Cigent CLI Reference Guide

Command Line Interface for FDE, PBA, and SED, V4.0.7

May 2026

Table of Contents

1. Introduction

The Cigent Command Line Interface (CLI) provides administrators and automation systems with programmatic access to Cigent security products. It enables management of Full Drive Encryption (FDE), Pre-Boot Authentication (PBA), and Self-Encrypting Drive (SED) features from a terminal or script.

Supported Targets

FDE — Cigent Full Drive Encryption. Encrypts the entire Windows OS volume using AES-256 XTS encryption with pre-boot authentication.

PBA — Cigent Pre-Boot Authentication. When combined with a supported self-encrypting drive (SED), creates a data-at-rest protection solution requiring authentication before booting. Supports single and multi-drive configurations.

SED — Self-Encrypting Drive management. Direct control of drive-level encryption features including range management, initialization, and secure erasure.

When to Use the CLI

The CLI is designed for environments where graphical interaction is impractical or undesirable:

- Remote Management and Monitoring (RMM) tool integration
- Scripted fleet deployment and configuration
- Automated compliance and policy enforcement
- Headless or server environments without a display

Supported Platforms

The Cigent CLI is available on Windows and Linux. Platform availability varies by target:

- FDE — Windows only.
- PBA — Windows and Linux.
- SED — Windows and Linux.

2. Getting Started

Syntax

```
CigentCLI <command> --target <fde|pba|sed> [options]
```

Commands are prefixed with the CigentCLI executable name. The --target option specifies which Cigent product the command applies to. Most commands require administrator credentials via --user and --password (or --secrets_stdin).

Global Options

`--json` Output results in JSON format instead of human-readable text.

`--secrets_stdin` Read credentials from STDIN as a JSON object. See Section 3 for details.

Status Codes

Every command returns a binary process exit code: 0 indicates success and 1 indicates an error. Scripts should check the process exit code directly (e.g., $? in Bash, $LASTEXITCODE in PowerShell).

When --json is specified, the detailed status code is provided in the status.code field of the JSON response. For non-JSON output, the status code is included in the command's output text upon completion.

The following table defines all status codes used by the CLI:

Code	Description
0	Success
1	Internal error (unexpected failure; check logs)
2	Invalid parameter
3	Authentication failed (invalid credentials or insufficient privileges)
4	Target Application not found Note: Also used for Incompatible Target Versions
5	Drive not found
6	Access denied
7	Operation in progress (indicates the operation is not finished and there will be another status update)

Use the download_logs command to retrieve logs for support purposes, or purge_logs to clear them.

Version Compatibility

When the CLI version is incompatible with the installed target application, the CLI will return status code 4.

JSON Response Format

When --json is specified, all commands return a JSON object containing a status field. Commands that return data include an additional data field. Most commands that perform an action (set, add, remove, etc.) return only the status:

Success:

```
{ "status": { "code": 0 } }
```

Failure:

```
{
  "status": {
```

```
    "code": 3
  }
}
```

Commands that query or list data (status, list_users, list_devices, etc.) include their results in the data field. These responses are shown in the individual command sections.

Note: For readability, the JSON examples in this guide are shown in pretty-printed form. Actual CLI output may be emitted as compact JSON on a single line.

Unix Epoch Time

Some fields and options use Unix epoch time. Unix epoch time is the number of seconds that have elapsed since January 1, 1970 00:00:00 UTC. Unless otherwise stated, epoch values in this guide are expressed in seconds and use UTC.

A value of 0 may have a special meaning depending on the command. For example:

- `expiration = 0 means no expiration`
- `erasure_epoch = 0 means disabled`

Examples:

- 1923683525 = a specific date and time in UTC
- 0 = disabled or no expiration, depending on the field

When automating the CLI, ensure your script generates epoch values in UTC and in seconds, not milliseconds.

3. Authentication & Secure Secret Handling

Most CLI commands require administrator credentials to perform privileged actions. Credentials can be provided in two ways (described below).

Important: Separate Credential Domains

The username and password used by the CLI are specific to the target application (FDE or PBA). They are not your operating system login, Active Directory credentials, or any other system account. FDE and PBA each maintain their own independent user databases:

- FDE users are created and managed through FDE user management commands. These credentials are used for pre-boot authentication and CLI access to FDE.
- PBA users are created and managed through PBA user management commands. These credentials are used for PBA pre-boot authentication and CLI access to PBA.
- FDE and PBA do not share users. A user account created in FDE does not exist in PBA, and vice versa.
- The --password is the admin password for the target application, not the OS admin password.
- The --target_password (used in add_user and edit_user) is the password being set for the target user within that application.

Command-Line Arguments (Interactive Use)

For interactive use, credentials can be passed directly as command-line arguments:

```
CigentCLI status --target fde --user admin --password "MyP@ssw0rd"
```

Warning: Command-line arguments may be visible in process listings and shell history. This method is not recommended for RMM or scripted environments.

STDIN Secret Ingestion (--secrets_stdin)

For RMM and scripted environments, the --secrets_stdin flag provides a secure mechanism for passing credentials. Secrets provided via STDIN are never written to log files, echoed to the console, or persisted to disk by the CLI. When --secrets_stdin is specified:

- STDIN is read exactly once
- STDIN must contain a single UTF-8 encoded JSON object
- Secrets are never logged, echoed, or persisted
- Only recognized keys are accepted; extraneous keys cause failure (status 2)
- Malformed JSON causes failure (status 2)
- Missing required keys cause failure (status 2)

Case A: Single Secret (Most Commands)

Most commands require only the administrator password:

```
{ "password": "MyP@ssw0rd" }
```

Case B: Two Secrets (User Management)

Commands that create or modify user credentials (add_user, edit_user) require both the admin password and the target user password:

```
{ "password": "AdminP@ss", "target_password": "UserP@ss123" }
```

4. General Commands

These commands are not specific to a target application.

cigent_status

Show all the Cigent (or otherwise branded, such as Digistor) products available on this device. These products include PBA, FDE, CLI, and Data Defense.

Usage

```
CigentCLI cigent_status [--json]
```

Example

```
CigentCLI cigent_status --json
```

JSON Response

```
{
  "data": {
      "Cigent CLI": "v4.0.3",
      "Cigent Data Defense": "Not installed",
      "Cigent Full Drive Encryption": "v1.2.1.11",
      "Cigent PBA": "v2.0.1.16"
    },
  "status": {
      "code": 0
    }
}
```

help

Displays:

```
Please see the PDF reference guide included with this application.
For the latest documentation, visit support.cigent.com.
```

Usage

```
CigentCLI help [--json]
```

Example

```
CigentCLI help --json
```

JSON Response

```
{
  data: {
    help:
      "-----------------------------------------------------------------\n" +
      "Please see the PDF reference guide included with this application.\n" +
      "For the latest documentation, visit support.cigent.com.\n" +
      "-----------------------------------------------------------------\n",
    version: "Cigent CLI Utility v4.0.3"
  },
  status: {
    code: 0
  }
}
```

5. Full Drive Encryption (FDE)

Cigent Full Drive Encryption provides transparent, full-drive encryption using AES-256 XTS to protect data at rest. Available on Windows. For scripted and RMM environments, use --secrets_stdin instead of --password (see Section 3). All commands require --target fde.

5.1 Status & Information

status

Returns overview of the FDE system from the dashboard and the system report.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI status --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe status --target fde --user test_fde_admin --password 1amAnAdmin2345! --
json
```

JSON Response


```
{
 "data": {
   "current_version": "1.2.1.11",
   "devices": [
      {
          "fde_encryption_percentage": "17.01",
          "fde_status": 1,
          "mount": "C:"
        }
    ],
   "license": "Trial",
   "license_expires": 1777562224,
   "maintenance_expires": 1869578224,
   "protected_volumes": 1,
   "total_users": 2,
   "workstation_id": "xHhGQFVBc8NHi8ALqkO"
  },
 "status": {
   "code": 0
  }
}
```


list_devices

List the USB devices and drives attached to the system, including FDE encryption status.

Options

Option	Required	Description
`--target`	Yes	Target application
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_devices --target fde [--json]
```

Example

```
.\CigentCLI.exe list_devices --target fde --json
```

JSON Response


```
{
 "data": {
   "drives": [
      {
          "fw": "ECFL13P7",
          "is_usb": false,
          "model": "Cigent Secure SSD+",
          "path": "\\\\.\\PHYSICALDRIVE0",
          "serial": "5112306061110000032",
          "size_bytes": 480103980544,
          "type": "internal",
          "volumes": [
               {
                     "fde_encryption_percentage": "17.06",
                     "fde_status": 1,
                     "guid": "{a9e8d4e8-d7bd-4501-9d94-e4bd9f95a9f9}",
                     "mount_path": "C:",
                     "partition_table_type": "gpt",
                     "volume_label": "",
                     "volume_size_bytes": 479296997888
                    }
              ]
         }
     ]
  },
 "status": {
   "code": 0
  }
}
```


5.2 User Management

add_user

Add a user to FDE. Note that FDE and PBA have different sets of users.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier

Option	Required	Description
`--password`	Yes	Admin password
`--target_user`	Yes	Username
`--email_address`	No	Email for user creation or notifications
`--target_password`	Yes	User password
`--role`	Yes	Role identifier
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI add_user --target fde --user <user> --password <password> --target_user
<target_user> [--email_address <email_address>] --role <role> --target_password
<password> [--json]
```

Example

```
.\CigentCLI.exe add_user --target fde --user test_fde_admin --password 1amAnAdmin2345!
--target_user demo_fde_user --target_password DemoFDEUser234! --role user --
email_address demo_fde_user@example.com --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

edit_user

Edit an existing FDE user. The target_user selects which user is being edited; the username itself cannot be changed. Password, role, and email address can be modified.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target_user`	Yes	Username
`--email_address`	No	Email for user creation or notifications
`--target_password`	No	User password
`--role`	No	Role identifier
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI edit_user --target fde --user <user> --password <password> --target_user
<target_user> [--email_address <email_address>] [--role <role>] --target_password
<password> [--json]
```

Example

```
.\CigentCLI.exe edit_user --target fde --user test_fde_admin --password 1amAnAdmin2345!
--target_user demo_fde_user --target_password DemoFDEUser345! --role admin --
email_address demo_fde_user+updated@example.com --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

remove_user

Remove a user from FDE. Removing a user permanently deletes them from the FDE environment. The user will no longer be able to authenticate to access the protected OS or the administrative console.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target_user`	Yes	Username
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI remove_user --target fde --user <user> --password <password> --target_user
<target_user> [--json]
```

Example

```
.\CigentCLI.exe remove_user --target fde --user test_fde_admin --password
1amAnAdmin2345! --target_user demo_fde_user --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

list_users

Retrieve a list of users: username, role, email, smart card ID, USB Token ID, Security Key ID and modes.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_users --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe list_users --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response


```
{
 "data": {
   "users": [
      {
          "auth_factors": {
               "password": true,
               "smartcard": false
              },
          "role": "admin",
          "username": "jeff"
         },
      {
          "auth_factors": {
               "password": true,
               "smartcard": false
              },
          "role": "admin",
          "username": "test_fde_admin"
         }
     ]
  },
 "status": {
   "code": 0
  }
}
```


5.3 Enrollment

add_enrollment_code

Add an enrollment code for self-enrolling a smartcard. The enrollment code length mirrors the minimum password length, which defaults to 15 characters for FDE. It requires the actual code, the number of uses allowed, and the expiration date. Setting the usage count to 0 means unlimited uses and setting Expiration to 0 means no expiration. Expiration is expressed as a Unix epoch value in seconds (UTC).

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--code`	Yes	Enrollment/activation code value
`--count`	Yes	Count
`--expiration`	Yes	Expiration
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI add_enrollment_code --target fde --user <user> --password <password> --code
<code> --count <count> --expiration <expiration> [--json]
```

Example

```
.\CigentCLI.exe add_enrollment_code --target fde --user test_fde_admin --password
1amAnAdmin2345! --code FDEEnroll_code2 --count 5 --expiration 0 --json
```

JSON Response

```
{
  "data": {
    "code_id": "E9eO4yZr",
    "expires_at": 0,
    "uses_remaining": 5
  },
  "status": {
    "code": 0
  }
}
```

list_enrollment_codes

Display all active enrollment codes, showing IDs. Note that the actual codes are never displayed.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_enrollment_codes --target fde --user <user> --password <password> [--
json]
```

Example

```
.\CigentCLI.exe list_enrollment_codes --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response

```
{
 "data": {
   "enrollment_codes": [
     ]
  },
 "status": {
   "code": 0
  }
}
```

5.4 Authentication Settings

enable

Enable FDE authentication after disabling. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI enable --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe enable --target fde --user test_fde_admin --password 1amAnAdmin2345! --
json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

disable

Disable FDE authentication, useful for software installations that require multiple reboots. All settings and configuration are preserved while disabled. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
--user	Yes	Admin user identifier
--password	Yes	Admin password
--target	Yes	Target application
--secrets_stdin	No	Read credentials from STDIN as JSON. See Section 3.
--json	No	Output results in JSON format.

Usage

```
CigentCLI disable --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe disable --target fde --user test_fde_admin --password 1amAnAdmin2345! --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

set_remember_me

Set "remember me" for the pre-boot authentication. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
--user	Yes	Admin user identifier
--password	Yes	Admin password
--target	Yes	Target application
--set_bool	Yes	Sets true or false
--secrets_stdin	No	Read credentials from STDIN as JSON. See Section 3.
--json	No	Output results in JSON format.

Usage

```
CigentCLI set_remember_me --target fde --user <user> --password <password> --set_bool <set_bool> [--json]
```

Example

```
.\CigentCLI.exe set_remember_me --target fde --user test_fde_admin --password 1amAnAdmin2345! --set_bool true --json
```

JSON Response

```
{
 "status": {
   "code": 0
```

```
  }
}
```

get_remember_me

Retrieve the "remember me" setting for the pre-boot environment authentication.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_remember_me --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe get_remember_me --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response

```
{
 "data": {
   "value": false
  },
 "status": {
   "code": 0
  }
}
```

set_failed_login_lockout

Set how many failed logins are allowed before the user is locked out. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--count`	Yes	Count
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_failed_login_lockout --target fde --user <user> --password <password> --
count <count> [--json]
```

Example

```
.\CigentCLI.exe set_failed_login_lockout --target fde --user test_fde_admin --password
1amAnAdmin2345! --count 6 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

get_failed_login_lockout

Retrieve how many failed logins are allowed before the user is locked out.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_failed_login_lockout --target fde --user <user> --password <password> [--
json]
```

Example

```
.\CigentCLI.exe get_failed_login_lockout --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response

```
{
 "data": {
   "value": 5
  },
 "status": {
   "code": 0
  }
}
```

set_failed_login_erase_count

Set the failed login erase setting. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--count`	Yes	Count
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_failed_login_erase_count --target fde --user <user> --password <password>
--count <count> [--json]
```

Example

```
.\CigentCLI.exe set_failed_login_erase_count --target fde --user test_fde_admin --
password 1amAnAdmin2345! --count 1 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

get_failed_login_erase_count

Retrieve the failed login erase setting.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_failed_login_erase_count --target fde --user <user> --password <password>
[--json]
```

Example

```
.\CigentCLI.exe get_failed_login_erase_count --target fde --user test_fde_admin --
password 1amAnAdmin2345! --json
```

JSON Response

```
{
 "data": {
   "value": 0
```

```
  },
 "status": {
   "code": 0
  }
}
```

5.5 Password Policy

set_password_min_length

Set the minimum password length. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--count`	Yes	Count
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_password_min_length --target fde --user <user> --password <password> --
count <count> [--json]
```

Example

```
.\CigentCLI.exe set_password_min_length --target fde --user test_fde_admin --password
1amAnAdmin2345! --count 16 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

get_password_min_length

Retrieve the minimum password length.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_password_min_length --target fde --user <user> --password <password> [--
json]
```

Example

```
.\CigentCLI.exe get_password_min_length --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response

```
{
 "data": {
   "value": 15
  },
 "status": {
   "code": 0
  }
}
```

set_password_history

Set the password history setting, which is how many passwords are remembered so the user must use new passwords. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--count`	Yes	Count
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_password_history --target fde --user <user> --password <password> --count
<count> [--json]
```

Example

```
.\CigentCLI.exe set_password_history --target fde --user test_fde_admin --password
1amAnAdmin2345! --count 5 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

get_password_history

Retrieve the password history setting, which is how many passwords are remembered so the user must use new passwords.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_password_history --target fde --user <user> --password <password> [--
json]
```

Example

```
.\CigentCLI.exe get_password_history --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response

```
{
 "data": {
   "value": 1
  },
 "status": {
   "code": 0
  }
}
```

5.6 Licensing

set_license

Add a new license key to the FDE by providing the license key text directly OR passing in a file. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--license_key`	No	The text of the license key
`--license_file`	No	Path to a license file
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_license --target fde --user <user> --password <password> [--license_key
<license_key>] [--license_file <license_file>] [--json]
```

Example

```
.\CigentCLI.exe set_license --target fde --user test_fde_admin --password
1amAnAdmin2345! --license_key <text> --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

get_request_code

Generate and return a request code to allow the user to request a license for this device.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_request_code --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe get_request_code --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response

```
{
 "data": {
   "request_code": "<text>"
  },
 "status": {
   "code": 0
  }
}
```

5.7 Maintenance

erase_drive

Cryptographically erase the protected data by erasing the encryption metadata, ensuring all data on the drive is unrecoverable. Note that the protected volume is not deleted but its contents are no longer usable. Once complete, power off the system.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI erase_drive --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe erase_drive --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

Note: This is a destructive operation. If the target drive is the OS drive, the system will likely crash before a JSON response can be returned to the console.

uninstall

Uninstall the FDE. Once initiated, the application will close and proceed to Windows where decryption will automatically start. Once decryption is complete, the application can be fully removed using Windows Add/Remove Programs. Requires a reboot.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI uninstall --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe uninstall --target fde --user test_fde_admin --password 1amAnAdmin2345!
--json
```

JSON Response

Note: This command initiates drive decryption. The CLI process remains active during decryption and the system automatically reboots upon completion. No JSON response is returned to the console.

download_logs

Retrieve the FDE logs from the device.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI download_logs --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe download_logs --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response


```
{
 "data": {
   "logs": [
      {
          "action": 3,
          "target_user": "test_fde_admin",
          "timestamp": 1774988871,
          "user": "jeff"
         },
      {
          "action": 3,
          "target_user": "demo_fde_user",
          "timestamp": 1774989256,
          "user": "test_fde_admin"
         },
      {
          "action": 4,
          "target_user": "demo_fde_user",
          "timestamp": 1774989262,
          "user": "test_fde_admin"
         },
      {
          "action": 5,
          "target_user": "'demo_fde_user'",
          "timestamp": 1774989268,
          "user": "test_fde_admin"
         },
      {
          "action": 3,
          "target_user": "demo_fde_user",
          "timestamp": 1774990531,
          "user": "test_fde_admin"
         },
      {
          "action": 4,
          "target_user": "demo_fde_user",
```

```
          "timestamp": 1774990537,
          "user": "test_fde_admin"
         },
      {
          "action": 5,
          "target_user": "'demo_fde_user'",
          "timestamp": 1774990543,
          "user": "test_fde_admin"
         }
     ]
  },
 "status": {
   "code": 0
  }
}
```


purge_logs

Purge the FDE logs from the device. An entry will be made in the log.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI purge_logs --target fde --user <user> --password <password> [--json]
```

Example

```
.\CigentCLI.exe purge_logs --target fde --user test_fde_admin --password
1amAnAdmin2345! --json
```

JSON Response


```
{
 "status": {
   "code": 0
  }
}
```

6. Pre-Boot Authentication (PBA)

Cigent Pre-Boot Authentication, combined with a supported self-encrypting drive, provides data-at-rest protection requiring authentication before the operating system boots. Available on Windows and Linux. Supports single and multi-drive configurations. For scripted and RMM environments, use --secrets_stdin instead of --password (see Section 3). All commands require --target pba.

6.1 Installation & Updates

install

Installs the PBA on the system. Note: provide a license key OR license file. If a username isn't provided the password becomes the setup password.

Options

Option	Required	Description
`--drive`	Yes	Drive or device reference
`--pba_bin_file`	Yes	Path to the PBA binary file for install or media creation
`--target_user`	No	The first admin user
`--target_password`	No	The default password (see PBA manual) will be used if this option is not set
`--license_file`	No	Path to a license file
`--license_key`	No	
`--allow_os`	No	Required flag if the target drive is the OS drive
`--raid`	No	RAID mode for PBA installation media/install
`--enable`	No	Enable PBA during install
`--no_verify`	No	Skip verification during PBA install
`--autoimport`	No	Auto-import drives during PBA install
`--open`	No	Open mode for PBA installation media/install
`--add_algos`	No	Install PBA with additional smartcard algorithms enabled
`--target`	Yes	Target application (pba)
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI install --target pba --drive <drive> --pba_bin_file <pba_bin_file> --
target_user <target_user> [--license_file <license_file>] [--license_key <license_key>]
[--allow_os] [--raid] [--enable] [--no_verify] [--autoimport] [--open] [--addl_algos] -
-target_password <password> [--json]
```

Example

```
.\CigentCLI.exe install --target pba --target_user test_pba_admin --target_password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --allow_os --pba_bin_file .\pba_v2.0.1.19.pkg --
enable --json
```

JSON Response

```
{
  "status": {
```

```
    "code": 0
  }
}
```

create_installation_media

Load the PBA installation material to a USB key. Note that the process will destroy any existing data on the key.

Options

Option	Required	Description
`--drive`	Yes	(from list_usb_drives)
`--raid`	Yes	(no value, just a flag)
`--open`	Yes	(no value, just a flag)
`--alt`	Yes	(no value, just a flag)
`--pba_bin_file`	Yes	Path to the PBA binary file for install or media creation
`--license_key`	No	
`--license_file`	No	Path to a license file
`--target`	Yes	Target application (pba)
`--brand`	Yes	Brand values: CIGENT = 1 DIGISTOR = 2 SEAGATE = 3 EVERFOX = 4 KANGURU = 5 VRS = 6 SWISSBIT = 7
`–force`	No	Allows the PBA to be installed on the OS drive (no value, just a flag)
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI create_installation_media --target pba --drive <drive> --raid --open --alt --
pba_bin_file <pba_bin_file> [--license_key <license_key>] [--license_file
<license_file>] [--json]
```

Example

```
.\CigentCLI.exe create_installation_media --target pba --drive \\.\PHYSICALDRIVE1 --brand 1 --
pba_bin_file .\pba_v2.0.1.16.pkg --force --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

update_pba

Updates the PBA on the specified drive, using the file provided. A digital signature verification is performed first to ensure integrity and authenticity. Failure of the signature verification will prevent the update.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--file`	Yes	File path for PBA update binary
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI update_pba --target pba --user <user> --password <password> --drive <drive> --file <file> [--json]
```

Example

```
.\CigentCLI.exe update_pba --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --pba_bin_file .\pba_v2.0.1.16.pkg --allow_os --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

uninstall

Uninstall the PBA. This removes all files, configuration, and user information. The OS environment will be preserved and boot normally. The action will start immediately. Multidrive systems: Enable Include secondary drives option (recommended) to remove protection from non-primary drives in addition to the primary.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--include_secondary`	Yes	The existence of the flag sets the behavior
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI uninstall --target pba --user <user> --password <password> --drive <drive> --include_secondary [--json]
```

Example

```
.\CigentCLI.exe uninstall --target pba --user test_pba_admin --password DemoPBAUser345!
--drive \\.\PHYSICALDRIVE0 --include_secondary --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

6.2 Status & Information

status

Returns overview of the PBA platform and attached USB devices and drives.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI status --target pba --user <user> --password <password> --drive <drive> [--
json]
```

Example

```
.\CigentCLI.exe status --target pba --user test_pba_admin --password DemoPBAUser345! --
drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "current_version": "2.0.1.16",
   "drive_serial": "511230606111000032",
   "drives": [
      {
          "brand": "Cigent",
          "fw": "ECFL13P7",
          "is_usb": false,
          "model": "Cigent Secure SSD+",
          "path": "\\\\.\\PHYSICALDRIVE0",
          "provisioned": true,
          "security_support": [
               "Opal SSC v2*"
              ],
          "serial": "511230606111000032",
          "size_bytes": 480103980544,
          "state": 23,
          "type": "internal"
```

```
        }
      ],
    "edition": "SE,RAID",
    "license": "Perpetual",
    "license_expires": 0,
    "maintenance_expires": 1806505769,
    "max_secondary_drives": 4,
    "preloaded": false,
    "protected_drives": 1,
    "total_users": 2
   },
  "status": {
    "code": 0
   }
}
```


list_devices

List the USB devices and drives attached to the system.
See Appendix D for decoding the drive state mask value.

Options

Option	Required	Description
`--target`	Yes	Target application
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_devices --target pba [--json]
```

Example

```
.\CigentCLI.exe list_devices --target pba --json
```

JSON Response


```
{
  "data": {
    "drives": [
      {
        "brand": "Cigent",
        "fw": "ECFL13P7",
        "is_usb": false,
        "model": "Cigent Secure SSD+",
        "path": "\\\\.\\PHYSICALDRIVE0",
        "provisioned": true,
        "security_support": [
          "Opal SSC v2*"
        ],
        "serial": "511230606111000032",
        "size_bytes": 480103980544,
        "state": 7,
        "type": "internal"
      },
      {
        "brand": "Unknown",
        "fw": "PMAP",
        "is_usb": true,
        "model": "DataTraveler 2.0",
        "path": "\\\\.\\PHYSICALDRIVE1",
```

```
        "provisioned": false,
        "security_support": [
          "none"
        ],
        "serial": "5B6C1E8A8826",
        "size_bytes": 1031798272,
        "state": 8,
        "type": "removable"
      }
    ]
  },
  "status": {
    "code": 0
  }
}
```


6.3 User Management

add_user

Add a user to PBA. Note that FDE and PBA have different sets of users.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target_user`	Yes	Username
`--email_address`	No	Email for user creation or notifications
`--target_password`	Yes	User password
`--role`	Yes	Role identifier
`--target`	Yes	Target application
`--require_pw_change`	No	Require the user to change password on next login
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI add_user --target pba --user <user> --password <password> --drive <drive> --target_user <target_user> [--email_address <email_address>] --role <role> [--require_pw_change] --target_password <password> [--json]
```

Example

```
.\CigentCLI.exe add_user --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --target_user demo_pba_user --target_password DemoPBAUser234! --role user --email_address demo_pba_user@example.com --json
```

JSON Response


```
{
```

```
 "status": {
   "code": 0
  }
}
```

edit_user

Edit an existing PBA user. The target_user selects which user is being edited; the username itself cannot be changed. Password, role, email, smartcard, USB token, and security key can be modified.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target_user`	Yes	Username
`--email_address`	No	Email for user creation or notifications
`--target_password`	No	User password
`--role`	No	Role identifier
`--target`	Yes	Target application
`--require_pw_change`	No	Require the user to change password on next login
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI edit_user --target pba --user <user> --password <password> --drive <drive> --target_user <target_user> [--email_address <email_address>] [--role <role>] [--require_pw_change] --target_password <password> [--json]
```

Example

```
.\CigentCLI.exe edit_user --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --target_user demo_pba_user --target_password DemoPBAUser456! --role admin --email_address demo_pba_user+updated@example.com --json
```

JSON Response

```
{
 "status": {
   "code": 1
  }
}
```

remove_user

Remove a user from PBA. Removing a user permanently deletes them from the PBA environment. The user will no longer be able to authenticate to access the protected OS or the administrative console.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target_user`	Yes	Username
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI remove_user --target pba --user <user> --password <password> --drive <drive>
--target_user <target_user> [--json]
```

Example

```
.\CigentCLI.exe remove_user --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --target_user demo_pba_user --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

list_users

Returns a list of users: username, role, email, smart card ID, USB Token ID, Security Key ID and modes.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_users --target pba --user <user> --password <password> --drive <drive>
[--json]
```

Example

```
.\CigentCLI.exe list_users --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "users": [
      {
          "auth_factors": {
               "cac": false,
               "password": true,
               "security key": false,
               "usb token": false
              },
          "email": "",
          "role": "admin",
          "username": "jeff"
         },
      {
          "auth_factors": {
               "cac": false,
               "password": true,
               "security key": false,
               "usb token": false
              },
          "email": "",
          "role": "admin",
          "username": "test_pba_admin"
         }
     ]
  },
 "status": {
   "code": 0
  }
}
```


6.4 Enrollment

add_enrollment_code

Add an enrollment code for self-enrolling a smartcard. The enrollment code length mirrors the minimum password length, which defaults to 8 characters for PBA. It requires the actual code, the number of uses allowed, and the expiration date. Setting the usage count to 0 means unlimited uses and setting Expiration to 0 means no expiration. Expiration is expressed as a Unix epoch value in seconds (UTC).

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--code`	Yes	Enrollment/activation code value
`--count`	Yes	Count
`--drive`	Yes	Drive
`--expiration`	Yes	Expiration
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI add_enrollment_code --target pba --user <user> --password <password> --code
<code> --count <count> --expiration <expiration> --drive <drive> [--json]
```

Example

```
.\CigentCLI.exe add_enrollment_code --target pba --user test_pba_admin --password
DemoPBAUser345! --code PBAEnroll_code12 --count 5 --expiration 0 --drive
\\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

remove_enrollment_code

Remove a single enrollment code associated with the provided code_id.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--code_id`	Yes	Activation/enrollment code
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI remove_enrollment_code --target pba --user <user> --password <password> --
drive <drive> --code_id <code_id> [--json]
```

Example

```
.\CigentCLI.exe remove_enrollment_code --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --code_id 7bHgQiPp --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

remove_all_enrollment_codes

Remove all enrollment codes.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application (pba)
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI remove_all_enrollment_codes --target pba --user <user> --password <password>
--drive <drive> [--json]
```

Example

```
.\CigentCLI.exe remove_all_enrollment_codes --target pba --user test_pba_admin --
password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

list_enrollment_codes

Display all active enrollment codes, showing IDs. Note that the actual codes are never displayed.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_enrollment_codes --target pba --user <user> --password <password> --
drive <drive> [--json]
```

Example

```
.\CigentCLI.exe list_enrollment_codes --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "enrollment_codes": [
     ]
  },
 "status": {
   "code": 0
  }
}
```

6.5 Drive Management

add_secondary_drive

Adds protection to a supported drive. This command supports drives that are NOT configured for another PBA.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application
`--drive`	Yes	Drive or device reference
`--target_drive`	Yes	Drive to be added
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI add_secondary_drive --target pba --user <user> --password <password> --drive
<drive> --target_drive <drive> [--json]
```

Example

```
.\CigentCLI.exe add_secondary_drive --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --target_drive \\.\PHYSICALDRIVE1 --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

remove_secondary_drive

Removes a protected drive and moves its status to non-protected.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--target`	Yes	Target application

Option	Required	Description
`--drive`	Yes	Drive or device reference
`--target_drive`	Yes	Drive to be removed
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI remove_secondary_drive --target pba --user <user> --password <password> --
drive <drive> --target_drive <drive> [--json]
```

Example

```
.\CigentCLI.exe remove_secondary_drive --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --target_drive \\.\PHYSICALDRIVE1 --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

import_secondary_drive

Migrates a non-protected drive to protected, for importable drives. Requires credentials from the original PBA: username and password.

Options

Option	Required	Description
`--user`	Yes	this PBA
`--password`	Yes	this PBA
`--target`	Yes	Target application
`--drive`	Yes	Drive or device reference
`--target_user`	Yes	Drive's PBA
`--target_password`	Yes	Drive's PBA
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI import_secondary_drive --target pba --user <user> --password <password> --
drive <drive> --target_user <target_user> --target_password <password> [--json]
```

Example

```
.\CigentCLI.exe import_secondary_drive --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --target_drive \\.\PHYSICALDRIVE1 --
target_user other_pba_admin --target_password other_pbA_p@ssword2 --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
```

```
}
```

6.6 Range Management

add_range

Add a range to the specified drive. The user can set the size (0 for all available) and the filesystem type. If the type is not specified it will be "ext".

Options

Option	Required	Description
`--password`	Yes	Admin password
`--user`	Yes	Admin user identifier
`--target`	Yes	Target application
`--range`	Yes	Range; avoids collision with role (-r)
`--drive`	Yes	Drive or device reference
`--size`	Yes	0 for all available
`--filesystem_type`	No	If not specified the range will be formatted as "ext"
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI add_range --target pba --user <user> --password <password> --range <range> --drive <drive> --size <size> --filesystem_type <filesystem_type> [--json]
```

Example

```
.\CigentCLI.exe add_range --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --range 8 --size 0 --json
```

JSON Response

```
{
  "status": {
    "code": 1,
    "message": "No space available"
  }
}
```

remove_range

Protected Storage volumes are created by aligning a protected storage range on the same range occupied by a data partition. This function removes the partition as well as the protected storage range that is aligned with it.

Options

Option	Required	Description
`--password`	Yes	Admin password
`--user`	Yes	Admin user identifier
`--drive`	Yes	Drive or device reference
`--range`	Yes	Range; avoids collision with role (-r)

Option	Required	Description
--target	Yes	Target application
--secrets_stdin	No	Read credentials from STDIN as JSON. See Section 3.
--json	No	Output results in JSON format.

Usage

```
CigentCLI remove_range --target pba --user <user> --password <password> --drive <drive>
--range <range> [--json]
```

Example

```
.\CigentCLI.exe remove_range --target pba --user test_pba_admin --password DemoPBAUser345! --
drive \\.\PHYSICALDRIVE0 --range 8 --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

lock_range

Locking a range will unmount the associated volume (if applicable) and will lock the range at the drive level to prevent read/write access.

Options

Option	Required	Description
--password	Yes	Admin password
--user	Yes	Admin user identifier
--target	Yes	Target application
--drive	Yes	Drive or device reference
--range	Yes	Range; avoids collision with role (-r)
--secrets_stdin	No	Read credentials from STDIN as JSON. See Section 3.
--json	No	Output results in JSON format.

Example

```
.\CigentCLI.exe lock_range --target pba --user test_pba_admin --password DemoPBAUser345! --
drive \\.\PHYSICALDRIVE0 --range 8 --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

Example

unlock_range

Unlocking a range will unlock the range at the drive level to allow read/write access and mount the volume associated with it (if applicable).

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--range`	Yes	Range; avoids collision with role (-r)
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Example

```
.\CigentCLI.exe unlock_range --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --range 8 --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

Example

list_ranges

Returns the current range configuration. The Range number specifies the range. The command will also display if the range is locked or unlocked. Lastly, a count of the number of ranges currently in use and the total available will be displayed.

Options

Option	Required	Description
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--user`	Yes	Admin user identifier
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_ranges --target pba --user <user> --password <password> --drive <drive> [--json]
```

Example

```
.\CigentCLI.exe list_ranges --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response


```
{
  "data": {
    "available": 8,
    "ranges": [
      {
        "shadow": "enabled,not visible"
      },
      {
        "global": "enabled,unlocked,in_use"
      },
      {
        "range_1": "enabled,unlocked,available"
      },
      {
        "range_2": "enabled,unlocked,available"
      },
      {
        "range_3": "enabled,unlocked,available"
      },
      {
        "range_4": "enabled,unlocked,available"
      },
      {
        "range_5": "enabled,unlocked,available"
      },
      {
        "range_6": "enabled,unlocked,available"
      },
      {
        "range_7": "enabled,unlocked,available"
      },
      {
        "range_8": "enabled,locked,available"
      }
    ],
    "total": 8
  },
  "status": {
    "code": 0
  }
}
```


6.7 Authentication Settings

enable

Enable PBA authentication after disabling. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference

Option	Required	Description
`--target`	Yes	Target application
`--include_secondary`	No	The existence of the flag sets the behavior
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI enable --target pba --user <user> --password <password> --drive <drive> [--include_secondary] [--json]
```

Example

```
.\CigentCLI.exe enable --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

disable

Disable PBA authentication, useful for software installations that require multiple reboots. All settings and configuration are preserved while disabled. Re-enabling requires booting from a PBA USB drive of the same version and authenticating as an administrator. Multidrive systems: Enable Include secondary drives option (recommended) to temporarily remove protection from non-primary drives in addition to the primary. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--include_secondary`	No	The existence of the flag sets the behavior
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI disable --target pba --user <user> --password <password> --drive <drive> [--include_secondary] [--json]
```

Example

```
.\CigentCLI.exe disable --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

set_remember_me

Set "remember me" for the pre-boot authentication. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--set_bool`	Yes	Sets true or false
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_remember_me --target pba --user <user> --password <password> --drive
<drive> --set_bool <set_bool> [--json]
```

Example

```
.\CigentCLI.exe set_remember_me --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --set_bool true --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

get_remember_me

Retrieve the "remember me" setting for the pre-boot environment authentication.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application

Option	Required	Description
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_remember_me --target pba --user <user> --password <password> --drive
<drive> [--json]
```

Example

```
.\CigentCLI.exe get_remember_me --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "value": false
  },
 "status": {
   "code": 0
  }
}
```

set_require_mfa

Set "Require MFA" to true or false. If set to true, the user must specify one and only one auth method to use along with the password. Supported second factors are: smartcard, security key (touch), or security key (PIN). If a user is not configured for a second factor when this is enabled, they will be required to enroll one during their next login attempt. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--set_bool`	Yes	(true or false)
`--auth_method`	Yes	
`--seckey_mode`	No	Security key mode, required when auth_method is security key
`--target_password`	No	User password
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_require_mfa --target pba --user <user> --password <password> --drive
<drive> --set_bool <set_bool> --auth_method <auth_method> [--seckey_mode <seckey_mode>]
--target_password <password> [--json]
```

Example

```
.\CigentCLI.exe set_require_mfa --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --target pba --set_bool true --auth_method
smartcard --json
```

JSON Response

```
{
 "status": {
   "code": 2
  }
}
```

get_require_mfa

Retrieve the "Require MFA" setting.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_require_mfa --target pba --user <user> --password <password> --drive
<drive> [--json]
```

Example

```
.\CigentCLI.exe get_require_mfa --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "value": false
  },
 "status": {
   "code": 0
  }
}
```

6.8 Password Policy

set_password_min_length

Set the minimum password length. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--count`	Yes	Count
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_password_min_length --target pba --user <user> --password <password> --drive <drive> --count <count> [--json]
```

Example

```
.\CigentCLI.exe set_password_min_length --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --count 12 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

get_password_min_length

Retrieve the minimum password length.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_password_min_length --target pba --user <user> --password <password> --drive <drive> [--json]
```

Example

```
.\CigentCLI.exe get_password_min_length --target pba --user test_pba_admin --password DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "value": 8
  },
 "status": {
   "code": 0
  }
}
```

set_password_history

Set or read the password history setting, which is how many passwords are remembered so the user must use new passwords. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--count`	Yes	Count
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_password_history --target pba --user <user> --password <password> --drive
<drive> --count <count> [--json]
```

Example

```
.\CigentCLI.exe set_password_history --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --count 5 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

get_password_history

Retrieve the password history setting, which is how many passwords are remembered so the user must use new passwords.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference

Option	Required	Description
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_password_history --target pba --user <user> --password <password> --drive
<drive> [--json]
```

Example

```
.\CigentCLI.exe get_password_history --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "value": 1
  },
 "status": {
   "code": 0
  }
}
```

6.9 Licensing

set_license

Add a new license key to the PBA by providing the license key text directly or passing in a file. The user can only use one of the license options.

Note: While loading a license before the PBA is loaded, any data for user will work. This command is idempotent; calling it with the same value has no adverse effect.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--license_key`	No	
`--license_file`	No	Path to a license file
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI set_license --target pba --user <user> --password <password> --drive <drive>
[--license_key <license_key>] [--license_file <license_file>] [--json]
```

Example

```
.\CigentCLI.exe set_license --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --license_key <text> --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

get_request_code

Generate and return a request code to allow the user to request a license for this device.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI get_request_code --target pba --user <user> --password <password> --drive
<drive> [--json]
```

Example

```
.\CigentCLI.exe get_request_code --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "value": "<text>"
  },
 "status": {
   "code": 0
  }
}
```

6.10 Data Erasure

erase_drive

Erase the entire drive. The following actions are performed: (1) crypto-erase (encryption key is changed), (2) full block-level erase (all blocks are zeroed), and (3) erase verification.

Note: there are potential limitations when running this command in the OS: the crypto-erase will finish but the system could stop working before the block erase is able to complete. If this is a concern, we recommend running an erase from the PBA itself OR run erase-after with an epoch of 1 and reboot.

Enable Include secondary drives option (recommended) to erase non-primary drives in addition to the primary.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--include_secondary`	Yes	The existence of the flag sets the behavior
`--allow_os`	No	No value, required if the target drive is the os drive
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI erase_drive --target pba --user <user> --password <password> --drive <drive>
--include_secondary [--allow_os] [--json]
```

Example

```
.\CigentCLI.exe erase_drive --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --include_secondary --allow_os --json
```

JSON Response

Note: This is a destructive operation. If the target drive is the OS drive, the system will likely crash before a JSON response can be returned to the console.

```
{
  "status": { "code": 0 },
  "data": {
    "erasure_result": "success"
  }
}

NOTE: erasure_result is an EraseVerifyRes enum value
```

erase_after

Erase the entire drive of primaries and secondaries if the PBA boots and the current time is after the specified time. The time is expressed as a Unix epoch value in seconds (UTC). Set the time to 0 to disable.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--erasure_epoch`	Yes	Date/time for erasure expressed as Unix epoch (UTC)
`--target`	Yes	Target application (pba)
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI erase_after --target pba --user <user> --password <password> --drive <drive>
--erasure_epoch <erasure_epoch> [--json]
```

Example

```
.\CigentCLI.exe erase_after --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --erasure_epoch 2208988800 --json
```

JSON Response

```
{
 "status": {
   "code": 0
  }
}
```

6.11 Logs

download_logs

Retrieve the PBA logs from the device.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI download_logs --target pba --user <user> --password <password> --drive
<drive> [--json]
```

Example

```
.\CigentCLI.exe download_logs --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response

```
{
 "data": {
   "logs": [
      {
          "action": 8,
          "source": "CLI",
          "target_user": "",
          "timestamp": 1774990900,
          "user": "test_pba_admin"
        },
      {
          "action": 8,
          "source": "CLI",
```

```
          "target_user": "",
          "timestamp": 1774990881,
          "user": "test_pba_admin"
         },
      {
          "action": 2,
          "source": "PBA",
          "target_user": "",
          "timestamp": 1774969797,
          "user": "jeff"
         },
      {
          "action": 0,
          "source": "PBA",
          "target_user": "",
          "timestamp": 1774969793,
          "user": "jeff"
         }
     ]
  },
 "status": {
   "code": 0
  }
}
```


purge_logs

Purge the PBA logs from the device. An entry will be made in the log.

Options

Option	Required	Description
`--user`	Yes	Admin user identifier
`--password`	Yes	Admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI purge_logs --target pba --user <user> --password <password> --drive <drive>
[--json]
```

Example

```
.\CigentCLI.exe purge_logs --target pba --user test_pba_admin --password
DemoPBAUser345! --drive \\.\PHYSICALDRIVE0 --json
```

JSON Response


```
{
 "status": {
   "code": 0
  }
}
```

7. Self-Encrypting Drive (SED)

Direct management of self-encrypting drive features including initialization, range management, and secure erasure. Available on Windows and Linux. For scripted and RMM environments, use --secrets_stdin instead of --password (see Section 3). All commands require --target sed.

7.1 Status & Information

list_devices

List the USB devices and drives attached to the system.

Options

Option	Required	Description
`--target`	Yes	Target application
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_devices --target sed [--json]
```

Example

```
.\CigentCLI.exe list_devices --target sed --json
```

JSON Response


```
{
  "data": {
    "drives": [
      {
        "brand": "",
        "enclosure_fw": "",
        "fw": "ELFI02.0",
        "is_usb": false,
        "model": "Cigent Secure SSD M.2 2230",
        "path": "\\\\.\\PHYSICALDRIVE0",
        "provisioned": false,
        "security_support": [
          "Opal SSC v2*"
        ],
        "serial": "5S12408020120000017",
        "size_bytes": 512110190080,
        "type": "internal"
      }
    ]
  },
  "status": {
    "code": 0
  }
}
```


7.2 Initialization

init_drive

Initialize the security subsystem on a self-encrypting drive. This must be done before any range or user operations. Sets the SED admin password.

Options

Option	Required	Description
`--password`	Yes	SED admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI init_drive --password <password> --drive <drive> --target sed [--json]
```

Example

```
.\CigentCLI.exe init_drive --password "P@ssw0rd1" --drive \\.\PHYSICALDRIVE0 --target
sed --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

deinit_drive

Deinitialize the security subsystem on a self-encrypting drive. This removes all ranges and user configuration but does not erase data. The drive can be re-initialized afterward.

Options

Option	Required	Description
`--password`	Yes	SED admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI deinit_drive --password <password> --drive <drive> --target sed [--json]
```

Example

```
.\CigentCLI.exe deinit_drive --password "P@ssw0rd1" --drive \\.\PHYSICALDRIVE0 --target
sed --json
```

JSON Response

```
{
  "status": {
    "code": 0
```

```
  }
}
```

7.3 Range Management

add_range

Add a locking range to the drive. Ranges define protected regions of the SSD that can be independently locked and unlocked.

Options

Option	Required	Description
`--password`	Yes	SED admin password
`--range`	Yes	Range number
`--size`	Yes	Size in bytes; 0 for all available
`--filesystem_type`	No	Filesystem type (e.g., ext4). If not specified, the value will default to ext4.
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI add_range --password <password> --range <range> --size <size> --drive <drive> --
target sed [--filesystem_type <type>] [--json]
```

Example

```
.\CigentCLI.exe add_range --password "P@ssw0rd1" --size 0 --filesystem_type ext4 --
range 1 --drive \\.\PHYSICALDRIVE0 --target sed --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

remove_range

Remove a locking range from the drive. Data in the range will no longer be accessible.

Options

Option	Required	Description
`--password`	Yes	SED admin password
`--range`	Yes	Range number
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application

Option	Required	Description
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI remove_range --password <password> --range <range> --drive <drive> --target sed [--json]
```

Example

```
.\CigentCLI.exe remove_range --password "P@ssw0rd1" --range 1 --drive \\.\PHYSICALDRIVE0 --target sed --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

lock_range

Lock a range on the drive. A locked range cannot be read or written until it is unlocked.

Options

Option	Required	Description
`--password`	Yes	SED admin password
`--range`	Yes	Range number
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI lock_range --password <password> --range <range> --drive <drive> --target sed [--json]
```

Example

```
.\CigentCLI.exe lock_range --password "P@ssw0rd1" --range 1 --drive \\.\PHYSICALDRIVE0 --target sed --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

unlock_range

Unlock a range on the drive to allow read/write access.

Options

Option	Required	Description
`--password`	Yes	SED admin password
`--range`	Yes	Range number
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI unlock_range --password <password> --range <range> --drive <drive> --target sed [--json]
```

Example

```
.\CigentCLI.exe unlock_range --password "P@ssw0rd1" --range 1 --drive \\.\PHYSICALDRIVE0 --target sed --json
```

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

list_ranges

List all ranges configured on the drive, including their lock state and availability.

Options

Option	Required	Description
`--password`	Yes	SED admin password
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI list_ranges --password <password> --drive <drive> --target sed [--json]
```

Example

```
.\CigentCLI.exe list_ranges --password "P@ssw0rd1" --drive \\.\PHYSICALDRIVE0 --target sed --json
```

JSON Response

```
{
  "data": {
    "available": 15,
    "ranges": [
      {
```

```
        "global": "disabled,n/a,in_use"
      },
      {
        "range_1": "disabled,n/a,available"
      },
      {
        "range_2": "disabled,n/a,available"
      },
      {
        "range_3": "disabled,n/a,available"
      },
      {
        "range_4": "disabled,n/a,available"
      },
      {
        "range_5": "disabled,n/a,available"
      },
      {
        "range_6": "disabled,n/a,available"
      },
      {
        "range_7": "disabled,n/a,available"
      },
      {
        "range_8": "disabled,n/a,available"
      },
      {
        "range_9": "disabled,n/a,available"
      },
      {
        "range_10": "disabled,n/a,available"
      },
      {
        "range_11": "disabled,n/a,available"
      },
      {
        "range_12": "disabled,n/a,available"
      },
      {
        "range_13": "disabled,n/a,available"
      },
      {
        "range_14": "disabled,n/a,available"
      },
      {
        "range_15": "disabled,n/a,available"
      }
    ],
    "total": 15
  },
  "status": {
    "code": 0
  }
}
```


7.4 Data Erasure & Recovery

erase_drive

Perform a cryptographic erase on the drive. This destroys all data by rotating the encryption key.

Options

Option	Required	Description
`--password`	Yes	SED admin password
`--allow_os`	No	Required flag if the target drive is the OS drive
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--secrets_stdin`	No	Read credentials from STDIN as JSON. See Section 3.
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI erase_drive --password <password> --drive <drive> --target sed [--allow_os] [--json]
```

Example

```
.\CigentCLI.exe erase_drive --password "P@ssw0rd1" --allow_os --drive
\\.\PHYSICALDRIVE0 --target sed --json
```

Note: This is a destructive operation. If the target drive is the OS drive, the system will likely crash before a JSON response can be returned to the caller.

JSON Response

```
{
  "status": {
    "code": 0
  }
}
```

psid_revert

The PSID Revert function returns the SSD to factory state: it erases and removes all ranges, ALL data on the drive, and deinitializes the security subsystem. The PSID is a Physical Security ID printed on the drive label. This operation does not require the SED password.

Options

Option	Required	Description
`--psid`	Yes	Physical Security ID (printed on drive label)
`--drive`	Yes	Drive or device reference
`--target`	Yes	Target application
`--json`	No	Output results in JSON format.

Usage

```
CigentCLI psid_revert --target sed --drive <drive> --psid <psid> [--json]
```

Example

```
.\CigentCLI.exe psid_revert --psid 5S1240802012CJD00120240806000017 --drive
\\.\PHYSICALDRIVE0 --target sed --json
```

Note: This is a destructive, irreversible operation that returns the drive to factory state. All data will be lost.

JSON Response

```
{
  "status": {
```

```
    "code": 0
  }
}
```

8. Appendix

A. Options Reference

Complete list of all CLI options:

Option	Notes	Valid Values
addl_algos	Enable additional smartcard algorithms	N/A
allow_os	Required flag if the target drive is the OS drive	N/A
alt	Alternate mode for PBA installation media creation	N/A
auth_method	Authentication Method	smartcard|security_key (one value only)
autoimport	Auto-import drives during PBA install	N/A
brand	An integer representing the drive brand	Brand values: CIGENT = 1 DIGISTOR = 2 SEAGATE = 3 EVERFOX = 4 KANGURU = 5 VRS = 6 SWISSBIT = 7
code	Enrollment code	Follows password rules
code_id	ID representing an enrollment code	Provided by the system
count	Count	Integer, depending on command
drive	Drive or device reference	The serial number of the drive or the identifier provided by the list_drives command.
email_address	Email for user creation or notifications	Note: the email address is not validated for sensibility or format
enable	Enable PBA during install	N/A
erasure_date	Date/time for erasure	Epoch Time, integer
erasure_epoch	Date/time for erasure expressed as Unix epoch (UTC)	Epoch Time, integer.
expiration	Expiration	Epoch Time, integer.
file	File path for PBA update binary	Valid file path
filesystem_type	Filesystem type	ntfs|ext|none

Option	Notes	Valid Values
include_secondary	The existence of the flag sets the behavior	N/A
json	If this is selected, output JSON	N/A
license_file	Path to a license file	Valid file path
license_key		The license key
min_password_length	Minimum password length	integer 1-128
no_verify	Skip verification during PBA install	N/A
open	Open mode for PBA installation media/install	N/A
password	Admin password	See Section C below
password_history_count	Password history count	Integer, 1-20
pba_bin_file	Path to the PBA binary file for install or media creation	Valid file path
psid	PSID	Valid serial id
raid	RAID mode for PBA installation media/install	N/A
range	Range; avoids collision with role (-r)	Integer, 1-8
require_pw_change	Require the user to change password on next login	true|false
role	Role identifier	user|admin (more to come)
seckey_mode	Security key mode, required when auth_method is security key	3|4
secrets_stdin	When specified, STDIN is read for passwords. See Section 3 for more details.	N/A
set_bool	Sets true or false	true|false
size	Size parameter	Integer
target	Target application	sed|fde|pba
target_password	User password	Must follow password rules
target_user	Username	Must follow username rules
user	Admin user identifier	Must follow username rules

B. Username Requirements

Requirement	Username
Length	1-40 characters
Uppercase letter: A-Z	May contain
Lowercase letter: a-z	May contain
Number: 0-9	May contain

Requirement	Username
Special: ~! @#$^&*()_-+=[]:<>.'/	May contain

C. Password Requirements

Requirement	Password
Length	8-128 characters (default minimum is 15 characters for FDE, 8 characters for PBA)
Uppercase letter: A-Z	Must contain at least 1
Lowercase letter: a-z	Must contain at least 1
Number: 0-9	Must contain at least 1
Special: ~!@#$^&*()_-+=[]:<>.	Must contain at least 1

D. Device State Values (PBA)

The state field returned by list_devices is a numeric bitmask. A bitmask value can represent one state or a combination of states. To interpret the value, add together the state values shown in the table below.

For example, a returned state value of 7 means:

1 + 2 + 4 = inited + disabled + additional

A returned state value of 23 means:

1 + 2 + 4 + 16 = inited + disabled + additional + preload

State Value	Label	Description
0	none	No recognized device state is set.
1	inited	The drive has been initialized for use with PBA.
2	disabled	PBA protection is configured but currently disabled.
4	secondary	The drive is configured as an secondary, non-primary protected drive.
8	importable	The drive appears to be an additional drive from another PBA configuration and may be imported, but credentials are not currently available.
16	preload	The drive is in a preloaded PBA state.
32	open preload	The drive is in an open preloaded PBA state.

State Value	Label	Description
64	installable	The drive is eligible for PBA installation.
128	installer	The drive contains PBA installation media or is acting as an installer device.
256	installable nonopal	The drive is eligible for installation but does not support Opal.
512	open primary	The drive is configured as an open primary PBA drive.
1024	open has db	The drive is in open mode and contains a PBA database.
2048	open disabled	The drive is in open mode and PBA protection is currently disabled.