



# BeyondTrust

## **Password Safe 23.1 Cache User Guide**

## Table of Contents

---

<b>Secrets Cache User Guide</b> .....	<b>3</b>
Requirements: Roles and Settings .....	4
Installation Specifications .....	6
Configuration .....	7
Advanced Settings .....	11

## Secrets Cache User Guide

Secrets Cache is a lightweight proxy for the Password Safe API, providing high performance throughput for password requests and Secrets Safe secrets.

Running as a specified Password Safe user, Secrets Cache makes requests via the Password Safe API for all managed account credentials and secrets to which the user has access. It then caches the returned system and account details, request details, credentials, and secrets in an encrypted state.

API calls to the Secrets Cache serve the locally cached data. The cache is refreshed every five minutes by default, or sooner if a request is due to expire before that time.

If communication with Password Safe is lost, the last known good managed account credentials are served from the local cache, even if the associated request has expired.

# Requirements: Roles and Settings

## Roles and Features

The Password Safe user running the Secrets Cache must have at least one managed account Smart Rule configured with the requestor or requestor/approver role, and must also have the Secrets Safe feature assigned.

## ISA Role

The Secrets Cache does not currently support ISA-based password requests; therefore, it's important to ensure the user running the cache does not have the ISA role defined for any managed account Smart Rules.

## Access Policy

### Auto Approval

The managed account Smart Rule configured with the requestor or requestor/approver roles must have an access policy assigned that has **View Password** access set to **Auto Approve**.

Policy Types

At least one Policy Type must be enabled in order for the Access Policy to take effect.

View Password

Approvers

.....  Auto Approve

### Daily Recurrence - Multi-day Checkouts

If the access policy is configured for **Daily** recurrence, ensure **Allow multi-day check-outs of accounts** is enabled.

All Day

Date Range  End Date

Start Date

End Date

Recurrence

Repeat

Repeat Presets

Allow multi-day check-outs of accounts.

## Managed Account Settings

### Enable for API Access

Ensure this option is enabled for managed accounts that will be cached.

### Default Release Duration

The **Default Release Duration** is used to determine how long account credentials are cached before being renewed.

### Concurrent Requests

If the managed accounts configured to be cached will also be used by other Password Safe users at the same time, concurrent requests should be set to zero (0 denotes unlimited) or a value greater than one. Requests performed by the Secrets Cache count as a request.

## Supported Operating Systems

- Windows Server 2012 R2 and above releases
- RHEL 64 bit version 7 or higher

## Supported APIs

- POST Auth/SignAppln
- POST Auth/Signout
- GET Requests
- POST Requests
- POST Aliases/{aliasId}/Requests
- GET Credentials/{requestId}
- GET Aliases/{aliasId}/Credentials/{requestId}
- GET ManagedAccounts
- GET ManagedAccounts?systemName={systemName}&accountName={accountName}
- GET Aliases
- GET Secrets/{secretId}/file/download
- GET Secrets/{secretId}
- GET Secrets



For details on each method, please see the [BeyondInsight and Password Safe API Guide](https://www.beyondtrust.com/docs/beyondinsight-password-safe/ps/api/index.htm) at <https://www.beyondtrust.com/docs/beyondinsight-password-safe/ps/api/index.htm>.

## Installation Specifications

Windows	
Installer	pspca-<version>-x64.exe
Location	C:\Program Files\BeyondTrust\Secrets Cache\pspca
Service Control	sc stop pspca, sc start pspca
Linux	
Installer	rpm -i PSPCA-<version>.x86_64.rpm
Location	/opt/pbps/pspca
Service Control	systemctl stop pspca, systemctl start pspca



**Note:** For Linux - In versions prior to 23.1, the RPM package name was *beyondtrust-password-safe-cache*. In 23.1 and above, the RPM package name is *beyondtrust-secrets-cache*. If a *beyondtrust-password-safe-cache* package is currently installed, it must be removed before installing Secrets Cache.

## Configuration



**Note:** All configuration changes must be run from an elevated Command Shell or prompt.

To configure the cache, call Secrets Cache with the **cfg** options **pspca cfg <args>**.

```
# /opt/pbps/pspca cfg
```

Config:

- Log File (log\_file):
  - Windows: C:\Program Files\BeyondTrust\Secrets Cache\logs\pspca.log
  - Linux: /var/opt/pbps/log/pspca.log
- Log Level (log\_level): INFO
- Password Safe:
  - Host (host): pbps\_bi.example.com
  - API RunAS (username): psreq
  - API Key (key): \*\*\*\*\*
- REST API Server: Listen Address (address): 0.0.0.0:443

Client API (Secrets Cache connections to Password Safe):

- Certificate Validation (password\_safe\_verify): disabled
- Ciphers List:
  - ECDH+AESGCM:ECDH+CHACHA20:DH+AESGCM:ECDH+AES256:DH+AES256:ECDH+AES128:DH+AES:RSA+AESGCM:RSA+AES:!aNULL:!MD5:!DSS
- SSLv2: disabled
- SSLv3: disabled
- TLSv1: disabled
- TLSv1.1: enabled
- TLSv1.2: enabled

REST Server (API Client connections to Secrets Cache):

- Certificate (cache\_certificate): bi\_client.example.com
  - Issuer: ca.company.com
  - Fingerprint: 96 47 18 4a db 25 d8 42 84 c4 ad e3 08 58 1f 1f ba 9a bc 91
- Certificate Validation (cache\_client\_verify): disabled
- Ciphers List:
  - ECDH+AESGCM:ECDH+CHACHA20:DH+AESGCM:ECDH+AES256:DH+AES256:ECDH+AES128:DH+AES:RSA+AESGCM:RSA+AES:!aNULL:!MD5:!DSS
- SSLv2: disabled
- SSLv3: disabled

- TLSv1: disabled
- TLSv1.1: enabled
- TLSv1.2: enabled

## Usage: cfg [options]

-L --log_file=<arg>	Log File name
-l --log_level=<arg>	Logging level (error, warning, info, debug, trace)
-h --host=<arg>	Password Safe host[:port]
-a --address=<arg>	Secrets Cache Listen Address[:port]
-u --username=<arg>	<username> Password Safe API requestor username
-k --key=<arg>	<key> Password Safe API Key
-c --client_certificate=<arg>	Password Safe Client certificate file (pem)
-V --password_safe_verify=<arg>	Password Safe certificate validation 0=no server validation 1=server validation required
-C --client_clear	Clears the Password Safe client certificate
-T --password_safe_ca=<arg>	Trusted Password Safe CA certificate file(s) (pem)
-s --cache_certificate=<arg>	Secrets Cache server certificate file (pem)
-v --cache_client_verify=<arg>	Secrets Cache client certificate validation 0=no client validation 1=client validation required
-t --cache_client_trusted_ca=<arg>	Secrets Cache trusted client CA certificate file(s) (pem)
-p --pem=<arg>	PEM encoded private key for Password Safe or cache certificate
-P --pem_passwd=<arg>	PEM private key passphrase
--export=<arg>	Export the Secrets Cache configuration
--import=<arg>	Import the Secrets Cache configuration
--export_db=<arg>	Export the Secrets Cache data
--import_db=<arg>	Import the Secrets Cache data
--passwd=<arg>	Password to be used to encrypt/decrypt the exported Secrets Cache configuration
-? --help	Display this usage message

## Examples



**Example:** Configure the target Password Safe server that the Secrets Cache will communicate with:

*Windows:*

```
C:\Program Files\BeyondTrust\Secrets Cache\pspca.exe cfg -u psreq -k 638AA550-37C4-7126-A9C1-22186D5A40A0 -h pbps_bi.example.com
```

*Linux:*



```
# /opt/pbps/pspca cfg -u psreq -k 638AA550-37C4-7126-A9C1-22186D5A40A0 -h pbps_  
bi.example.com
```



**Example:** To validate the Password Safe server certificate, define a trusted CA and require validation:

Windows:

```
C:\Program Files\BeyondTrust\Secrets Cache\pspca.exe cfg -T password_safe_ca.pem -V 1
```

Linux:

```
# /opt/pbps/pspca cfg -T password_safe_ca.pem -V 1
```



**Example:** To connect the Secrets Cache to the Password Safe REST API using the client certificate:

Windows:

```
C:\Program Files\BeyondTrust\Secrets Cache\pspca.exe cfg -c client_cert.pem -p client_  
key.pem -P <pem_password>
```

Linux:

```
# /opt/pbps/pspca cfg -c client_cert.pem -p client_key.pem -P <pem_password>
```



**Example:** To change the local configuration for logging and the listen port of the Secrets Cache:

Windows:

```
C:\Program Files\BeyondTrust\Secrets Cache\pspca.exe cfg -L /var/log/pspca.log -l warning  
-a 0.0.0.0:8443
```

Linux:

```
# /opt/pbps/pspca cfg -L /var/log/pspca.log -l warning -a 0.0.0.0:8443
```



**Example:** To provide custom settings for the server certificate used by the Secrets Cache REST interface:

Windows:

```
C:\Program Files\BeyondTrust\Secrets Cache\pspca.exe cfg -s server_cert.pem -p server_  
key.pem -P <pem_password>
```



Linux:

```
# /opt/pbps/pspca cfg -s server_cert.pem -p server_key.pem -P <pem_password>
```



**Example:** To require client certificates to be provided to the Secrets Cache REST interface using a defined trusted client CA and require validation:

Windows:

```
C:\Program Files\BeyondTrust\Secrets Cache\pspca.exe cfg -t client_ca.pem -v 1
```

Linux:

```
# /opt/pbps/pspca cfg -t client_ca.pem -v 1
```



**Example:** To export the Secrets Cache configuration for recovering and/or replicating the cache:

Windows:

```
C:\Program Files\BeyondTrust\Secrets Cache\pspca.exe cfg --export=cache_config.cfg --export_db=cache_data.cfg --passwd <secret>
```

Linux:

```
# /opt/pbps/pspca cfg --export=cache_config.cfg --export_db=cache_data.cfg --passwd <secret>
```



**Example:** To import the Secrets Cache configuration for recovering and/or replicating the cache:

Windows:

```
C:\Program Files\BeyondTrust\Secrets Cache\pspca.exe cfg --import=cache_config.cfg --import_db=cache_data.cfg --passwd <secret>
```

Linux:

```
# /opt/pbps/pspca cfg --import=cache_config.cfg --import_db=cache_data.cfg --passwd <secret>
```

## Advanced Settings

The following advanced settings can be configured outside the configuration tool:

- **LogFile:**
  - **Windows LogFile:** Location of log file (default `C:\Program Files\BeyondTrust\Secrets Cache\logs\pspca.log`).
  - **Linux LogFile:** Location of log file (default `/var/opt/pbps/log/pspca.log`).
- **runuser:** The unprivileged user that is used to run the cache service on Linux (default `nobody`).
- **http\_rest:** Define custom settings for the HTTP REST interface.
  - **listen\_port:** The port the cache uses to listen for incoming API calls (default `443`).
  - **listen\_host:** The interface the cache uses to listen for incoming API calls (default `0.0.0.0`).
- **password\_safe:** Define settings for Password Safe interactions.
  - **managed\_accounts\_limit:** Maximum number of managed accounts to retrieve from Password Safe (default `100000`).
  - **secrets\_limit:** Maximum number of secrets to retrieve from Secrets Safe (default: `100000`).
  - **rotation\_policy:** Can be set to `1` (rotate on retrieval) or `2` (never rotate) if **Allow API Rotation Override** is enabled in Password Safe's access policy (default `0`).
  - **http\_timeout:** HTTP timeout interval in seconds (default `60`).
  - **request\_reason:** Reason provided for the request (default: **Secrets Cache Refresh**).
  - **refresh\_interval:** How often the cache checks with Password Safe in milliseconds (default `300000`).
  - **use\_prev\_creds:** Set to `1` to server up a previous credential if the current credential is null (default `0`).

## Windows

Windows advanced settings are stored in the registry. If the setting is not there, it uses the default value:

- `[HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\BeyondTrust\PBPS\pspca_cfg]`
  - `LogFile=C:\Program Files\BeyondTrust\Secrets Cache\logs\pspca.log`
- `[HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\BeyondTrust\PBPS\pspca_cfg\http_rest]`
  - `listen_port=443`
  - `listen_host=0.0.0.0`

## Linux

The advanced options are stored in JSON format in `/etc/opt/pbps/pspca.conf`. If an option is not included, the default value is used.

```
{
  "LogFile": "/var/opt/pbps/log/pspca.log",
  "runuser": "nobody",
  "http_rest": {
    "listen_port": 443,
    "listen_host": "0.0.0.0"
  },
  "password_safe": {
```

```
"http_timeout": "psapi",  
"managed_accounts_limit": 100000,  
"refresh_interval": 300000,  
"request_reason": "Secrets Cache Refresh",  
"rotation_policy": 0,  
"use_prev_creds": false  
}  
}
```