentication

For SSH connections, PrivX authenticates target hosts by their SSH host keys.

You can store SSH host keys when creating or editing a host entry via Settings→Hosts.

When connecting to a host, the key/certificate of which is not yet trusted by PrivX, the connection starts as follows:

• **If no other keys are stored for the host**: PrivX administrators may accept the new host key for all subsequent connections. Regular PrivX user can accept a new host key if the Trust On First Use is enabled for the host.

• **If another key has been stored for the host**: Regular PrivX users are prevented from connecting to the host. A PrivX administrator must explicitly accept the key for all subsequent connections.
You can store SSH host keys and set Trust on First Use behavior by editing host entries on the Settings→Hosts page.

### 7.6.2 RDP Target-Host Authentication

For RDP connections, PrivX authenticates target hosts by their RDP-server certificate.

By default, PrivX automatically accepts the certificate encountered upon the first RDP connection. Connections will fail if the RDP-server certificate changes, or is renewed prior to the start of the renewal period. The renewal period starts one month prior to the expiry date. Connection will automatically accept the new certificate during the renewal period.

If connections fail due to the RDP-server certificate changing, you can re-enable by deleting the stored RDP certificate:

1. On the Settings→Hosts page, Edit the target RDP server.

2. Under the RDP host certificate section, click

   ![RDP host certificate link]

   Upon next RDP connection, PrivX automatically accepts the new RDP-server certificate for subsequent RDP connections.

You can adjust the time window during which the RDP-server certificate may be automatically renewed. To do this:

1. Modify the RDP proxy settings, located at the following path on your PrivX servers:

   ```
   /opt/privx/etc/rdp-proxy.toml
   ```

   Locate and modify the following settings:

   - **renewal_period_months**: Allow RDP-server certificate to be renewed from this many months before the NotAfter field marked in the current RDP-server certificate.

   - **renewal_period_days**: Allow RDP-server certificate to be renewed from this many days before the NotAfter field marked in the current RDP-server certificate.

   These two settings are cumulative. For example, to allow the RDP-server certificate to be renewed from two months and 15 days before the current one expires:

   ```toml
   [certificates]
   update_automatically = true
   renewal_period_months = 2
   renewal_period_days = 15
   ```

2. Save your changes to the RDP-proxy configuration.

   Restart the PrivX services to apply the new settings:

   ```
   # systemctl restart privx
   ```
7.7 Authentication to AWS Services

You can use PrivX to authenticate and authorize users of the AWS Command Line Interface (AWS CLI). PrivX issues short-term credentials to allow users to perform their tasks, which removes the need for manually provisioning public keys or other persistent credentials.

7.7.1 Prerequisites

- PrivX grants access to AWS services via the AWS Command Line Interface (CLI account). The person receiving access must have a Unix or Mac account with AWS CLI set up.

- The CLI account must be set up with PrivX agents. For instructions about setting up and using PrivX agents, see Section 8.4.

- Verify the system time on the host with the CLI account. Short-term credentials issued by PrivX are very time-sensitive. Even a clock skew of few minutes may prevent successful authentication.

7.7.2 PrivX Authentication to AWS Services

Set up AWS users and roles for delegating AWS-service permissions, then grant these roles to PrivX users:

1. In your AWS management interface, create an IAM user for obtaining AWS roles. To do this, provide the AWS user with the following permission policies:

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": "sts:GetFederationToken",
         "Resource": [ "*" ]
      },
      {
         "Effect": "Allow",
         "Action": "sts:AssumeRole",
         "Resource": [ "*" ]
      },
      {
         "Effect": "Allow",
         "Action": [ "iam:ListRoles",
                     "iam:ListAttachedRolePolicies",
                     "iam:GetPolicy",
                     "iam:GetPolicyVersion",
                     "iam:ListPolicies",
                     "iam:ListPolicyVersions",
                     "ec2:DescribeInstances",
                     "ec2:DescribeRegions" ],
         "Resource": [ "*" ]
      }
   ]
}
```
Note

The `ec2:Describe*` permissions are only required for scanning AWS hosts, which is functionality associated to all AWS directories in PrivX. The `ec2:Describe*` permissions may be omitted if you only want to import AWS roles, however note that doing so results in the AWS directory in PrivX having a constant error status.

PrivX uses this AWS user to import and delegate AWS roles. For this reason you must also create a Secret Access Key for the user. Copy the Access Key ID and the Secret Access Key of the user, which are required later for configuring PrivX.

2. In your AWS management interface, create AWS role(s) that grant access to AWS services. The AWS role(s) should have permissions like the following (replace the example Actions with services and actions this role should allow access to):

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": [ "*" ]
    }
  ]
}
```

Also set the trust relationships of the AWS role similar to the following (replace `<iam_user_arn>` with the ARN of the previously-created IAM user):

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": `<iam_user_arn>`
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

Note

To allow CLI users to assume the role via PrivX, the AWS role must have a unique Name tag.
Tip

Place PrivX-specific roles under a unique path (such as /privx/). PrivX can be configured to import roles from this path, ignoring irrelevant roles in your AWS environment.

3. On the Settings→Directories page of the PrivX GUI, add a directory of the type Amazon Web Services. Provide the Access key ID and Secret Access Key of the previously-created IAM user.

Also under Advanced AWS settings, select Enable AWS role federation. If you only want to import roles under a certain path, set this path to AWS role prefix for federated roles

Save the directory configuration. To verify that roles were imported successfully, ensure that the previously-configured AWS roles are visible on the page Settings→Deployment→Configure Amazon Web Service role grants.

4. Grant AWS roles to PrivX roles. To do this, go to the page Settings→Deployment→Configure Amazon Web Service role grants, and Edit the AWS roles. PrivX users belonging to specified roles can now gain access according to any AWS roles granted to them.

7.7.3 Using AWS Services

After you have set up access to AWS services (as per Section 7.7.2), you can use the services authorized to you as follows:

1. Log into the CLI account on your workstation, activate the PrivX agent if necessary.

2. Use the PrivX agent to log into PrivX (replace username with your PrivX-user name):

   $ privx-agent-ctl login username

3. Determine and assume an AWS role available for your account (replace example_aws_role with the name of the role you want to assume):

   $ privx-agent-ctl aws-role list
   $ privx-agent-ctl aws-role set example_aws_role

   You may verify that you have successfully assumed the role by listing the AWS roles. It should be marked with an asterisk (*). For example:

   $ privx-agent-ctl aws-role list
   Available AWS roles:
   example_aws_role_0
   *example_aws_role
   example_ws_role_2
   ...

   You are now permitted to run any AWS services permitted by the AWS role. For example, if the AWS role gives you read access to the s3 service, you can now run:

   $ aws s3 ls s3://bucket.example.com/
Chapter 8 Establishing and Managing Connections

This chapter describes:

• Adding targets for users to connect to.

• Different ways in which PrivX users can connect to targets. Additional setup instructions for methods that require them.

• Proxying connections, for targets not directly accessible from PrivX servers.

8.1 Setting up Known Targets

This section describes manual target setup. For information about importing targets from existing directories, see Section 8.3 instead.

PrivX users can connect to accounts and hosts that have been set up as known targets.

You can add known targets to PrivX by adding hosts. To do this, go to Settings→Hosts and click Add Host. PrivX host entries are used for specifying, among other things:

• The host address.

• Who should be allowed to access the host, and as what identity. For example, allow members of Example Role 01 to log in as exampleuser on the host.

• What methods (host services) can be used to access the host: SSH, RDP, and Web.

Known targets offer the following benefits:

• To see and connect to known targets, PrivX users can go to the Connections page and select Search from known hosts. PrivX users will only see those known targets that they are authorized to.

• Known targets support additional authentication methods (instead of just user-provided password). For more information about the configurations required for additional authentication methods, see Chapter 7.
Chapter 8 Establishing and Managing Connections

- PrivX can periodically check the status of known targets and indicate when targets are unreachable. Settings related to status checks are under the [health-check-options] section in the host-store configuration /opt/privx/etc/hoststore.toml

Note
All known targets must have unique machine IDs: If you need to add cloned machines to PrivX, ensure the uniqueness of the machine's ID and regenerate it if necessary.

8.1.1 Proxying Connections to Hosts

For target hosts that are inaccessible from PrivX servers (such as hosts in protected networks), you may set up PrivX Extenders for relaying connections.

Any target host accessed via an Extender must specify the Extender name in its Address. Example Address syntax in IPv4 and IPv6 format:

```
exampleextender/192.0.2.100
exampleextender/2001:DB8::64
```

Save your host changes. Subsequent connections to the host are proxied via the PrivX Extender.

For more information about PrivX-Extender requirements and setup, see Section 2.1.3 and Section 3.2.1 respectively.

8.1.2 SSH Targets

Note
You can use the PrivX host-deployment script to automatically add SSH targets (while also enabling certificate-based authentication). For more information about using the host-deployment script, see Section 7.2.

To allow connections to SSH hosts, add a host entry with the following considerations in mind:

- Add a Service with the type SSH, specifying the address and the port of the SSH server on the host.

  When the Trust on first use option is disabled, SSH host keys must also be added here. In this case regular PrivX users can connect only if the host key matches one of the provided values. PrivX administrators can establish connections even if the host key is missing or incorrect.

- Add Accounts specifying which PrivX roles may access the host, and which target accounts they are given access as.

  The access options of the accounts can be limited under the Allowed Service Options:

  Shell

  Shell/terminal access.
**RDP Targets**

File Transfer

File transfers through `subsystem/sftp` and `exec/scp` channels.

Exec

All `exec` channel functions except `exec/scp`.

Tunnels

`direct-tcpip` channel for local and `forwarded-tcpip` channel for remote port forwarding.

X11 Forwarding

`x11` channel for X Window System graphical access.

Other

Other SSH channels.

By default all **Allowed Service Options** are enabled. The default values can be edited in the Host Store configuration file, located under the following path on PrivX servers:

```
/opt/privx/etc/hoststore.toml
```

### 8.1.3 RDP Targets

To allow connections to RDP hosts, add a host entry with the following considerations in mind:

- Add a **Service** with the type **RDP**, specifying the address and the port of the RDP server on the host.
- Add **Accounts** specifying which PrivX roles may access the host, and which target accounts they are given access as.

The access options of the accounts can be limited under the **Allowed Service Options**:

- File Transfer
- Audio
- Clipboard

By default all **Allowed Service Options** are enabled. The default values can be edited in the Host Store configuration file, located under the following path on PrivX servers:

```
/opt/privx/etc/hoststore.toml
```

You may also provide **Windows application restrictions**, for limiting users to certain applications. Note that the target Windows host must be configured to allow the listed applications to be used over RDP.

### 8.1.4 Web Targets
You can use PrivX to connect to websites. To allow connections to websites, add a host entry with the following considerations in mind:

- Add a **Service** with the type **Web**, specifying the address of the website.

**Note**

Since connections to web targets are provided via a PrivX Web Proxy, you need to provide the address in proxy format. For example (replace `exampleproxy` with your Web access gateway name, and `https://www.example.com/` with the address of the website):

```
exampleproxy/https://www.example.com/
```

If you want PrivX to automatically fill in login credentials for the website, also provide the following **Additional settings**:

- **Login-request address**: The verified address of the login request. For example, in form logins this may be the URL of the webpage plus the URL specified by the `action` attribute of the form. Recommended for improved security.

- **Password property**: The verified `id` of the password field in the login form. Recommended for improved security.

- **Login-page address**: The login-page address. Only needed if the login page is not under the `Address` of this web service. For example, while you could have an AWS service with the `Address`:

```
exampleproxy/https://example.signin.aws.amazon.com/console
```

That website may redirect you to a different address for login:

```
https://us-east-1.signin.aws.amazon.com
```

- **Authentication type**: Set to **Automatic** for most websites, such as websites using forms for authentication. Set to **Basic** for websites using the **Basic HTTP Authentication Scheme** (defined in RFC 7617).

- **Username-field name**: The `name` of the username field in the login form. Only required if PrivX is unable to automatically detect this field.

- **Password-field name**: The `name` of the password field in the login form. Only required if PrivX is unable to automatically detect this field.

- Add **Accounts** specifying which PrivX roles may access the website. If you want PrivX to automatically fill in login credentials for the website, also provide **Usernames** and **Passwords** in the account mappings.

The access options of the accounts can be limited under the **Allowed Service Options**:

- File Transfer
- Audio
• Clipboard

By default all Allowed Service Options are enabled. The default values can be edited in the Host Store configuration file, located under the following path on PrivX servers:

```
/opt/privx/etc/hoststore.toml
```

**Note**

When set up to do so, PrivX automatically fills up login credentials, but will not actually log into the web service. Users must manually click any Login buttons to log in.

For HTTPS targets, by default PrivX only allows connections using TLSv1.2 and later.

### 8.2 Connecting via the PrivX GUI

To connect to targets via the PrivX GUI:

1. Log into the PrivX GUI using your PrivX account.

2. On the Home page under New Connection, provide target details (such as the target-host address) to receive suggestions about available targets, then select a target to connect to it. Your accessible targets are also listed and selectable on the Connections→Available Hosts page.

   You may alternatively go to Connections→Manual Connection page and manually specify your target. The user must have the connections-manual permission to use this method.

**Note**

Connections are authenticated according to the role-based rules and the enabled authentication methods.

You may need to adjust locale settings for keyboard commands to be transferred correctly.

**Tip**

Quickly access a recent target from the Home page by clicking an entry under Recent connections, or on the Connections→Connection History page.

### 8.2.1 SSH GUI Features

#### Open multiple terminals

On the tab bar, click + to open additional terminals.

#### Copy-paste

- On a Mac, you can select text and use command-c and command-v to copy-paste, or right-click and use the context menu.
• On a Windows or Linux you can select text and right-click to use the context menu. Alternatively you can select some text, then press Ctrl-C or Ctrl-Insert to copy. Paste using Ctrl-V or Shift-Insert.

**Note**

To use applications that specifically require Ctrl-V for other things than pasting, enable *Send Ctrl-V to server* under terminal **Settings**. Enabling this setting also disables pasting with Ctrl-V.

• The left Alt (option on a Mac) key also functions as the Meta key (for example in Emacs).

**Transfer files**

You can upload, download, and remove files from the *File Transfers* tab.

To change directory, click a directory or . You can perform additional actions on files via their menus.

• **To download a file**: Double-click the file, or perform a Download action via its menu.

• **To upload a file**: Drag-and-drop the file to the file-transfer view. Alternatively, click **Upload next** to the file-transfer view and select the local file to upload.

**Change text, locale and theme**

From the **Settings** tab, you can set the font size, character encoding, and the locale used for the connection. You can also switch between a dark and a light theme.

**Open URLs**

To open HTTP/HTTPS links in a new tab on Windows or Linux, hold Ctrl and double-click the link. To open links on Mac, hold command and double-click.

### 8.2.2 RDP GUI Features

**Copy-paste**

To copy-paste text to the target host, place the text into the **Clipboard** tab, then paste normally on the target host. Text copied on the target host is automatically made available in the **Clipboard**.

**Transfer files**

Upon connecting, PrivX automatically mounts a network drive called *Transfers on PrivX* to the target host. File downloads and uploads must be performed via this network drive.

You can copy files from the target host to *Transfers on PrivX*, then download them from the **Files** tab. Similarly, you can upload files by placing them to *Transfers on PrivX* on the **Files** tab, then move the file to another location on the target host.
• **To download a file**: Double-click the file, or perform a Download action via its menu.

• **To upload a file**: Drag-and-drop the file to the file-transfer view. Alternatively, click **Upload** next to the file-transfer view and select the local file to upload.

Set keyboard layout

To change your keyboard layout, go to the **Settings** tab, select your layout, then restart the connection to apply the changes. Set this to match the keyboard layout defined for the server (not the client-side layout).

Fullscreen view

To toggle fullscreen view, click

![Note]

Toggling fullscreen mode also re-establishes the RDP connection.

### 8.3 Importing Known Targets from Directories

This section describes automatic target import. For information about manually setting up targets, see Section 8.1 instead.

You can set up PrivX to automatically add existing hosts from cloud platforms. Such hosts can later be connected to via PrivX.

For example, to add hosts from Amazon Web Services (AWS):

1. In your AWS, add a policy to allow host scans. To do this, access your AWS and navigate to **IAM→Policies**, then create a policy with the following JSON:

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "***"
        }
    ]
}
```
2. Create an IAM user with permissions to use the host-scan policy. This can be done on the IAM→Users page.

The IAM user must have Programmatic access, and be attached with the host-scan policy.

Note the Access key ID and the Secret access key of the user. These are required later for configuring PrivX against AWS.

3. Configure PrivX to scan and add the AWS hosts.

Log into PrivX as superuser (or other privx-admin user). On the Settings→Directories page, click Add Directory.

Fill in the basic information of the directory. To allowPrivX to detect AWS hosts, add the Access key ID and the Secret access key of the IAM user.
Save the directory settings. PrivX begins importing hosts from AWS.

After a moment, you may verify the directory status back on the Settings→Directories page. The Connection should be in the OK state, and list the number of instances found on AWS.

To list the imported hosts, click List Hosts.

You may then Edit hosts to add services and account mappings to them, from Settings→Hosts page.

8.3.1 Removing Hosts from Directories

When PrivX notices that non-local hosts were deleted at the directory source, PrivX marks the associated host entries as deleted. After deleted entries reach a certain age, they are removed completely from PrivX.

The interval is controlled by the following settings in role-store settings /opt/privx/etc/host-store.toml

• host_housekeeping_run_interval - The interval at which PrivX checks for and removes old deleted-host entries. 168 hours (1 week) by default.

• hosts_deleted_age - The duration for which deleted-host entries are preserved. 168 hours (1 week) by default.

8.4 SSH Connections with Native Clients
This section describes how to establish SSH connections with native clients.

Users can connect to target hosts/accounts using the SSH clients installed on their workstations, without needing to use the PrivX GUI. Connections are authenticated against PrivX. For example, if PrivX allows you to access a target account with certificate authentication, your native clients will also connect using certificate authentication without prompting you for target-account credentials.

### 8.4.1 Prerequisites

- Specify which roles may be used for SSH-native-client connections.

  When using the GUI, PrivX users have access to targets provided by any of their roles. However with SSH-native-client connections, access is further limited to roles with the *Allow PrivX agent* option.

  The *Allow PrivX agent* option is enabled at **Settings → Roles**, under the *SSH Options* of each role.

- Your PrivX license must allow using native client connections. For more information about licenses, see Section 3.3.

- After upgrading PrivX, you may need to re-download the extender-config.toml file for the native client connections to work with via Extenders. For more instructions about extender configuration file, see Section 3.2.1.

- For agent-based connections only: PrivX agent must be set up on the user’s workstation. For instructions setting up PrivX agents, see Section 3.2.3.

### 8.4.2 Connecting with Native SSH Clients using PrivX Bastion

Using native SSH clients, you can connect to targets via PrivX SSH Bastion. PrivX SSH Bastion provides the following connection modes:

- **Interactive**: Access PrivX Bastion to list and select possible targets.

- **Direct**: Specify your connection target directly to the native client.

#### Note

PrivX-Bastion connections are verified against the PrivX-Bastion host key. You may verify and install these host keys from the **Connections → Native Clients** page.

### Connecting Interactively

To connect via PrivX Bastion interactively:

1. Connect to PrivX SSH Bastion using your PrivX account. For example, with `ssh`, `sftp`, or `scp`:

```
ssh -p 2222 privxuser@privx.example.com
```
Replace the example values as follows:

- `privxuser` - Your PrivX-user name.
- `privx.example.com` - Your PrivX-server address.
- `local/path` - Local file/directory path for scp.
- `remote/path` - Remote file/directory path for scp.

Provide your PrivX-user password when prompted.

2. You will be presented with a list of possible targets. Select a target to connect to it.

**Connecting Directly**

To directly connect via PrivX Bastion, provide:

- Target-user name
- Target-host address
- PrivX-user name
- PrivX-server address
- *(Optional)* Extender name
- *(Optional)* Target port

By default bastion runs on your PrivX servers, port 2222.

Full bastion syntax is as follows:

```
targetuser%extender%targethost%targetport%privx-user@privx.example.com
```

Common case leaves out the extender and the target port, leaving the syntax as following:

```
targetuser%targethost%privx-user@privx.example.com
```

Following are examples of ssh, scp and sftp usage with the connection string:

```
ssh -p 2222 targetuser%targethost%privx-user@privx.example.com
scp -P 2222 targetuser%targethost%privx-user@privx.example.com:example.txt /target/directory
sftp -P 2222 targetuser%targethost%privx-user@privx.example.com
```

Following is an example using PrivX Extender:

```
ssh -P 2222 targetuser%extender%targethost%privx-user@privx.example.com
```
scp -P 2222 example.txt \
targetuser%extender%targethost%privx-user@privx.example.com:/tmp

If you use native-client connections with bastion syntax often, consider specifying the connection parameters in the users’ client configuration (typically at /etc/ssh/ssh_config or ~/.ssh/config) using Host blocks. For example:

```
Host targethost.example.com
  Port 2222
  User targetuser%targethost%privx-user
  Hostname privx.example.com
```

After which you can connect with much simpler syntax:

```
$ ssh targethost.example.com
```

User sessions with native SSH clients can be monitored. For more information about viewing session audit data, see Section 9.1. For more information about setting up session recording for a host, see Section 9.3.

Optionally you can use the PrivX agent to connect using the native clients; for more information see Section 8.4.3 and Section 8.4.4.

**Note**

When connecting via PrivX Bastion you must verify the PrivX host keys (instead of the target server host keys). You can add the PrivX host keys to your known_hosts file by running (replace privx.example.com with your PrivX-server address):

```
ssh-keyscan -p 2222 privx.example.com >> ~/.ssh/known_hosts
```

### 8.4.3 Connecting with Native SSH Clients Using PrivX Agent (Unix and MacOS)

After PrivX agent is set up for your workstation account, you can establish connections to targets using native clients with the agent. To do this on Unix or MacOS:

1. Log into the workstation as the user for whom native clients have been set up.

   You may verify that the agent is running with:

   ```
   $ privx-agent-ctl status
   ```

   The command should return a message similar to the following:

<table>
<thead>
<tr>
<th>PrivX SSH Agent Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>PrivX Server          <a href="https://privx.example.com">https://privx.example.com</a></td>
</tr>
<tr>
<td>Login status          logged out</td>
</tr>
</tbody>
</table>

   If necessary, you can manually start the PrivX agent with:

   ```
   $ ./privx-agent-unix bash
   ```
2. Via the terminal, authenticate against PrivX using your PrivX credentials. For example (replace \textit{username} with your PrivX user name):

\begin{verbatim}
$ privx-agent-ctl login username
\end{verbatim}

You may verify your login status with:

\begin{verbatim}
$ privx-agent-ctl status
\end{verbatim}

After entering your PrivX credentials correctly, your native SSH clients (such as \textit{ssh}) will authenticate connections via PrivX. For a list of valid connection targets, run:

\begin{verbatim}
$ privx-agent-ctl target list
\end{verbatim}

Accessible targets and granting roles:

\begin{verbatim}
bilberry
  alice@10.1.55.144:222           Example Role 01
...
\end{verbatim}

You could then connect to one of the listed targets. In this example, by running:

\begin{verbatim}
$ ssh alice@10.1.55.144 -p 222
\end{verbatim}

\section*{8.4.4 Connecting with Native SSH Clients Using PrivX Agent (Windows)}

After PrivX agent is set up for your workstation account, you can establish connections to targets using native clients with the agent. To do this on Windows:

1. Use the PrivX agent to authenticate with PrivX. To do this, right click the PrivX-agent tray icon, then click \textit{Login}. Log in using your PrivX credentials. Complete multi-factor authentication if required.

\begin{itemize}
  \item \textbf{Role (optional)}: You may choose to log in with the permissions of a specific PrivX role. By default, you are logged in using any applicable role.
  \item \textbf{Target}: The target host
  \item \textbf{Client}: The native client used for connecting; PuTTY for connecting through SSH or PSFTP through SFTP.
\end{itemize}

After providing the connection settings, click \textit{Connect}. Alternatively, you can directly use your SSH client for connecting.
8.5 RDP Connections with Native Clients

This section describes how to establish RDP connections with native clients.

Users can connect to target hosts/accounts using the RDP clients installed on their workstations, without needing to use the PrivX GUI. Connections are authenticated against PrivX.

PrivX provides the following connection modes:

- **Interactive**: Access PrivX RDP Bastion to list and select possible targets.
- **Direct**: Specify your connection target directly to the native client.

**Note**

PrivX-Bastion connections are verified against the PrivX-Bastion host certificate. You may verify the certificate from the **Connections→Native Clients** page.

**Connecting Interactively**

To connect to targets with native clients interactively:

1. Use your native client to connect to a PrivX server.
2. Provide your PrivX credentials when prompted.
3. You are shown the targets where you are allowed access. Select a target to connect to it.

**Connecting Directly**

To directly connect to a target you know, provide the native client with the following parameters:

- **Host**: Address of a PrivX server.
- **User**: Credentials and target identification in the following format:

  `<target_username>%<extender_name>%<target_hostname>%<privx_username>`

  Where the `<extender_name>` is only required for target hosts behind Extenders.

**Note**

% characters in user names must be escaped with %%. For example, `%example%user%` becomes `%%example%%user%%`

Values may be separated using either % or | - The separator character can be escaped by doubling (%% or ||).

- **Password**: Your PrivX-user password.
Figure 8.1. Direct-connection example with Windows Remote Desktop Client

**Note**

When MFA is enabled, users must connect using the interactive method.

RDP certificate authentication is only supported through the PrivX GUI.

RDP with native clients via PrivX does not support file transfers via drive redirection when session recording is enabled. In such scenarios users may copy-paste to transfer files.

### 8.6 Website Access via PrivX

This section describes how to log into web services via PrivX.

You can use PrivX to connect to HTTP and HTTPS websites. Web connections established via PrivX offer the following benefits:

- Session-recording support for improved auditability.
- For sites that require login, you may store credentials in PrivX. PrivX automatically fills in the credentials, allowing users to log in without knowing any passwords. Access is provided in a role-based fashion.

To enable web connections your deployment must include at least one PrivX Carrier, and one PrivX Web Proxy. For system requirements and setup instructions, see Section 2.1.3 and Section 3.2.2 respectively.

After the required components are set up, add HTTP/HTTPS targets as described in Section 8.1.4.
8.7 AWS CLI Connection with Native Client

You can use PrivX to authenticate and authorize users of the AWS Command Line Interface (AWS CLI). For instructions about the required configurations, see Section 7.7.

8.8 Monitoring and Managing Connections

You can monitor ongoing and prior connections on the Monitor page. The page allows you to:

- See which user is currently connecting or has connected to which destination via SSH or RDP. Note that the page does not show agent-based connections.
- Filter connections by their status and search words.
- Terminate a connection.

8.9 Restricting Users' Access to Applications in RDP Connections

On RDP connections you can restrict which applications each user is able to access on the target Windows host. When adding or editing a host, the Applications entry allows you to specify restricted applications to each host user separately:

- **Application Name (Required):** is the name of the application visible to the user when connecting to said application. The name should be unique, and can be searched.
- **Identifier (Required):** the name of the application's executable, otherwise known as alias (for example, mspaint, wordpad)
- **Arguments:** command-line arguments to be passed to the application.
- **Directory:** the working directory for the application.

Note that you need to configure the target Windows host to accept and use the remote applications you specify. For instructions about configuring applications for RDP, see https://social.technet.microsoft.com/wiki/contents/articles/10817.publishing-remoteapps-in-windows-server-2012.aspx
Chapter 9 Auditing

This chapter describes how to use and configure PrivX auditing features.

9.1 Viewing Audit Data

In the PrivX GUI, you can find audit data from the following locations:

- For connection-specific audit events, go to Monitor→Connections and click one of the connection entries.
- For global audit events, see Monitor→Events.
- PrivX microservices generate logs to /var/log/messages. These may be useful in troubleshooting scenarios.

Note

To obtain video playback from connections, enable session recording as described in Section 9.3.

9.2 Using CEF Format for Audit Data

PrivX audit data can alternatively be formatted in the Common Event Format (CEF). Switching the audit logging to CEF may allow easier interoperability with some SIEM systems.

To switch the audit data format to CEF, edit the /opt/privx/etc/shared-config.toml file, setting the audit_event_format parameter to the following:

```toml
audit_event_format = "cef"
```

Restart PrivX to apply the changes:

```
# systemctl restart privx
```

9.2.1 Logging CEF Audit Messages to External SIEM

To send CEF log messages to an external SIEM, edit the /etc/rsyslog.conf file, and add the following:

```
# Send messages using rsyslog "forwarding output module".
```
On network error, try to reconnect 100 times to avoid lost messages.

```
local6.* action{type="omfwd" target="example.siem.net" port="1234" protocol="tcp"
    action.resumeRetryCount="100"
    queue.type="linkedList" queue.size="10000"
}
```

Drop the local6 messages from default messages.
```
local6.none /var/log/messages
```

Restart syslog:
```
# systemctl restart rsyslog
```

For more information on configuring rsyslog, see https://www.rsyslog.com/guides/

### 9.2.2 Logging CEF Audit Messages Internally

If you want to use the internal syslog service, add the following to the /etc/rsyslog.conf file:

```
# Create a template for CEF messages.
template(name="SSH_CefFormat" type="string"
    string="%TIMESTAMP% %HOSTNAME% %syslogtag%%msg:::drop-last-lf\n")
```

```
# Use the template for local6 (PrivX CEF logger uses that facility)
local6.* /var/log/ceflog;SSH_CefFormat
```

Drop the local6 messages from default messages.
```
*.info;mail,authpriv,cron,local6.none /var/log/messages
```

Restart syslog:
```
# systemctl restart rsyslog
```

### 9.3 Session Recording Setup

This section describes the procedures for setting up session recording.

When session recording is enabled, connection-specific audit events also provide:

- Video playback. With SSH sessions you can search for keyword occurrences.
- Transferred files.
- Clipboard (RDP only).
- Channel logs (SSH only).

To enable session recording for connections to a host:

1. On the Settings→Hosts page, Edit the host.

2. Under Options, enable the setting Session Recording. Click Save to apply your changes.

   Subsequent sessions to the host are recorded. You can view the playback and transferred files from the connection-specific audit events, available from Monitor→Connections.
Session recordings should not be stored on PrivX servers as they may consume lots of disk space; you should configure PrivX to store session recordings on an external share instead (such as NFS). To set up external storage share for PrivX session recordings:

1. On your external storage server, create a share for storing PrivX session recordings. The share must be a directory that satisfies the following:
   
   • The share must be mountable by all PrivX servers.
   • The share must be readable and writable by the `privx` system user of every PrivX server.

2. On each PrivX server, install any extensions required for mounting the external-storage share. For example, to mount NFS shares you will likely need to install `nfs-utils`; for SMB shares you will likely need `cifs-utils`. These extension packages are available from the RHEL/CentOS public repositories.

3. On each PrivX server, mount the external share to a local directory. To enable mounting the share on system startup, we recommend adding the mount directive to `/etc/fstab`.

   To allow the GUI to display other connection logs when the NFS server is unavailable, mount the share with options like the following:

   ```
   soft
timeo=10
retry=1
   ```

4. On each PrivX server, configure PrivX to store session recordings to the mounted share. To do this, edit the `data_folder` setting in `/opt/privx/etc/shared-config.toml` (replace `/path/to/privx-trails` with the local directory to which the external share is mounted):

   ```
   data_folder="/path/to/privx-trails"
   ```

   Save your changes, then restart PrivX services to apply the changes:

   ```
   # systemctl restart privx
   ```

   By default PrivX retains session recordings indefinitely. You may configure PrivX to automatically delete old recordings, using the following settings:

   • `trail_expiry` in `/opt/privx/etc/shared-config.toml` - Delete recordings older than the specified number of days. Set to `-1` to disable automatic deletion.

   • `housekeeping_interval_for_trails` in `/opt/privx/etc/connectionmanager.toml` - The interval at which PrivX checks for and deletes expired recordings. Specified in hours.

   To apply new configurations, restart the PrivX services:

   ```
   # systemctl restart privx
   ```

   **Note**

   PrivX generates keyframe data when opening RDP session recordings for the first time. Note that this may take up to several minutes for large RDP and web-connection trails.
PrivX indexes session recordings when they are searched for the first time. Depending on the duration of the recording, the first search may take some time. SSH transcripts require roughly ten times the storage space compared to the original video recording.

9.4 Log-Collector Setup

You can configure PrivX to forward audit events to external log collectors. The required configurations are provided separately per supported log collector.

AWS CloudWatch

1. Create an AWS user with permissions for pushing logs to CloudWatch. The user's permissions should be similar to the following (Sid is arbitrary):

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "examplesid",
            "Effect": "Allow",
            "Action": "events:PutEvents",
            "Resource": "*"
        }
    ]
}
```

Obtain the Access key ID and the Secret access key of this AWS user, required later for configuring PrivX.

2. Create a rule in CloudWatch for collecting PrivX logs. The rule must have the following Event pattern:

```json
{
    "source": [  
        "com.ssh.privx"
    ]
}
```

To associate actions to collected logs, also create Targets for the rule.

3. Add your log collector to PrivX.

In the PrivX GUI, navigate to the page **Settings→Deployment→Configure cloud log collectors**, then click **Add Log Collector**. Set Service to *Amazon CloudWatch Events*, and provide the other required settings as well. Click **Save** to apply your settings.

PrivX logs are now sent to your AWS CloudWatch and processed according to all the applicable rule targets.

Azure Event Hubs

1. In Azure Active Directory, register PrivX as an application.
2. In Azure Event Hubs, create an event hub for PrivX. Your Access Control must allow the previously-registered PrivX application to access this hub.

3. To process incoming events, you may create consumers for the hub.

4. Add your log collector to PrivX.
   In the PrivX GUI, navigate to the page **Settings→Deployment→Configure cloud log collectors**, then click **Add Log Collector**. Set **Service** to **Azure Event Hubs**, and provide the other required settings as well. Click **Save** to apply your settings.

   PrivX logs are now sent to the hub and processed according to all the applicable rules.

### 9.5 Matching Certificate-Based-Login Messages

PrivX users logging in with certificate-based authentication generate log messages both on the PrivX server and on the target host. These messages can be matched by the certificate serial.

For example, an SSH connection to a Unix host may generate the following messages:

- **On the PrivX server (in /var/log/messages):**
  ```
  Dec 5 15:57:27 privx.example.com SSH-PRIVX-AUDIT[6825]:
  [event="Authorization-certificate-granted" eventID="401"
  keyID="alice@127.0.0.1:45910-serial-2571351803943628705"
  message="certificate-created" target="127.0.0.1:45910"
  username="alice"]
  ```

- **And on the target host (typically in /var/log/secure or /var/log/auth.log):**
  ```
  Dec 5 15:57:28 ld-jizhouya sshd[22799]: Accepted publickey for alice
  from 192.0.2.102 port 38126 ssh2: RSA-CERT ID alice@127.0.0.1:45910
  serial 2571351803943628705 (serial 2571351803943628705)
  CA RSA SHA256:aVOPjQAB2b+y64OJ8UozVe5EksgsCC1E9UQN/MEq4c
  ```

As another example, an RDP connection to a Windows host may generate:

- **On the PrivX server (in /var/log/messages):**
  ```
  Dec 5 08:24:41 dhcp-10-1-54-160.hel.fi.ssh.com SSH-PRIVX-AUDIT[14189]:
  [event="Authorization-certificate-granted" eventID="401"
  SSH-PrivX-service="AUTHORIZER" message="RDP-certificate-created"
  serial="1A654E1CD607153C"
  sha1-fingerprint="..." sha256-fingerprint="...
  target="127.0.0.1:47390" upn="alice@example.com" username="alice"]
  ```

- **And on the target host (in Windows Event Viewer→Windows Logs→Security→Event details):**
  ```
  Audit Success 5.12.2018 15.25.09 Microsoft-Windows-Security-Auditing
  4768 Kerberos Authentication Service "A Kerberos authentication ticket (TGT) was requested.
  ```

  Account Information:
  Account Name: alice
9.6 Audit Events in PrivX

This section lists the audit events raised by the PrivX system. Note that some of the listed audit-event types are not used by the current PrivX version.

PrivX uses syslog facilities to write audit events. Audit events are stored to the system's default syslog location. On RHEL/CentOS 7 the default location is `/var/log/messages`.

The debug logs for each microservice are in the following locations:

- `/var/log/privx/auth.log` - Logs for the OAuth2 microservice.
- `/var/log/privx/authorizer.log` - Logs for the authorizer microservice.
- `/var/log/privx/connectionmanager.log` - Logs for the connection manager microservice.
- `/var/log/privx/hoststore.log` - Logs for the hoststore microservice.
- `/var/log/privx/indexerservice.log` - Logs for the indexer microservice.
- `/var/log/privx/keyvault.log` - Logs for the keyvault microservice.
- `/var/log/privx/monitorservice.log` - Logs for the monitor microservice.
- `/var/log/privx/rdpmitm.log` - Logs for the RDP-Bastion microservice.
- `/var/log/privx/rdpproxy.log` - Logs for the RDP-proxy microservice.
- `/var/log/privx/redemption.log` - Logs for the RDP-trail-encoder microservice.
- `/var/log/privx/rolestore.log` - Logs for the role-store microservice.
- `/var/log/privx/sshmitm.log` - Logs for the SSH-Bastion microservice.
• /var/log/privx/sshproxy.log - Logs for the SSH-proxy microservice.
• /var/log/privx/trail-indexer.log - Logs for the trail-indexer microservice.
• /var/log/privx/userstore.log - Logs for the user-store microservice.
• /var/log/privx/watchdog.log - Logs for the watchdog microservice.
• /var/log/privx/workflowengine.log - Logs for the workflow-engine microservice.

<table>
<thead>
<tr>
<th>Event Name</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>License-error</td>
<td>0</td>
<td>The system license does not allow operation.</td>
</tr>
<tr>
<td>Configuration-error</td>
<td>1</td>
<td>The system configuration is invalid.</td>
</tr>
<tr>
<td>Service-starting</td>
<td>10</td>
<td>The service is starting.</td>
</tr>
<tr>
<td>Service-running</td>
<td>11</td>
<td>The service is running.</td>
</tr>
<tr>
<td>Service-stopped</td>
<td>12</td>
<td>The service has been stopped.</td>
</tr>
<tr>
<td>Housekeeping-started</td>
<td>20</td>
<td>The service is started housekeeping.</td>
</tr>
<tr>
<td>Housekeeping-running</td>
<td>21</td>
<td>Housekeeping is running.</td>
</tr>
<tr>
<td>Housekeeping-completed</td>
<td>22</td>
<td>Housekeeping is completed.</td>
</tr>
<tr>
<td>User-logged-in</td>
<td>100</td>
<td>User has logged in to the system.</td>
</tr>
<tr>
<td>User-login-failed</td>
<td>102</td>
<td>User login operation failed.</td>
</tr>
<tr>
<td>User-MFA-challenge-sent</td>
<td>103</td>
<td>User tried to log in without MFA pin code.</td>
</tr>
<tr>
<td>User-MFA-challenge-accepted</td>
<td>104</td>
<td>User successfully authenticated with MFA pin code.</td>
</tr>
<tr>
<td>User-MFA-challenge-setup-sent</td>
<td>105</td>
<td>User was MFA setup information.</td>
</tr>
<tr>
<td>Access-token-granted</td>
<td>106</td>
<td>Access token granted</td>
</tr>
<tr>
<td>User-access-token-refreshed</td>
<td>110</td>
<td>User refreshed the access token</td>
</tr>
<tr>
<td>User-access-token-refresh-failed</td>
<td>111</td>
<td>User access token refresh failed.</td>
</tr>
<tr>
<td>OAuth-client-authenticated</td>
<td>121</td>
<td>OAuth client authenticated</td>
</tr>
<tr>
<td>User-login-attempt-rate-limited</td>
<td>130</td>
<td>User login attempt rate limited.</td>
</tr>
<tr>
<td>Role-added</td>
<td>201</td>
<td>New role added to the system.</td>
</tr>
<tr>
<td>Role-modified</td>
<td>202</td>
<td>Role has been modified.</td>
</tr>
<tr>
<td>Role-removed</td>
<td>203</td>
<td>Role has been removed.</td>
</tr>
<tr>
<td>Directory-added</td>
<td>210</td>
<td>New directory added to the system.</td>
</tr>
<tr>
<td>Directory-modified</td>
<td>211</td>
<td>Directory has been modified.</td>
</tr>
<tr>
<td>Directory-removed</td>
<td>212</td>
<td>Directory has been removed.</td>
</tr>
<tr>
<td>User-roles-modified</td>
<td>220</td>
<td>The user's role associations were changed</td>
</tr>
<tr>
<td>AWS-token-granted</td>
<td>230</td>
<td>AWS token was granted to a user</td>
</tr>
<tr>
<td>AWS-token-grant-failed</td>
<td>231</td>
<td>AWS token grant failed</td>
</tr>
<tr>
<td>LogConf-collector-created</td>
<td>232</td>
<td>LogConf collector created</td>
</tr>
<tr>
<td>LogConf-collector-modified</td>
<td>233</td>
<td>LogConf collector modified</td>
</tr>
<tr>
<td>LogConf-collector-removed</td>
<td>234</td>
<td>LogConf collector removed</td>
</tr>
<tr>
<td>Event Name</td>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>LogConf-collector-failed</td>
<td>235</td>
<td>LogConf collector failed</td>
</tr>
<tr>
<td>Connection-requested</td>
<td>300</td>
<td>Connection was requested.</td>
</tr>
<tr>
<td>Connection-authenticated</td>
<td>301</td>
<td>Connection was authenticated.</td>
</tr>
<tr>
<td>Connection-rejected</td>
<td>302</td>
<td>Connection was rejected.</td>
</tr>
<tr>
<td>Connection-closed</td>
<td>303</td>
<td>Connection was closed.</td>
</tr>
<tr>
<td>Connection-failed</td>
<td>304</td>
<td>Connection was closed with an error.</td>
</tr>
<tr>
<td>Client-authenticated</td>
<td>305</td>
<td>Client was authenticated.</td>
</tr>
<tr>
<td>Session-added</td>
<td>310</td>
<td>A session was added to a connection.</td>
</tr>
<tr>
<td>Session-removed</td>
<td>311</td>
<td>A session was removed from a connection.</td>
</tr>
<tr>
<td>Session-rejected</td>
<td>312</td>
<td>A session was rejected.</td>
</tr>
<tr>
<td>File-upload</td>
<td>320</td>
<td>File upload performed.</td>
</tr>
<tr>
<td>File-download</td>
<td>321</td>
<td>File download performed.</td>
</tr>
<tr>
<td>Host-key-matched</td>
<td>324</td>
<td>Host key matched</td>
</tr>
<tr>
<td>Host-key-denied</td>
<td>325</td>
<td>Host key denied</td>
</tr>
<tr>
<td>Host-key-accepted</td>
<td>326</td>
<td>Host key accepted</td>
</tr>
<tr>
<td>Host-key-saved</td>
<td>327</td>
<td>Host key saved</td>
</tr>
<tr>
<td>Extender-connected</td>
<td>328</td>
<td>Extender connected</td>
</tr>
<tr>
<td>Extender-disconnected</td>
<td>329</td>
<td>Extender disconnected</td>
</tr>
<tr>
<td>File-removed</td>
<td>330</td>
<td>File removed via SSH.</td>
</tr>
<tr>
<td>Folder-removed</td>
<td>331</td>
<td>Folder removed via SSH.</td>
</tr>
<tr>
<td>File-moved</td>
<td>332</td>
<td>File moved.</td>
</tr>
<tr>
<td>Folder-created</td>
<td>333</td>
<td>Folder created.</td>
</tr>
<tr>
<td>Connection-audit-started</td>
<td>334</td>
<td>Connection audit started.</td>
</tr>
<tr>
<td>Connection-audit-failed</td>
<td>335</td>
<td>Connection audit failed.</td>
</tr>
<tr>
<td>Authorization-requested</td>
<td>400</td>
<td>A client requested an authorization.</td>
</tr>
<tr>
<td>Authorization-certificate-granted</td>
<td>401</td>
<td>An authorization certificate granted.</td>
</tr>
<tr>
<td>Authorization-role-key-granted</td>
<td>402</td>
<td>An authorization role key granted.</td>
</tr>
</tbody>
</table>
| Authorization-role-key-sign-
  operation-rejected            | 403  | An authorization role key sign operation rejected.|
| Authorization-role-key-sign-
  operation-accepted           | 404  | An authorization role key sign operation accepted.|
<p>| Authorization-rejected           | 405  | An authorization was rejected.                   |
| Authorization-Passphrase-returned| 407  | Authorization passphrase was returned            |
| Principal-added                  | 410  | A principal was added.                           |
| Principal-removed                | 411  | A principal was removed.                         |
| Trusted-client-added             | 420  | A trusted client was added.                      |
| Trusted-client-modified          | 421  | A trusted client was modified.                   |
| Trusted-client-removed           | 423  | A trusted client was removed.                    |
| License-updated                  | 430  | The service license was updated.                 |</p>
<table>
<thead>
<tr>
<th>Event Name</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA-Certificate-Created</td>
<td>440</td>
<td>CA certificate was created.</td>
</tr>
<tr>
<td>CA-Certificate-Deleted</td>
<td>441</td>
<td>CA certificate was deleted.</td>
</tr>
<tr>
<td>EE-Certificate-Enrolled</td>
<td>442</td>
<td>End entity certificate was enrolled</td>
</tr>
<tr>
<td>EE-Certificate-Revoked</td>
<td>443</td>
<td>End entity certificate was revoked</td>
</tr>
<tr>
<td>CA-Certificate-Enrolled</td>
<td>444</td>
<td>CA certificate was enrolled</td>
</tr>
<tr>
<td>CA-Certificate-Revoked</td>
<td>445</td>
<td>CA certificate was revoked</td>
</tr>
<tr>
<td>User-added</td>
<td>500</td>
<td>New user added to the system.</td>
</tr>
<tr>
<td>User-modified</td>
<td>501</td>
<td>User has been modified.</td>
</tr>
<tr>
<td>User-removed</td>
<td>502</td>
<td>User has been removed.</td>
</tr>
<tr>
<td>User-password-modified</td>
<td>510</td>
<td>User password has been modified.</td>
</tr>
<tr>
<td>User-authenticated</td>
<td>520</td>
<td>User has been authenticated.</td>
</tr>
<tr>
<td>User-authentication-failed</td>
<td>521</td>
<td>User authentication has failed.</td>
</tr>
<tr>
<td>Workflow-added</td>
<td>600</td>
<td>A workflow was added.</td>
</tr>
<tr>
<td>Workflow-modified</td>
<td>601</td>
<td>A workflow was modified.</td>
</tr>
<tr>
<td>Workflow-removed</td>
<td>602</td>
<td>A workflow was removed.</td>
</tr>
<tr>
<td>Request-added</td>
<td>610</td>
<td>A request was added.</td>
</tr>
<tr>
<td>Request-removed</td>
<td>612</td>
<td>A request was removed.</td>
</tr>
<tr>
<td>Decision-made</td>
<td>620</td>
<td>A decision has been made on a request.</td>
</tr>
<tr>
<td>Email-sent</td>
<td>630</td>
<td>A email notification has been sent.</td>
</tr>
<tr>
<td>Email-configuration-Modified</td>
<td>631</td>
<td>Email configuration has been modified.</td>
</tr>
<tr>
<td>Email-not-sent</td>
<td>632</td>
<td>Email not sent.</td>
</tr>
<tr>
<td>Log-downloaded</td>
<td>700</td>
<td>Log files have been downloaded.</td>
</tr>
<tr>
<td>Log-level-modified</td>
<td>710</td>
<td>The log level was modified.</td>
</tr>
<tr>
<td>Host-added</td>
<td>801</td>
<td>A host was added.</td>
</tr>
<tr>
<td>Host-modified</td>
<td>802</td>
<td>A host was modified.</td>
</tr>
<tr>
<td>Host-removed</td>
<td>803</td>
<td>A host was removed.</td>
</tr>
<tr>
<td>Host-service-connection-re-established</td>
<td>804</td>
<td>A host service connection re-established.</td>
</tr>
<tr>
<td>Host-service-connection-failure</td>
<td>805</td>
<td>A host service connection failed.</td>
</tr>
<tr>
<td>Connection-terminated</td>
<td>900</td>
<td>Connection terminated.</td>
</tr>
<tr>
<td>Connection-terminated-for-host</td>
<td>901</td>
<td>Connection terminated for host.</td>
</tr>
<tr>
<td>Connection-terminated-for-user</td>
<td>902</td>
<td>Connection terminated for user.</td>
</tr>
<tr>
<td>Licensed-connection-count-exceeded</td>
<td>903</td>
<td>Licensed connection count exceeded.</td>
</tr>
<tr>
<td>Trail-opened</td>
<td>1000</td>
<td>Trail opened.</td>
</tr>
<tr>
<td>Trail-open-failed</td>
<td>1001</td>
<td>Failed to open trail.</td>
</tr>
<tr>
<td>Trail-file-open-failed</td>
<td>1002</td>
<td>Failed to open trail file.</td>
</tr>
<tr>
<td>Trail-file-read-failed</td>
<td>1003</td>
<td>Failed to read trail file.</td>
</tr>
<tr>
<td>Trail-removed</td>
<td>1004</td>
<td>Trail removed.</td>
</tr>
<tr>
<td>Trail-remove-failed</td>
<td>1005</td>
<td>Failed to remove trail.</td>
</tr>
<tr>
<td>Event Name</td>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>Trail-file-integrity-failed</td>
<td>1006</td>
<td>Trail file integrity check failed.</td>
</tr>
<tr>
<td>Config-checksum-added</td>
<td>1100</td>
<td>A config file checksum was added.</td>
</tr>
<tr>
<td>Config-checksum-changed</td>
<td>1101</td>
<td>A config file checksum has changed.</td>
</tr>
<tr>
<td>Transcript-status-scheduled</td>
<td>1201</td>
<td>Transcript status</td>
</tr>
<tr>
<td>Transcript-status-indexing</td>
<td>1202</td>
<td>Transcript status</td>
</tr>
<tr>
<td>Transcript-status-indexed</td>
<td>1203</td>
<td>Transcript status</td>
</tr>
<tr>
<td>Transcript-status-error</td>
<td>1204</td>
<td>Transcript status</td>
</tr>
<tr>
<td>Transcript-status-not-indexed</td>
<td>1205</td>
<td>Transcript status</td>
</tr>
<tr>
<td>Transcript-trail-removed</td>
<td>1206</td>
<td>Transcript trail removed.</td>
</tr>
<tr>
<td>Transcript-opened</td>
<td>1207</td>
<td>Transcript opened.</td>
</tr>
<tr>
<td>Disk-Full</td>
<td>1301</td>
<td>Disk Full.</td>
</tr>
</tbody>
</table>

Table 9.1. List of Audit Events
Chapter 10 Troubleshooting

This section describes the usual solutions to fixing issues related to, for example, setup, initial login, and services being unavailable. Also restarting PrivX and accessing debug logs are instructed.

Reading Debug Logs

Messages in syslog /var/log/messages and /var/log/privx/*.log may provide additional information about issues in the setup and in client connections.

- /var/log/privx/auth.log - Logs for the OAuth2 microservice.
- /var/log/privx/authorizer.log - Logs for the authorizer microservice.
- /var/log/privx/connectionmanager.log - Logs for the connection manager microservice.
- /var/log/privx/hoststore.log - Logs for the hoststore microservice.
- /var/log/privx/indexerservice.log - Logs for the indexer microservice.
- /var/log/privx/keyvault.log - Logs for the keyvault microservice.
- /var/log/privx/monitor.service.log - Logs for the monitor microservice.
- /var/log/privx/rdpmitm.log - Logs for the RDP-Bastion microservice.
- /var/log/privx/rdpproxy.log - Logs for the RDP-proxy microservice.
- /var/log/privx/redemption.log - Logs for the RDP-trail-encoder microservice.
- /var/log/privx/rolestore.log - Logs for the role-store microservice.
- /var/log/privx/sshmitm.log - Logs for the SSH-Bastion microservice.
- /var/log/privx/sshproxy.log - Logs for the SSH-proxy microservice.
- /var/log/privx/trail-indexer.log - Logs for the trail-indexer microservice.
- /var/log/privx/userstore.log - Logs for the user-store microservice.
• /var/log/privx/watchdog.log - Logs for the watchdog microservice.
• /var/log/privx/workflowengine.log - Logs for the workflow-engine microservice.

For more details about what each microservice does, see Appendix B.

For more information about enabling debug logging, see Section 5.2.

**Checking microservice statuses**

You can verify the microservice statuses on the status page, available at (replace `https://privx.example.com` with the address of your PrivX server):

`https://privx.example.com/status.html`

**Restarting PrivX**

Some issues may be fixed by restarting the PrivX service. You can do this by running the following command on your PrivX servers:

```
# systemctl restart privx
```

**Services Are Down**

If microservices report 502 errors, check the database connectivity. Make sure the `pg_hba.conf` on the PostgreSQL server allows PrivX servers to connect to the PrivX database with `md5` authentication.

More information about the error may be provided by the microservice logs.

**Connections fail with error Too many Authentication Failures**

PrivX connects using all the available credentials (such as certificate, public key, and stored credentials). For target users who allow many roles, the OpenSSH `MaxAuthTries` may be exceeded before PrivX offers the correct authentication method.

To fix the error, increase the `MaxAuthTries` in the target-host OpenSSH configuration, restart the target-host OpenSSH server, then try connecting again.

**Problems Related to Authentication**

More information about authentication issues may be found from syslog messages.
Since PrivX uses short-term certificates for user authentication, deviations in system times (minutes or more) will easily result in failing authentications. It is extremely important to synchronize the clocks between all the involved systems:

- PrivX servers
- Hosts serving as connection endpoints.

We strongly recommend using authenticated NTP clock synchronization on all the involved systems.

Directory Users Are Not Listed

PrivX applies a default pre-filter for matching user records. The pre-filter may need to be adjusted depending on your user-directory structure. For more information about adjusting how users are matched, see Section 6.5.2.

List Users View Does Not Display All Attributes

Sometimes in List users view some of the columns show Anonymous, or are empty. In this case it is possible you may need to adjust the attribute mapping in your directory settings. For example in default OpenLDAP setup you could have the following attribute mappings:

- sn=full_name
- uid=principal

Contacting Support

You can request technical support and let us know about PrivX issues on the SSH Communications Security support site at https://help.ssh.com/

When submitting a support ticket, please attach PrivX troubleshooting data. To generate troubleshooting data, run the following on your PrivX server:

```
# /opt/privx/scripts/troubleshoot.sh info
```

And attach the generated troubleshooting-data package to the support ticket. By default the package is generated to the current working directory, with the name:

```
Privx-troubleshoot-data-<host>-<timestamp>.tar.gz
```

Caution

Troubleshooting data may contain security critical data, and must be distributed with appropriate care.
For additional usage instructions related to the troubleshooting script, you may run:

```bash
# /opt/privx/scripts/troubleshoot.sh -h
```
Appendix A Example Nginx Load-Balancer Configuration

You can use a Nginx web server as a load balancer in PrivX-HA deployments. This chapter provides some example configurations for an Nginx load balancer:

- Nginx configuration with TLS passthrough.
- Nginx configuration with TLS termination.

The Nginx used for the load balancer must be built with the `ngx_stream_core_module`. On Red Hat and CentOS, you may install the `nginx-mod-stream` package (available in the EPEL repository). If you are building Nginx yourself, you must configure Nginx with the stream module, similarly to the following:

```
# ./configure --with-stream
```

The sample configurations require that the SELinux policy allows Nginx process to listen at ports 2222 and 3389. Run the following commands to add these ports to the list of allowed listen ports for Nginx:

```
semanage port -a -t http_port_t -p tcp 2222
semanage port -a -t http_port_t -p tcp 3389
```

In the sample configurations, the IP addresses 192.0.2.100 and 192.0.2.101 need to be substituted with your PrivX-server addresses.

To disconnect a PrivX server, append `down` to the server address. For example:

```
upstream privx_http {
    hash $remote_addr consistent;
    server 192.0.2.100:443;
    server 192.0.2.101:443 down;
}
```
Then reload the Nginx configuration with:

```
# systemctl reload nginx
```

To reconnect a PrivX server, remove the `down` keyword and reload the configuration.

### A.1 Configuration with TLS Passthrough

```
# NGINX Load-balancer example with TLS passthrough.
# Requests are distributed between servers based on client IP addresses.
# The upstream module contains the IP addresses of the PrivX servers.
# In this example two instances of PrivX are running
# with private IP addresses 192.0.2.100 and 192.0.2.101
# HTTP requests except for CRLs are redirected to HTTPS.

load_module "/usr/lib64/nginx/modules/ngx_stream_module.so";

events {
    worker_connections 1024;
}

stream {
    upstream privx_http {
        hash $remote_addr consistent;
        server 192.0.2.100:80;
        server 192.0.2.101:80;
    }

    upstream privx_https {
        hash $remote_addr consistent;
        server 192.0.2.100:443;
        server 192.0.2.101:443;
    }

    upstream privx_ssh-mitm {
        hash $remote_addr consistent;
        server 192.0.2.100:2222;
        server 192.0.2.101:2222;
    }

    upstream privx_rdp-mitm {
        hash $remote_addr consistent;
        server 192.0.2.100:3389;
        server 192.0.2.101:3389;
    }

    server {
        listen 80;
        proxy_pass privx_http;
        proxy_connect_timeout 1s;
    }

    server {
```
A.2 Configuration with TLS Termination

# NGINX Load-balancer example with TLS termination.
# Requests are distributed between servers based on client IP addresses.
# The upstream module contains the IP addresses of the PrivX servers.
# In this example two instances of PrivX are running with private IP addresses 192.0.2.100 and 192.0.2.101
# HTTP requests except for CRLs are redirected to HTTPS.

load_module "/usr/lib64/nginx/modules/ngx_stream_module.so";

events {
    worker_connections 1024;
}

http {
    upstream privx_http {
        ip_hash;
        server 192.0.2.100:80;
        server 192.0.2.101:80;
    }

    upstream privx_https {
        ip_hash;
        server 192.0.2.100:443;
        server 192.0.2.101:443;
    }

    client_max_body_size 50M;

    server {
        listen 80 default_server;
        listen [::]:80 default_server;
        location / {

            listen 443;
            proxy_pass privx_https;
            proxy_connect_timeout 1s;
        }

        server {
            listen 2222;
            proxy_pass privx_ssh-mitm;
            proxy_connect_timeout 1s;
        }

        server {
            listen 3389;
            proxy_pass privx_rdp-mitm;
            proxy_connect_timeout 1s;
        }
    }
}
Appendix A Example Nginx Load-Balancer Configuration

```nginx
proxy_set_header X-Forwarded-Host $http_host;
proxy_pass http://privx_http;
}
}

server {
listen 443 ssl http2 default_server;
listen [:]:443 ssl http2 default_server;
ssl_certificate "/etc/nginx/server.crt";
ssl_certificate_key "/etc/nginx/server.key";

location / {
    proxy_set_header X-Forwarded-Host $http_host;
    proxy_pass https://privx_https;
}

location ~ ^/(ssh-proxy|ssh-mitm|rdp-proxy|rdp-mitm)/ {
    proxy_pass https://privx_https;
    proxy_set_header Host $http_host;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "Upgrade";
}
}

stream {
    upstream privx_ssh-mitm {
        hash $remote_addr consistent;
        server 192.0.2.100:2222;
        server 192.0.2.101:2222;
    }

    upstream privx_rdp-mitm {
        hash $remote_addr consistent;
        server 192.0.2.100:3389;
        server 192.0.2.101:3389;
    }

    server {
        listen 2222;
        proxy_pass privx_ssh-mitm;
        proxy_connect_timeout 1s;
    }

    server {
        listen 3389;
        proxy_pass privx_rdp-mitm;
        proxy_connect_timeout 1s;
    }
}
```

© 2015—2020 SSH Communications Security
Corporation
PrivX™ 12.1 Administrator Manual
Appendix B PrivX Architecture

Figure B.1. PrivX general architecture
See the following sections for more information on the PrivX microservices.

1. **Nginx**

   Nginx is an open source HTTP server and reverse proxy. It routes traffic from the public facing 443 and 80 ports to individual PrivX microservices. It also routes internal microservice TLS traffic. It also serves static files. All traffic is HTTPS apart from Windows Certificate Revocation List requests which in accordance with Microsoft implementation must be over HTTP port 80. All external and internal communication is through REST APIs.

2. **Static Files**

   Nginx serves the static files from /www/data -directory on the PrivX host. The files consist of the HTML5 javascript UI and accompanying HTML, CSS and image files. The UI will function on top of PrivX through REST APIs.

3. **Authentication**

   The Authentication microservice authenticates clients and users to PrivX system via OAuth2. It will authenticate the requesting client either against the local user store or a specified remote user source such as an Active Directory, LDAP or Open ID Connect provider.

4. **Role Store**

   The Role Store integrates to both user providers (sources) (B) and to host directories (targets) (C & D). It essentially fetches users from remote sources (B) and maps the users to PrivX roles according to pre-defined rules or explicit role grants. Other microservices will query Role Store for user's roles on on-demand basis. The role store will also fetch target hosts from cloud providers (C & D) and push them to host store (9). Role Store keeps discovered users and mapped roles in-memory for fast access and resolving.

5. **User Store**

   The User Store is a PrivX-specific user directory. By default, the initial administrator account is created into the user directory. The User Store allows PrivX to act as a standalone solution and it also allows the creation of temporary users which necessarily need not be created in a user directory upstream from PrivX.

6. **Workflow Engine**

   The Workflow Engine allows users to request roles and for the approvers to grant or deny role requests. It will also push out email notifications to approvers.

7. **Monitor Service**

   The Monitor Service collects health data and audit events from PrivX microservices and associated extenders.

8. **Keyvault**
The Keyvault microservice stores all sensitive data within PrivX. The sensitive data includes stored credentials, private keys and so forth.

9. **Host Store**

The Host Store stores all locally added hosts as well as hosts discovered from cloud providers (C & D). It also keeps track of service, account and role mappings and the proxy microservices will check the host configuration from the host store before allowing access to a target host.

10. **Authorizer**

Authorizer creates the ephemeral certificates needed to access SSH and RDP target hosts via certificate authentication. It will also provide public keys and/or stored credentials from the keyvault to proxies for authentication.

11. **Connection Manager**

The Connection Manager keeps track of all ongoing proxied connections to target hosts. The administrator can also terminate connections, or if the connection was audited, playback the audit trail from the trail storage (14).

12. **SSH Proxy**

SSH Proxy establishes SSH connections to target hosts. It will verify user's roles and check that the roles match the configuration found for the host from the host store. The PrivX UI will establish a websocket connection to the SSH Proxy for the VT100 terminal emulation. If the connection is audited, the SSH Proxy will create a trail file to trail storage (14) and also play it back on Connection Manager's (11) request. The proxy can connect to the target host directly or via the PrivX extender (18).

13. **RDP Proxy**

RDP proxy establishes RDP connections to target hosts. It will verify user's roles and check that the roles match the configuration found for the host from the host store. The PrivX UI will establish a websocket connection to the RDP Proxy desktop session. If the connection is audited, the RDP Proxy will create a trail file to trail storage (14) and also play it back on Connection Manager's (11) request. The proxy can connect to the target host directly or via the PrivX extender (18).

14. **Audit Trail Storage**

The audit trail storage is a directory on the PrivX host. Ideally, this directory would be mounted to a secure NAS/SAN solution.

15. **rsyslog**

The PrivX microservices write audit-, debug- and systemlogs to local syslog file. Rsyslog (or equivalent) should be configured to transfer the log files to an external SIEM solution (A).

16. **Postgres**
PrivX uses PostgreSQL as the storage mechanism for configuration data such as role mappings as well as manually added users or hosts and discovered cloud hosts.

17. **Redis**

Redis is used as a notification mechanism between the microservices. No sensitive data is put in it, only timestamps.

18. **PrivX Extender**

PrivX Extender enables PrivX to reach firewalled private networks or virtual private clouds. Once deployed in the private network, it will establish a number of websocket connections to PrivX to route traffic from PrivX proxies to the target network.

19. **PrivX Carrier**

PrivX Carrier provides a virtualized runtime environment via Docker for the Firefox containers used for Web Access.

20. **PrivX Web Proxy**

PrivX Web Proxy intercepts the traffic from the Firefox containers to the target hosts and provides secrets on the fly.
Appendix C PrivX Analytics

The PrivX analytics feature is used to gather data about the operating system, CPU, memory, device name, geographical location, and the version of PrivX running on your installation. After first use, PrivX servers track these parameters and periodically send the data to SSH Communications Security Corporation.

Upon license activation, PrivX servers uniquely identify the node on which the installation exists. This interaction is not related to any individual user and hence completely anonymous. This data is used to understand usage patterns, and to improve the product to better serve our customers.

Analytics data is sent on every license refresh operation, which is scheduled to run automatically every 12 hours. Please note that the data is also sent if you manually refresh or update the license information, which is done by clicking Update on the Settings→License page.

Analytics are enabled by default upon license activation. You can opt out of sending analytics at any time by going to the Settings→License page and setting Send analytics data to No.
Index

Symbols
2FA, 74

A
accessing target hosts, 65
Accounts (host configuration), 65
Active Directory, 63
Add Hosts, 97
admin rights, 66
agent, 105
all attributes are not listed
  troubleshooting, 125
Allow PrivX agent (SSH option), 106
approvals, 64
audit events (list), 118
audit trails, 114
audit trail storage, 133
authentication, 71
  troubleshooting, 124
authentication service, 132
authorizer, 9, 133

C
carrier, 134
certificate, 9
certificate authentication, 75
  manual setup, 84
  script-based setup, 79
certificate authentication for RDP, 90
certificate authentication for SSH, 78
configuring target hosts, 77
connection manager, 112, 133
connections
  current, 112
  managing, 112
  monitoring, 112
  terminating, 112

d debug logs
  troubleshooting, 123
deployment, 77

rotate CA, 56
SSH target host configuration, 84
directory (in UI), 9
documentation conventions, 10

E
e-mail-notification setup, 66
encryption keys, 18, 56
extender, 134

H
host cannot be redeployed (error), 98
host service, 9
host store, 9, 133
HSM integration, 18
HTTP/HTTPS connections, 24, 100, 111

I
Import host instance tags from the directory
  (directory setting), 79
installation, 21

K
kerberos authentication for PrivX users, 72
keyvault, 132
known host, 9

L
LDAP, 63
ldap, 64
LDAPS setup, 68
license, 21
license activation
  online, 32
license deactivation, 33
license features, 16
license management, 32
licensing, 21
local user directory, 9
local users, 43, 63
local user store, 132
loginratelimmit, 67
login with personal account, 85
  script-based setup, 79, 85
log locations, 118
log settings, 56

M
managing connections, 112
mfa, 74
microservices, 131
monitoring connections, 112
monitor service, 132
multi-factor authentication, 74
my requests, 64

N
native client, 9, 106

O
OAuth2 service, 9
online licensing, 32

P
permissions, 63
target-host access, 65
postgres, 133
preparing for installation, 13
prerequisites, 13
principal, 9, 84
privx-admin (role), 65
PrivX Carrier
requirements, 14
setup, 24
upgrade, 39
PrivX deployment, 9
PrivX Extender
host configuration, 79, 98
requirements, 14
setup, 23
upgrade, 39
PrivX user, 10
PrivX Web Proxy
requirements, 14
setup, 24
upgrade, 39
public-key authentication, 89

R
RDP proxy, 133
redis, 134
refresh_token_valid, 66
refresh (directory action), 70
refresh interval, 70
requests, 64, 66
e-mail notifications, 66
require password change, 67
reset MFA pairing, 75
restarting PrivX
troubleshooting, 124
role, 10
roles, 64
role store, 10, 132
rotating encryption keys, 56
rsyslog, 133

S
securing directory connections, 68
server setup, 21
server upgrade, 37
services are down
troubleshooting, 124
Settings
Hosts, 97
set up server, 21
smart-card authentication, 75
SSH host keys, 92
SSH options, 106
SSH proxy, 133
SSH Target Host Configuration
principal, 88
public key, 89
STARTTLS setup, 68
stored password, 90
system upgrade, 37
system workflow, 34

T
target, 10
target host, 10
target user, 10
terminating connections, 112
TOFU, 92
too many authentication failures
troubleshooting, 124
troubleshooting, 123
  all attributes are not listed, 125
  authentication, 124
  debug logs, 123
  restarting PrivX, 124
  services are down, 124
  too many authentication failures, 124
  users are not listed, 125
trust on first use, 92
  two-factor authentication, 74

U
  upgrading PrivX, 37
  user authentication, 71
  users, 43, 63
    permissions, 64
  users are not listed
    troubleshooting, 125

W
  web connections, 24, 99, 111
  web proxy, 134
  workflow engine, 132
  workflows, 66
    approvals, 64