BeyondTrust

Password Safe 23.3
Cache User Guide

TC:2/16/2024

Table of Contents

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.

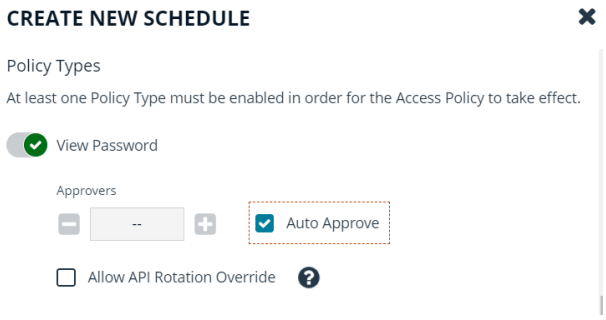
ISA Role

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

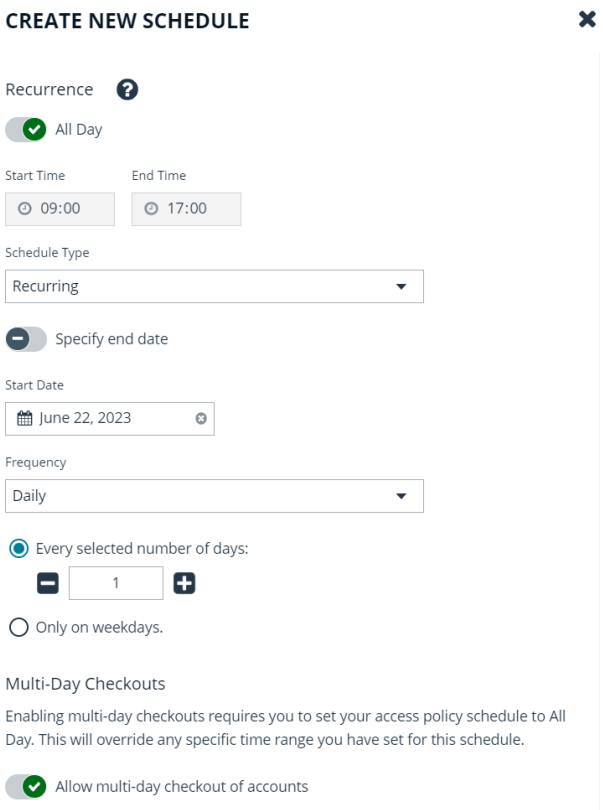
Auto Approval

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**.



Daily Recurrence - Multi-day Checkouts

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



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/SignAppIn
- 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-Safe/Secrets/{secretId}/file/download
- GET Secrets-Safe/Secrets/{secretId}
- GET Secrets-Safe/Secrets

For details on each method, please see the BeyondInsight and Password Safe API Guide 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:

Windows

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

LogFile

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\LogFile
```

Full path to the log file. If not provided, no log file is created. This value is set to *<pspca install directory>\logs\pspca.log* during the first run of pspca.

Type: String

runuser

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\runuser
```

Account to use to run the Secrets Cache service on Linux. This registry value is set to *nobody* during the first run of pspca, but is not used on Windows.

Type: String.

http_rest\listen_host

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\http_rest\listen_host
```

The IP address on which Secrets Cache listens for REST API requests. This value is set to 0.0.0.0 during the first run of pspca.

Type: String.

http_rest\listen_port

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\http_rest\listen_port
```

The port on which Secrets Cache listens for REST API requests. This value is set to 443 during the first run of pspca.

Type: String or DWORD.

password_safe\host

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\host
```

Password Safe API hostname/IP and port number.

Type: String.

password_safe\http_timeout

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\http_timeout
```

Maximum number of seconds to wait for a response from the Password Safe API (default: 60). If set to 0, Secrets Cache will wait indefinitely.

Type: String.

password_safe\managed_accounts_limit

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\managed_accou
nts_limit
```

Maximum number of managed accounts to retrieve from Password Safe (default: 100000). This is used to set the value of the limit query parameter in the **GETManagedAccounts** request that is sent to the Password Safe API during a cache refresh.

Type: DWORD.

password_safe\refresh_interval

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\refresh_inter
val
```

Default cache refresh interval (R) in milliseconds (default: 300000). During a cache refresh, the expiry date of each credential release request is examined, and the earliest expiry date (E) and the current time (T) are determined. If T < E < T + R, then the next cache refresh will be rescheduled to take place at time E.

Type: DWORD or QWORD.

password_safe\request_duration

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\request_duration
```

Default credential release request duration in minutes (default: 120). This is used to set the value of the **DurationMinutes** field in the body of a **POST Requests or POST Aliases/{AliasId}/Requests** request that is sent to the Password Safe API during a cache refresh, if the managed account or managed account alias does not have a **DefaultReleaseDuration** value set.

Type: DWORD or QWORD.

password_safe\request_reason

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\request_reason
```

Request reason (default: *Secrets Cache Refresh*, legacy default *Password Cache Refresh*). This is used to set the value of the Reason field in the body of a **POST Requests** or a **POST Aliases/{AliasId}/Requests** request that is sent to the Password Safe API during a cache refresh.

Type: String.

password_safe\rotation_policy

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\rotation_policy
```

Credential rotation policy:

- 0 - always rotate (default)
- 1 - rotate only if credentials are retrieved
- 2 - never rotate

This is used in the following situations:

- To set the value of the **RotateOnCheckin** field in the body of a **POST Requests** or a **POST Aliases/{AliasId}/Requests** request that is sent to the Password Safe API during a cache refresh.
- To decide whether to update a credential release request to rotate the credential on check-in or expiry when that credential is retrieved via the **Secrets Cache REST API**.

Type: DWORD.

password_safe\secrets_limit

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\secrets_limit
```

Maximum number of secrets to retrieve from Secrets Safe (default: 100000). This is used to set the value of the limit query parameter in the **GETSecrets-Safe/Secrets** request that is sent to the Password Safe API during a cache refresh.

Type: DWORD.

password_safe\use_prev_creds

```
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\BeyondTrust\PBPS\pspca_cfg\password_safe\use_prev_creds
```

Whether to return previous credentials if current credentials are unavailable when credentials are requested via the **Secrets Cache REST API** (default: 0 - no).

Type: DWORD (0/1).

Linux

For the advanced options and values available, refer to the Windows section, above. For Linux, the options and selections are stored in JSON format in **/etc/opt/pbps/pspca.conf**. The example below shows how the options are entered for the file. 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
    }
}
```