code42cli

Release code42cli v1.16.6

Code42 Software

Apr 12, 2023

CONTENTS

CHAPTER

ONE

USER GUIDES

1.1 Get started with the Code42 command-line interface (CLI)

1.1.1 Licensing

This project uses the MIT License.

1.1.2 Installation

You can install the Code42 CLI from PyPI, from source, or from distribution.

From PyPI

The easiest and most common way is to use `pip`:

```
python3 -m pip install code42cli
```

To install a previous version of the Code42 CLI via `pip`, add the version number. For example, to install version 0.5.3, enter:

```
python3 -m pip install code42cli==0.5.3
```

Visit the project history on PyPI to see all published versions.

From source

Alternatively, you can install the Code42 CLI directly from source code:

```
git clone https://github.com/code42/code42cli.git
```

When it finishes downloading, from the root project directory, run:

```
python setup.py install
```

From distribution

If you want create a `.tar` ball for installing elsewhere, run the following command from the project's root directory:

```
python setup.py sdist
```

After it finishes building, the `.tar` ball will be located in the newly created `dist` directory. To install it, enter:

```
python3 -m pip install code42cli-[VERSION].tar.gz
```

1.1.3 Updates

To update the CLI, use the pip `--upgrade` flag.

```
python3 -m pip install code42cli --upgrade
```

1.1.4 Authentication

Important: The Code42 CLI currently only supports token-based authentication.

Create a user in Code42 to authenticate (basic authentication) and access data via the CLI. The CLI returns data based on the roles assigned to this user. To ensure that the user's rights are not too permissive, create a user with the lowest level of privilege necessary. See our Role assignment use cases for information on recommended roles. We recommend you test to confirm that the user can access the right data.

If you choose not to store your password in the CLI, you must enter it for each command that requires a connection.

The Code42 CLI supports local accounts with MFA (multi-factor authentication) enabled. The Time-based One-Time Password (TOTP) must be provided at every invocation of the CLI, either via the `--totp` option or when prompted.

The Code42 CLI currently does **not** support SSO login providers or any other identity providers such as Active Directory or Okta.

1.1.5 Proxy Support

Note: Proxy support was added in code42cli version 1.16.0

The Code42 CLI will attempt to connect through a proxy if the `https_proxy`/`HTTPS_PROXY` environment variable is set.

Windows and Mac

For Windows and Mac systems, the CLI uses Keyring when storing passwords.

Red Hat Enterprise Linux

To use Keyring to store the credentials you 2enter in the Code42 CLI, enter the following commands before installing.

```
yum -y install python-pip python3 dbus-python gnome-keyring libsecret dbus-x11
pip3 install code42cli
```

If the following directories do not already exist, create them:

```
mkdir -p ~/.cache
mkdir -p ~/.local/share/keyring
```

In the following commands, replace the example value `\n` with the Keyring password (if the default Keyring already exists).

```
eval "$(dbus-launch --sh-syntax)"
eval "$(printf '\n' | gnome-keyring-daemon --unlock)"
eval "$(printf '\n' | /usr/bin/gnome-keyring-daemon --start)"
```

Close out your D-bus session and GNOME Keyring:

```
pkill gnome
pkill dbus
```

If you do not use Keyring to store your credentials, the Code42 CLI will ask permission to store your credentials in a local flat file with read/write permissions for only the operating system user who set the password. Alternatively, you can enter your password with each command you enter.

Ubuntu

If Keyring doesn't support your Ubuntu system, the Code42 CLI will ask permission to store your credentials in a local flat file with read/write permissions for only the operating system user who set the password. Alternatively, you can enter your password with each command you enter.

To learn more about authenticating in the CLI, follow the *Configure profile guide*.

1.1.6 Troubleshooting and support

Code42 command not found

If your python installation has added itself to your environment's PATH variable, then running `code42` *should* just work.

However, if after installation the `code42` command is not found, the CLI has some helpers for this (added in version 1.10):

You can execute the CLI by calling the python module directly:

```
python3 -m code42cli
```

And the base `code42` command now has a `--script-dir` option that will print out the directory the `code42` script was installed into, so you can manually add it to your PATH, enabling the `code42` command to work.

On Mac/Linux:

Run the following to make `code42` visible in your shell's PATH (to persist the change, add it to your shell's configuration file):

```
export PATH=$PATH:$(python3 -m code42cli --script-dir)
```

On Windows:

```
$env:Path += ";$(python -m code42cli --script-dir)"
```

To persist the change, add the updated PATH to your registry:

```
Set-ItemProperty -Path 'Registry::HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\
↪Session Manager\Environment' -Name PATH -Value $env:Path
```

Debug mode

Debug mode may be useful if you are trying to determine if you are experiencing permissions issues. When debug mode is on, the CLI logs HTTP request data to the console. Use the `-d` flag to enable debug mode for a particular command. `-d` can appear anywhere in the command chain:

```
code42 <command> <subcommand> <args> -d
```

File an issue on GitHub

If you are experiencing an issue with the Code42 CLI, select *New issue* at the project repository to create an issue. See the Github guide on creating an issue for more information.

Contact Code42 Support

If you don't have a GitHub account and are experiencing issues, contact Code42 support.

1.1.7 What's next?

Learn how to *Set up a profile*.

1.2 Configure profile

Use the *code42 profile* set of commands to establish the Code42 environment you're working within and your user information.

1.2.1 User token authentication

Use the following command to create your profile with user token authentication:

```
code42 profile create --name MY_FIRST_PROFILE --server example.authority.com --username␣
↪security.admin@example.com
```

Your profile contains the necessary properties for authenticating with Code42. After running `code42 profile create`, the program prompts you about storing a password. If you agree, you are then prompted to enter your password.

Your password is not shown when you do `code42 profile show`. However, `code42 profile show` will confirm that a password exists for your profile. If you do not set a password, you will be securely prompted to enter a password each time you run a command.

1.2.2 API client authentication

Once you've generated an API Client in your Code42 console, use the following command to create your profile with API client authentication:

```
code42 profile create-api-client --name MY_API_CLIENT_PROFILE --server example.authority.
↪com --api-client-id 'key-42' --secret 'code42%api%client%secret'
```

Note: Remember to wrap your API client secret with single quotes to avoid issues with bash expansion and special characters.

1.2.3 View profiles

You can add multiple profiles with different names and the change the default profile with the `use` command:

```
code42 profile use MY_SECOND_PROFILE
```

When you use the `--profile` flag with other commands, such as those in `security-data`, that profile is used instead of the default profile. For example,

```
code42 security-data search -b 2020-02-02 --profile MY_SECOND_PROFILE
```

To see all your profiles, do:

```
code42 profile list
```

1.2.4 Profiles with Multi-Factor Authentication

If your Code42 user account requires multi-factor authentication, the MFA token can either be passed in with the `--totp` option, or if not passed you will be prompted to enter it before the command executes.

1.3 V2 File Events

> **Warning:** V1 file events, saved searches, and queries are **deprecated**.

For details on the updated File Event Model, see the V2 File Events API documentation on the Developer Portal.

V1 file event APIs were marked deprecated in May 2022 and will be no longer be supported after May 2023.

Use the `--use-v2-file-events True` option with the `code42 profile create` or `code42 profile update` commands to enable your code42 CLI profile to use the latest V2 file event data model.

Use `code42 profile show` to check the status of this setting on your profile:

```
% code42 profile update --use-v2-file-events True

% code42 profile show

test-user-profile:
        * username = test-user@code42.com
        * authority url = https://console.core-int.cloud.code42.com
        * ignore-ssl-errors = False
        * use-v2-file-events = True
```

For details on setting up a profile, see the *profile set up user guide*.

Enabling this setting will use the V2 data model for querying searches and saved searches with all `code security-data` commands. The response shape for these events has changed from V1 and contains various field remappings, renamings, additions and removals. Column names will also be different when using the `Table` format for outputting events.

1.3.1 V2 File Event Data Example

Below is an example of the new file event data model:


```
{
    "@timestamp": "2022-07-14T16:53:06.112Z",
    "event": {
        "id": "0_c4e43418-07d9-4a9f-a138-29f39a124d33_1068825680073059134_
↪1068826271084047166_1_EPS",
        "inserted": "2022-07-14T16:57:00.913917Z",
        "action": "application-read",
        "observer": "Endpoint",
        "shareType": [],
        "ingested": "2022-07-14T16:55:04.723Z",
        "relatedEvents": []
    },
    "user": {
        "email": "engineer@example.com",
        "id": "1068824450489230065",
        "deviceUid": "1068825680073059134"
    },
    "file": {
        "name": "cat.jpg",
        "directory": "C:/Users/John Doe/Downloads/",
        "category": "Spreadsheet",
        "mimeTypeByBytes": "application/vnd.openxmlformats-officedocument.spreadsheetml.
↪sheet",
        "categoryByBytes": "Spreadsheet",
        "mimeTypeByExtension": "image/jpeg",
        "categoryByExtension": "Image",
        "sizeInBytes": 4748,
        "owner": "John Doe",
        "created": "2022-07-14T16:51:06.186Z",
        "modified": "2022-07-14T16:51:07.419Z",
        "hash": {
            "md5": "8872dfa1c181b823d2c00675ae5926fd",
            "sha256": "14d749cce008711b4ad1381d843745395603406 22f0e8b9eb2fe3bba77ddbd64",
            "md5Error": null,
            "sha256Error": null
        },
        "id": null,
        "url": null,
        "directoryId": [],
        "cloudDriveId": null,
        "classifications": []
    },
    "report": {
        "id": null,
        "name": null,
        "description": null,
        "headers": [],
        "count": null,
        "type": null
```


(continues on next page)

(continued from previous page)


```
    },
    "source": {
        "category": "Device",
        "name": "DESKTOP-1",
        "domain": "192.168.00.000",
        "ip": "50.237.00.00",
        "privateIp": [
            "192.168.00.000",
            "127.0.0.1"
        ],
        "operatingSystem": "Windows 10",
        "email": {
            "sender": null,
            "from": null
        },
        "removableMedia": {
            "vendor": null,
            "name": null,
            "serialNumber": null,
            "capacity": null,
            "busType": null,
            "mediaName": null,
            "volumeName": [],
            "partitionId": []
        },
        "tabs": [],
        "domains": []
    },
    "destination": {
        "category": "Cloud Storage",
        "name": "Dropbox",
        "user": {
            "email": []
        },
        "ip": null,
        "privateIp": [],
        "operatingSystem": null,
        "printJobName": null,
        "printerName": null,
        "printedFilesBackupPath": null,
        "removableMedia": {
            "vendor": null,
            "name": null,
            "serialNumber": null,
            "capacity": null,
            "busType": null,
            "mediaName": null,
            "volumeName": [],
            "partitionId": []
        },
        "email": {
            "recipients": null,
```


(continues on next page)

(continued from previous page)


```
            "subject": null
        },
        "tabs": [
            {
                "title": "Files - Dropbox and 1 more page - Profile 1 - Microsoft Edge",
                "url": "https://www.dropbox.com/home",
                "titleError": null,
                "urlError": null
            }
        ],
        "accountName": null,
        "accountType": null,
        "domains": [
            "dropbox.com"
        ]
    },
    "process": {
        "executable": "C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe
↪",
        "owner": "John doe"
    },
    "risk": {
        "score": 17,
        "severity": "CRITICAL",
        "indicators": [
            {
                "name": "First use of destination",
                "weight": 3
            },
            {
                "name": "File mismatch",
                "weight": 9
            },
            {
                "name": "Spreadsheet",
                "weight": 0
            },
            {
                "name": "Remote",
                "weight": 0
            },
            {
                "name": "Dropbox upload",
                "weight": 5
            }
        ],
        "trusted": false,
        "trustReason": null
    }
}
```

1.4 Ingest file event data or alerts into a SIEM tool

This guide provides instructions on using the CLI to ingest Code42 file event data or alerts into a security information and event management (SIEM) tool like LogRhythm, Sumo Logic, or IBM QRadar.

1.4.1 Considerations

To ingest file events or alerts into a SIEM tool using the Code42 command-line interface, the Code42 user account running the integration must be assigned roles that provide the necessary permissions.

The CEF format is not recommended because it was not designed for insider risk event data. Code42 file event data contains many fields that provide valuable insider risk context that have no CEF equivalent. However, if you need to use CEF, the JSON-to-CEF mapping at the bottom of this document indicates which fields are included and how the field names map to other formats.

1.4.2 Before you begin

First install and configure the Code42 CLI following the instructions in *Getting Started*.

1.4.3 Run queries

You can get file events in either a JSON or CEF format for use by your SIEM tool. Alerts data and audit logs are available in JSON format. You can query the data as a scheduled job or run ad-hoc queries.

Learn more about searching *File Events*, *Alerts*, and *Audit Logs* using the CLI.

Run a query as a scheduled job

Use your favorite scheduling tool, such as cron or Windows Task Scheduler, to run a query on a regular basis. Specify the profile to use by including `--profile`.

File Exposure Events

An example using the `send-to` command to forward only the new file event data since the previous request to an external syslog server:

```
code42 security-data send-to syslog.example.com:514 -p UDP --profile profile1 -c syslog_
↪sender
```

Alerts

An example to send to the syslog server only the new alerts that meet the filter criteria since the previous request:

```
code42 alerts send-to syslog.example.com:514 -p UDP --profile profile1 --rule-name
↪"Source code exfiltration" --state OPEN -i
```

Audit Logs

An example to send to the syslog server only the audit log events that meet the filter criteria from the last 30 days.

```
code42 audit-logs send-to syslog.example.com:514 -p UDP --profile profile1 --actor-
↪username 'sean.cassidy@example.com' -b 30d
```

As a best practice, use a separate profile when executing a scheduled task. Using separate profiles can help prevent accidental updates to your stored checkpoints, for example, by adding `--use-checkpoint` to adhoc queries.

Run an ad-hoc query

Examples of ad-hoc queries you can run are as follows.

File Exposure Events

Print file events since March 5 for a user in raw JSON format:

```
code42 security-data search -f RAW-JSON -b 2020-03-05 --c42-username 'sean.
↪cassidy@example.com'
```

Print file events since March 5 where a file was synced to a cloud service:

```
code42 security-data search -t  CloudStorage -b 2020-03-05
```

Write to a text file the file events in raw JSON format where a file was read by browser or other app for a user since March 5:

```
code42 security-data search -f RAW-JSON -b 2020-03-05 -t ApplicationRead --c42-username
↪'sean.cassidy@example.com' > /Users/sangita.maskey/Downloads/c42cli_output.txt
```

Alerts

Print alerts since May 5 where a file's cloud share permissions changed:

```
code42 alerts print -b 2020-05-05 --rule-type FedCloudSharePermissions
```

Audit Logs

Print audit log events since June 5 which affected a certain user:

```
code42 audit-logs search -b 2021-06-05 --affected-username 'sean.cassidy@examply.com'
```

Example Outputs

Example output for a single file exposure event (in default JSON format):


```
{
    "eventId": "0_c4b5e830-824a-40a3-a6d9-345664cfbb33_942704829036142720_
↪944009394534374185_342",
    "eventType": "CREATED",
    "eventTimestamp": "2020-03-05T14:45:49.662Z",
    "insertionTimestamp": "2020-03-05T15:10:47.930Z",
    "filePath": "C:/Users/sean.cassidy/Google Drive/",
    "fileName": "1582938269_Longfellow_Cloud_Arch_Redesign.drawio",
    "fileType": "FILE",
    "fileCategory": "DOCUMENT",
    "fileSize": 6025,
    "fileOwner": "Administrators",
    "md5Checksum": "9ab754c9133afbf2f70d5fe64cde1110",
    "sha256Checksum": "8c6ba142065373ae5277ecf9f0f68ab8f9360f42a82eb1dec2e1816d93d6b1b7",
    "createTimestamp": "2020-03-05T14:29:33.455Z",
    "modifyTimestamp": "2020-02-29T01:04:31Z",
    "deviceUserName": "sean.cassidy@example.com",
    "osHostName": "LAPTOP-091",
    "domainName": "192.168.65.129",
    "publicIpAddress": "71.34.10.80",
    "privateIpAddresses": [
        "fe80:0:0:0:8d61:ec3f:9e32:2efc%eth2",
        "192.168.65.129",
        "0:0:0:0:0:0:0:1",
        "127.0.0.1"
    ],
    "deviceUid": "942704829036142720",
    "userUid": "887050325252344565",
    "source": "Endpoint",
    "exposure": [
        "CloudStorage"
    ],
    "syncDestination": "GoogleBackupAndSync"
}
```


Example output for a single alert (in default JSON format):


```
{
    "type$": "ALERT_DETAILS",
    "tenantId": "c4b5e830-824a-40a3-a6d9-345664cfbb33",
    "type": "FED_CLOUD_SHARE_PERMISSIONS",
    "name": "Cloud Share",
    "description": "Alert Rule for data exfiltration via Cloud Share",
    "actor": "leland.stewart@example.com",
    "target": "N/A",
    "severity": "HIGH",
    "ruleId": "408eb1ae-587e-421a-9444-f75d5399eacb",
    "ruleSource": "Alerting",
    "id": "7d936d0d-e783-4b24-817d-f19f625e0965",
```


(continues on next page)

(continued from previous page)


```
    "createdAt": "2020-05-22T09:47:33.8863230Z",
    "state": "OPEN",
    "observations": [{"type$": "OBSERVATION",
        "id": "4bc378e6-bfbd-40f0-9572-6ed605ea9f6c",
        "observedAt": "2020-05-22T09:40:00.0000000Z",
        "type": "FedCloudSharePermissions",
        "data": {
            "type$": "OBSERVED_CLOUD_SHARE_ACTIVITY",
            "id": "4bc378e6-bfbd-40f0-9572-6ed605ea9f6c",
            "sources": ["GoogleDrive"],
            "exposureTypes": ["PublicLinkShare"],
            "firstActivityAt": "2020-05-22T09:40:00.0000000Z",
            "lastActivityAt": "2020-05-22T09:45:00.0000000Z",
            "fileCount": 1,
            "totalFileSize": 6025,
            "fileCategories": [{"type$": "OBSERVED_FILE_CATEGORY", "category": "Document
↪", "fileCount": 1, "totalFileSize": 6025, "isSignificant": false}],
            "files": [{"type$": "OBSERVED_FILE", "eventId": "1hHdK6Qe6hez4vNCtS-UimDf-
↪sbaFd-D7_3_baac33d0-a1d3-4e0a-9957-25632819eda7", "name": "1590140395_Longfellow_Cloud_
↪Arch_Redesign.drawio", "category": "Document", "size": 6025}],
            "outsideTrustedDomainsEmailsCount": 0, "outsideTrustedDomainsTotalDomainCount
↪": 0, "outsideTrustedDomainsTotalDomainCountTruncated": false}}]
}
```


Example output for a single audit log event (in default JSON format):


```
{
    "type$": "audit_log::logged_in/1",
    "actorId": "1015070955620029617",
    "actorName": "sean.cassidy@example.com",
    "actorAgent": "py42 1.17.0 python 3.7.10",
    "actorIpAddress": "67.220.16.122",
    "timestamp": "2021-08-30T16:16:19.165Z",
    "actorType": "USER"
}
```


1.4.4 CEF Mapping

The following tables map the file event data from the Code42 CLI to common event format (CEF).

Attribute mapping

The table below maps JSON fields, CEF fields, and Forensic Search fields to one another.

JSON field	CEF field	Forensic Search field
actor	suser	Actor
cloudDriveId	aid	n/a
createTimestamp	fileCreateTime	File Created Date
deviceUid	deviceExternalId	n/a

continues on next page

Table 1 – continued from previous page

JSON field	CEF field	Forensic Search field
deviceUserName	suser	Username (Code42)
domainName	dvchost	Fully Qualified Domain Name
eventId	externalID	n/a
eventTimestamp	end	Date Observed
exposure	reason	Exposure Type
fileCategory	fileType	File Category
fileName	fname	Filename
filePath	filePath	File Path
fileSize	fsize	File Size
insertionTimestamp	rt	n/a
md5Checksum	fileHash	MD5 Hash
modifyTimestamp	fileModificationTime	File Modified Date
osHostName	shost	Hostname
processName	sproc	Executable Name (Browser or Other App)
processOwner	spriv	Process User (Browser or Other App)
publicIpAddress	src	IP Address (public)
removableMediaBusType	cs1, Code42AEDRemovableMediaBusType	Device Bus Type (Removable Media)
removableMediaCapacity	cn1, Code42AEDRemovableMediaCapacity	Device Capacity (Removable Media)
removableMediaName	cs3, Code42AEDRemovableMediaName	Device Media Name (Removable Media)
removableMediaSerialNumber	cs4	Device Serial Number (Removable Media)
removableMediaVendor	cs2, Code42AEDRemovableMediaVendor	Device Vendor (Removable Media)
sharedWith	duser	Shared With
syncDestination	destinationServiceName	Sync Destination (Cloud)
url	filePath	URL
userUid	suid	n/a
windowTitle	requestClientApplication	Tab/Window Title
tabUrl	request	Tab URL
emailSender	suser	Sender
emailRecipients	duser	Recipients

Event mapping

See the table below to map file events to CEF signature IDs.

Exfiltration event	CEF field
CREATED	C42200
MODIFIED	C42201
DELETED	C42202
READ_BY_APP	C42203
EMAILED	C42204

1.5 Manage legal hold custodians

Once you create a legal hold matter in the Code42 console, you can use the Code42 CLI to add or release custodians from the matter.

To see a list of all the users currently in your organization:

- Export a list from the Users action menu.
- Use the *CLI users commands*.

Use the `legal-hold` commands to manage legal hold custodians.

- To view a list of legal hold matters for your organization, including the matter ID, use the following command: `code42 legal-hold list`
- To see a list of all the custodians currently associated with a legal hold matter, enter `code42 legal-hold show <matterID>`.

1.5.1 Get CSV template

To add multiple custodians to a legal hold matter:

1. Generate a CSV template. Below is an example command that generates a template to use when bulk adding custodians to legal hold matter. Once generated, the CSV file is saved to your current working directory. `code42 legal-hold bulk generate-template add`

   To generate a template to use when bulk releasing custodians from a legal hold matter:

   `code42 legal-hold bulk generate-template remove`

   The CSV templates for `add` and `remove` have the same columns, but the commands generate different default filenames.
2. Use the CSV template to enter the matter ID(s) and Code42 usernames for the custodians you want to add to the matters. To get the ID for a matter, enter `code42 legal-hold list`.
3. Save the CSV file.

1.5.2 Add custodians to a legal hold matter

You can add one or more custodians to a legal hold matter using the Code42 CLI.

Add multiple custodians

Once you have entered the matter ID and user information in the CSV file, use the `bulk add` command with the CSV file path to add multiple custodians at once. For example:

```
code42 legal-hold bulk add /Users/admin/add_users_to_legal_hold.csv
```

Add a single custodian

To add a single custodian to a legal hold matter, use the following command as an example:

```
code42 legal-hold add-user --matter-id 123456789123456789 --username user@example.com
```

Options

- `--matter-id` (required): The identification number of the legal hold matter. To get the ID for a matter, run the command `code42 legal-hold list`.
- `--username` (required): The Code42 username of the custodian to add to the matter.
- `--profile` (optional): The profile to use to execute the command. If not specified, the default profile is used.

1.5.3 Release custodians

You can release one or more custodians from a legal hold matter using the Code42 CLI.

Release multiple custodians

To release multiple custodians at once:

1. Enter the matter ID(s) and Code42 usernames to the *CSV file template you generated*.
2. Save the file to your current working directory.
3. Use the `bulk remove` command with the file path of the CSV you created. For example: `code42 legal-hold bulk remove /Users/admin/remove_users_from_legal_hold.csv`

Release a single custodian

Use `remove-user` to release a single custodian. For example:

```
code42 legal-hold remove-user --matter-id 123456789123456789 --username user@example.com
```

Options are the same as `add-user` shown above.

1.5.4 View matters and custodians

You can use the Code42 CLI to get a list of all the legal hold matters for your organization, or get full details for a matter.

List legal hold matters

To view a list of legal hold matters for your organization, use the following command:

```
code42 legal-hold list
```

This command produces the matter ID, name, description, creator, and creation date for the legal hold matters.

View matter details

To view active custodians for a legal hold matter, enter `code42 legal-hold show` with the matter ID, for example:

```
code42 legal-hold show 123456789123456789
```

To view active custodians for a legal hold matter, as well as the details of the preservation policy, enter

```
code42 legal-hold show <matterID> --include-policy
```

To view all custodians (including inactive) for a legal hold matter, enter

```
code42 legal-hold show <matterID> --include-inactive
```

List legal hold events

To view a list of legal hold administrative events, use the following command:

```
code42 legal-hold search-events
```

This command takes the optional filters of a specific matter uid, beginning timestamp, end timestamp, and event type.

Learn more about the *Legal Hold* commands.

1.6 Clean up your environment by deactivating devices

Your Code42 environment may contain many old devices that are no longer active computers and that have not connected to Code42 in quite some time. In order to clean up your environment, you can use the CLI to deactivate these devices in bulk.

1.6.1 Generate a list of devices

You can generate a list of devices using `code42 devices list`. By default, it will display the list of devices at the command line, but you can also output it in a number of file formats. For example, to generate a CSV of active devices in your environment, use this command:

```
code42 devices list --active -f CSV
```

To save to a file, redirect the output to a file in your shell:

```
code42 devices list --active -f CSV > output.csv
```

Filter the list

You can filter or edit the list of devices in your spreadsheet or text editor of choice, but the CLI has some parameters built in that can help you to filter the list of devices to just the ones you want to deactivate. To see a full list of available parameters, run `code42 devices list -h`.

Here are some useful parameters you may wish to leverage when curating a list of devices to deactivate:

- `--last-connected-before DATE|TIMESTAMP|SHORT_TIME` - allows you to only see devices that have not connected since a particular date. You can also use a timestamp or short time format, for example `30d`.
- `--exclude-most-recently-connected INTEGER` - allows you to exclude the most recently connected device (per user) from the results. This allows you to ensure that every user is left with at least N device(s), regardless of how recently they have connected.

- `--created-before DATE|TIMESTAMP|SHORT_TIME` - allows you to only see devices created before a particular date.

1.6.2 Deactivate devices

Once you have a list of devices that you want to remove, you can run the `code42 devices bulk deactivate` command:

```
code42 devices bulk deactivate list_of_devices.csv
```

The device list must be a file in CSV format containing a `guid` column with the unique identifier of the devices to be deactivated. The deactivate command can also accept some optional parameters:

- `--change-device-name` - prepends `deactivated_<current_date>` to the beginning of the device name, allowing you to have a record of which devices were deactivated by the CLI and when.
- `--purge-date yyyy-MM-dd` - allows you to change the date on which the deactivated devices' archives will be purged from cold storage.

To see a full list of available options, run `code42 devices bulk deactivate -h`.

The `code42 devices bulk deactivate` command will output the guid of the device to be deactivated, plus a column indicating the success or failure of the deactivation. To change the format of this output, use the `-f` or `--format` option.

You can also redirect the output to a file, for example:

```
code42 devices bulk deactivate devices_to_deactivate.csv -f CSV > deactivation_results.
↪csv
```

Deactivation will fail if the user running the command does not have permission to deactivate the device, or if the user owning the device is on legal hold.

Generate the list and deactivate in a single command

You can also pipe the output of `code42 devices list` directly to `code42 devices bulk deactivate`. When using a pipe, make sure to use `-` as the input argument for `code42 devices bulk deactivate` to indicate that it should read from standard input.

Here is an example:

```
code42 devices list --active \
--last-connected-before 365d \
--exclude-most-recently-connected 1 \
-f CSV \
| code42 devices bulk deactivate - \
-f CSV \
> deactivation_results.csv
```

This lists all devices that have not connected within a year *and* are not a user's most-recently-connected device, and then attempts to deactivate them.

Learn more about *Managing Devices*.

1.7 Write custom extension scripts using the Code42 CLI and py42

While the Code42 CLI aims to provide an easy way to automate many common Code42 tasks, there will likely be times when you need to script something the CLI doesn't have out-of-the-box.

To accommodate for those scenarios, the Code42 CLI exposes a few helper objects in the `code42cli.extensions` module that make it easy to write custom scripts with `py42` that use features of the CLI (like profiles) to reduce the amount of boilerplate needed to be productive.

1.7.1 Before you begin

The Code42 CLI is a python application written using the click framework, and the exposed extension objects are custom `click` classes. A basic knowledge of how to define `click` commands, arguments, and options is required.

The `sdk_options` decorator

The most important extension object is the `sdk_options` decorator. When you decorate a command you've defined in your script with `@sdk_options`, it will automatically add `--profile` and `--debug` options to your command. These work the same as in the main CLI commands.

Decorating a command with `@sdk_options` also causes the first argument to your command function to be the `state` object, which contains the initialized py42 sdk. There's no need to handle user credentials or login, the `sdk_options` does all that for you using the CLI profiles.

The `script` group

The `script` object exposed in the extensions module is a `click.Group` subclass, which allows you to add multiple sub-commands and group functionality together. While not explicitly required when writing custom scripts, the `script` group has logic to help handle and log any uncaught exceptions to the `~/.code42cli/log/code42_errors.log` file.

If only a single command is added to the `script` group, the group will default to that command, so you don't need to explicitly provide the sub-command name.

An example command that just prints the username and ID that the sdk is authenticated with:

```
import click
from code42cli.extensions import script, sdk_options

@click.command()
@sdk_options
def my_command(state):
    user = state.sdk.users.get_current()
    print(user["username"], user["userId"])

if __name__ == "__main__":
    script.add_command(my_command)
    script()
```

1.7.2 Ensuring your script runs in the Code42 CLI python environment

The above example works as a standalone script, if it were named `my_script.py` you could execute it by running:

```
python3 my_script.py
```

However, if the Code42 CLI is installed in a different python environment than your `python3` command, it might fail to import the extensions.

To workaround environment and path issues, the CLI has a `--python` option that prints out the path to the python executable the CLI uses, so you can execute your script with`$(code42 --python) script.py` on Mac/Linux or `&$(code42 --python) script.py` on Windows to ensure it always uses the correct python path for the extension script to work.

1.7.3 Installing your extension script as a Code42 CLI plugin

The above example works as a standalone script, but it's also possible to install that same script as a plugin into the main CLI itself.

Assuming the above example code is in a file called `my_script.py`, just add a file `setup.py` in the same directory with the following:

```
from distutils.core import setup

setup(
    name="my_script",
    version="0.1",
    py_modules=["my_script"],
    install_requires=["code42cli"],
    entry_points="""
        [code42cli.plugins]
        my_command=my_script:my_command
    """,
)
```

The `entry_points` section tells the Code42 CLI where to look for the commands to add to its main group. If you have multiple commands defined in your script you can add one per line in the `entry_points` and they'll all get installed into the Code42 CLI.

Once your `setup.py` is ready, install it with pip while in the directory of `setup.py`:

```
$(code42 --python) -m pip install .
```

Then running `code42 -h` should show `my-command` as one of the available commands to run!

1.8 Manage Users

You can use the CLI to manage user information, update user roles, and move users between organizations.

To view a all the users currently in your organization, you can export a list from the Users list in the Code42 console or you can use the `list` command.

You can use optional flags to filter the users you want to view. The following command will print all active users with the `Desktop User` role who belong to the organization with UID `1234567890`:

```
code42 users list --org-uid 1234567890 --role-name "Desktop User" --active
```

To change the information for one or more users, provide the user UID and updated information with the `update` or `bulk update` commands.

1.8.1 Manage User Roles

Apply Code42's user roles to user accounts to provide administrators with the desired set of permissions. Each role has associated permissions, limitations, and recommended use cases.

View User Roles

View a user's current roles and other details with the `show` command:

```
code42 users show "sean.cassidy@example.com"
```

Alternatively, pass the `--include-roles` flag to the `list` command. The following command will print a list of all active users and their current roles:

```
code42 users list --active --include-roles
```

Update User Roles

Use the following command to add a role to a user:

```
code42 users add-role --username "sean.cassidy@example.com" --role-name "Desktop User"
```

Similarly, use the `remove-role` command to remove a role from a user.

1.8.2 Manage User Risk Profile info

To set a start or end/departure date on a User's profile (useful for users on the "New Hire" and "Departing" Watchlists):

```
code42 users update-start-date 2020-03-10 user@example.com

code42 users update-departure-date 2022-06-20 user@example.com
```

To clear the value of start_date/end_date on a User's profile, use the `--clear` option to the above commands:

```
code42 users update-departure-date --clear user@example.com
```

To update a User's Risk Profile notes field:

```
code42 users update-risk-profile-notes user@example.com "New note text"
```

By default, the note text will overwrite notes are already on the profile. To keep existing note data, use the `--append` option:

```
code42 users update-risk-profile-notes user@example.com "Additional note text" --append
```

1.8.3 Deactivate a User

You can deactivate a user with the following command:

```
code42 users deactivate sean.cassidy@example.com
```

To deactivate multiple users at once, enter each username on a new line in a CSV file, then use the `bulk deactivate` command with the CSV file path. For example:

```
code42 users bulk deactivate users_to_deactivate.csv
```

Similarly, use the `reactivate` and `bulk reactivate` commands to reactivate a user.

1.8.4 Assign an Organization

Use Organizations to group users together in the Code42 environment.

You'll need an organization's unique identifier number (UID) to move a user into it. You can use the `list` command to view a list of all current user organizations, including UIDs:

```
code42 users orgs list
```

Use the `show` command to view all the details of a user organization. As an example, to print the details of an organization associated with the UID `123456789` in JSON format:

```
code42 users show 123456789 -f JSON
```

Once you've identified your organizations UID number, use the `move` command to move a user into that organization. In the following example a user is moved into the organization associated with the UID `1234567890`:

```
code42 users move --username sean.cassidy@example.com --org-id 1234567890
```

Alternatively, to move multiple users between organizations, fill out the `move` CSV file template, then use the `bulk move` command with the CSV file path.

```
code42 users bulk move bulk-command.csv
```

1.8.5 Get CSV Template for bulk commands

The following command generates a CSV template for each of the available bulk user commands. The CSV file is saved to the current working directory.

```
code42 users bulk generate-template [update|move|add-alias|remove-alias|update-risk-
↪profile]
```

Once generated, fill out and use each of the CSV templates with their respective bulk commands.

```
code42 users bulk [update|move|deactivate|reactivate|add-alias|remove-alias|update-risk-
↪profile] bulk-command.csv
```

A CSV with a `username` column and a single username on each new line is used for the `reactivate` and `deactivate` bulk commands. These commands are not available as options for `generate-template`.

Learn more about *Managing Users*.

1.9 Configure Trusted Activities

You can add trusted activities to your organization to prevent file activity associated with these locations from appearing in your security event dashboards, user profiles, and alerts.

1.9.1 Get CSV Template

The following command generates a CSV template to either create, update, or remove multiple trusted activities at once. The CSV file is saved to the current working directory.

```
code42 trusted-activities bulk generate-template [create|update|remove]
```

You can then fill out and use each of the CSV templates with their respective bulk commands.

```
code42 trusted-activities bulk [create|update|remove] bulk-command.csv
```

1.9.2 Add a New Trusted Activity

Use the `create` command to add a new trusted domain or Slack workspace to your organization's trusted activities.

```
code42 trusted-activities create DOMAIN mydomain.com --description "a new trusted␣
↪activity"
```

To add multiple trusted activities at once, enter information about the trusted activity into the `create` CSV file template. For each activity, the `type` and `value` fields are required.

`type` indicates the category of activity:

- `DOMAIN` indicates a trusted domain
- `SLACK` indicates a trusted Slack workspace

`value` indicates either the name of the domain or Slack workspace.

Then use the `bulk create` command with the CSV file path. For example:

```
code42 trusted-activities bulk create create_trusted_activities.csv
```

1.9.3 Update a Trusted Activity

Use the `update` command to update either the value or description of a single trusted activity. The `resource_id` of the activity is required. The other fields are optional.

```
code42 trusted-activities update 123 --value my-updated-domain.com --description "an␣
↪updated trusted activity"
```

To update multiple trusted activities at once, enter information about the trusted activity into the `update` CSV file template, then use the `bulk update` command with the CSV file path.

```
code42 trusted-activities bulk update update_trusted_activities.csv
```

Note: The `bulk update` command cannot be used to clear the description of a trusted activity because you cannot indicate an empty string in a CSV format. Pass an empty string to the `description` option of the `update` command to clear the description of a trusted activity.

For example: `code42 trusted-activities update 123 --description ""`

1.9.4 Remove a Trusted Activity

Use the `remove` command to remove a single trusted activity. Only the `resource_id` of an activity is required to remove it.

```
code42 trusted-activities remove 123
```

To remove multiple trusted activities at once, enter information about the trusted activity into the `remove` CSV file template, then use the `bulk remove` command with the CSV file path.

```
code42 trusted-activities bulk remove remove_trusted_activities.csv
```

Learn more about the *Trusted Activities* commands.

1.10 Add Users to Alert Rules

Once you create an alert rule in the Code42 console, you can use the CLI `alert-rules` commands to add and remove users from your existing alert rules.

To see a list of all the users currently in your organization:

- Export a list from the Users action menu.
- Use the *CLI users commands*.

1.10.1 View Existing Alert Rules

You'll need the ID of an alert rule to add or remove a user.

To view a list of all alert rules currently created for your organization, including the rule ID, use the following command:

```
code42 alert-rules list
```

Once you've identified the rule ID, view the details of the alert rule as follows:

```
code42 alert-rules show <rule-ID>
```

Example output

Example output for a single alert rule in default JSON format.


```
{
    "type$": "ENDPOINT_EXFILTRATION_RULE_DETAILS_RESPONSE",
    "rules": [
        {
            "type$": "ENDPOINT_EXFILTRATION_RULE_DETAILS",
            "tenantId": "c4e43418-07d9-4a9f-a138-29f39a124d33",
            "name": "My Rule",
            "description": "this is your rule!",
            "severity": "HIGH",
            "isEnabled": false,
            "fileBelongsTo": {
                "type$": "FILE_BELONGS_TO",
                "usersToAlertOn": "ALL_USERS"
            },
            "notificationConfig": {
                "type$": "NOTIFICATION_CONFIG",
                "enabled": false
            },
            "fileCategoryWatch": {
                "type$": "FILE_CATEGORY_WATCH",
                "watchAllFiles": true
            },
            "ruleSource": "Alerting",
            "fileSizeAndCount": {
                "type$": "FILE_SIZE_AND_COUNT",
                "fileCountGreaterThan": 2,
                "totalSizeGreaterThanInBytes": 200,
                "operator": "AND"
            },
            "fileActivityIs": {
                "type$": "FILE_ACTIVITY",
                "syncedToCloudService": {
                    "type$": "SYNCED_TO_CLOUD_SERVICE",
                    "watchBox": false,
                    "watchBoxDrive": false,
                    "watchDropBox": false,
                    "watchGoogleBackupAndSync": false,
```


(continues on next page)

(continued from previous page)


```
                    "watchAppleIcLoud": false,
                    "watchMicrosoftOneDrive": false
                },
                "uploadedOnRemovableMedia": true,
                "readByBrowserOrOther": true
            },
            "timeWindow": 15,
            "id": "404ff012-fa2f-4acf-ae6d-107eabf7f24c",
            "createdAt": "2021-04-27T01:55:36.4204590Z",
            "createdBy": "sean.cassidy@example.com",
            "modifiedAt": "2021-09-03T01:46:13.2902310Z",
            "modifiedBy": "sean.cassidy@example.com",
            "isSystem": false
        }
    ]
}
```


1.10.2 Add a User to an Alert Rule

You can manage the users who are associated with an alert rule once you know the rule's `rule_id` and the user's `username`.

To add a single user to your alert rule, use the following command:

```
code42 alert-rules add-user --rule-id <rule-id> -u sean.cassidy@example.com
```

Alternatively, to add multiple users to your alert rule, fill out the `add` CSV file template, then use the `bulk add` command with the CSV file path.

```
code42 alert-rules bulk add users.csv
```

You can remove single or multiple users from alert rules similarly using the `remove-user` and `bulk remove` commands.

1.10.3 Get CSV Template

The following command will generate a CSV template to either add or remove users from multiple alert rules at once. The CSV file will be saved to the current working directory.

```
code42 alert-rules bulk generate-template [add|remove]
```

You can then fill out and use each of the CSV templates with their respective bulk commands.

```
code42 alert-rules bulk [add|remove] /Users/my_user/bulk-command.csv
```

Learn more about the *Alert Rules* commands.

1.11 Add and Manage Cases

To create a new case, only the name is required. Other attributes are optional and can be provided through the available flags.

The following command creates a case with the `subject` and `assignee` user indicated by their respective UIDs.

```
code42 cases create My-Case --subject 123 --assignee 456 --description "Sample case"
```

1.11.1 Update a Case

To further update or view the details of your case, you'll need the case's unique number, which is assigned upon creation. To get this number, you can use the `list` command to view all cases, with optional filter values.

To print to the console all open cases created in the last 30 days:

```
code42 cases list --begin-create-time 30d --status OPEN
```

Example Output

Example output for a single case in JSON format.


```
{
    "number": 42,
    "name": "My-Case",
    "createdAt": "2021-9-17T18:29:53.375136Z",
    "updatedAt": "2021-9-17T18:29:53.375136Z",
    "description": "Sample case",
    "findings": "",
    "subject": "123",
    "subjectUsername": "sean.cassidy@example.com",
    "status": "OPEN",
    "assignee": "456",
    "assigneeUsername": "elvis.presley@example.com",
    "createdByUserUid": "789",
    "createdByUsername": "andy.warhol@example.com",
    "lastModifiedByUserUid": "789",
    "lastModifiedByUsername": "andy.warhol@example.com"
}
```


Once you've identified your case's number, you can view further details on the case, or update its attributes.

The following command will print all details of your case.

```
code42 cases show 42
```

If you've finished your investigation and you'd like to close your case, you can update the status of the case. Similarly, other attributes of the case can be updated using the optional flags.

```
code42 cases update 42 --status CLOSED
```

1.11.2 Get CSV Template

The following command will generate a CSV template to either add or remove file events from multiple cases at once. The csv file will be saved to the current working directory.

```
code42 cases file-events bulk generate-template [add|remove]
```

You can then fill out and use each of the CSV templates with their respective bulk commands.

```
code42 cases file-events bulk [add|remove] bulk-command.csv
```

1.11.3 Manage File Exposure Events Associated with a Case

The following example command can be used to view all the file exposure events currently associated with a case, indicated here by case number `42`.

```
code42 cases file-events list 42
```

Use the `file-events add` command to associate a single file event, referred to by event ID, to a case.

Below is an example command to associate some event with ID `event_abc` with case number `42`.

```
code42 cases file-events add 42 event_abc
```

To associate multiple file events with one or more cases at once, enter the case and file event information into the `file-events add` CSV file template, then use the `bulk add` command with the CSV file path. For example:

```
code42 cases file-events bulk add my_new_cases.csv
```

Similarly, the `file-events remove` and `file-events bulk remove` commands can be used to remove a file event from a case.

1.11.4 Export Case Details

You can use the CLI to export the details of a case into a PDF.

The following example command will download the details from case number `42` and save a PDF with the name `42_case_summary.pdf` to the provided path. If a path is not provided, it will be saved to the current working directory.

```
code42 cases export 42 --path /Users/my_user/cases/
```

Learn more about the *Managing Cases*.

1.12 Using Bulk Commands

Bulk functionality is available for many Code42 CLI methods, more details on which commands have bulk capabilities can be found in the *Commands Documentation*.

All bulk methods take a CSV file as input.

The `generate-template` command can be used to create a CSV file with the necessary headers for a particular command.

For instance, the following command will create a file named `devices_bulk_deactivate.csv` with a single column header row of `guid`.

```
code42 devices bulk generate-template deactivate
```

The CSV file can contain more columns than are necessary for the command, however then the header row is **required**.

If the CSV file contains the *exact* number of columns that are necessary for the command then the header row is **optional**, but columns are expected to be in the same order as the template.

To run a bulk method, simply pass the CSV file path to the desired command. For example, you would use to following command to deactivate multiple devices within your organization at once:

```
code42 devices bulk deactivate devices_bulk_deactivate.csv
```

1.13 Manage watchlist members

1.13.1 List created watchlists

To list all the watchlists active in your Code42 environment, run:

```
code42 watchlists list
```

1.13.2 List all members of a watchlist

You can list watchlists either by their Type:

```
code42 watchlists list-members --watchlist-type DEPARTING_EMPLOYEE
```

or by their ID (get watchlist IDs from `code42 watchlist list` output):

```
code42 watchlists list-members --watchlist-id 6e6c5acc-2568-4e5f-8324-e73f2811fa7c
```

A "member" of a watchlist is any user that the watchlist alerting rules apply to. Users can be members of a watchlist either by being explicitly added (via console or `code42 watchlists add [USER_ID|USERNAME]`), but they can also be implicitly included based on some user profile property (like working in a specific department). To get a list of only those "members" who have been explicitly added (and thus can be removed via the `code42 watchlists remove [USER_ID|USERNAME]` command), add the `--only-included-users` option to `list-members`.

1.13.3 Add or remove a single user from watchlist membership

A user can be added to a watchlist using either the watchlist ID or Type, just like listing watchlists, and the user can be identified either by their user_id or their username:

```
code42 watchlist add --watchlist-type NEW_EMPLOYEE 9871230
```

```
code42 watchlist add --watchlist-id 6e6c5acc-2568-4e5f-8324-e73f2811fa7c user@example.com
```

1.13.4 Bulk adding/removing users from watchlists

The bulk watchlist commands read input from a CSV file.

Like the individual commands, they can take either a user_id/username or watchlist_id/watchlist_type to identify who to add to which watchlist. Because of this flexibility, the CSV does require a header row identifying each column.

You can generate a template CSV with the correct header values using the command:

```
code42 watchlists bulk generate-template [add|remove]
```

If both username and user_id are provided in the CSV row, the user_id value will take precedence. If watchlist_type and watchlist_id columns are both provided, the watchlist_id will take precedence.

Note: For watchlists that track additional metadata for a user (e.g. the "departure date" for a user on the Departing watchlist), that data can be added/updated via the code42 users bulk update-risk-profile command.

You can re-use the same CSV file for both commands, just add the required risk profile columns to the CSV.

For example, to bulk add users to multiple watchlists, with appropriate `start_date`, `end_date`, and `notes` values, create a CSV (in this example named `watchlists.csv`) with the following:

```
username,watchlist_type,start_date,end_date,notes
user_a@example.com,DEPARTING_EMPLOYEE,,2023-10-10,
user_b@example.com,NEW_EMPLOYEE,2022-07-04,,2022 Summer Interns
```

Then run `code42 watchlists bulk add watchlists.csv` followed by `code42 users bulk update-risk-profile watchlists.csv`

1.14 (DEPRECATED) Manage Detection List Users

Note: Detection Lists have been replaced by Watchlists.

Functionality for adding users to Departing Employee and High Risk Employee categories has been migrated to the `code42 watchlists` command group.

Functionality for listing and managing User Risk Profiles (e.g. adding Cloud Aliases, Notes, and Start/End dates to a user profile) has been migrated to the `code42 users` command group.

Use the `departing-employee` commands to add employees to or remove employees from the Departing Employees list. Use the `high-risk-employee` commands to add employees to or remove employees from the High Risk list, or update risk tags for those users.

To see a list of all the users currently in your organization:

- Export a list from the Users action menu.
- Use the *CLI users commands*.

1.14.1 Get CSV template

To add multiple users to the Departing Employees list:

1. Generate a CSV template. Below is an example command for generating a template to use to add employees to the Departing Employees list. Once generated, the CSV file is saved to your current working directory.

```
code42 departing-employee bulk generate-template add
```

1. Use the CSV template to enter the employees' information. Only the Code42 username is required. If added, the departure date must be in yyyy-MM-dd format. Note: you are only able to add departure dates during the `add` operation. If you don't include `--departure-date`, you can only add one later by removing and then re-adding the employee.
2. Save the CSV file.

1.14.2 Add users to the Departing Employees list

Once you have entered the employees' information in the CSV file, use the `bulk add` command with the CSV file path to add multiple users at once. For example:

```
code42 departing-employee bulk add /Users/astrid.ludwig/add_departing_employee.csv
```

1.14.3 Remove users

You can remove one or more users from the High Risk Employees list. Use `code42 departing-employee remove` to remove a single user.

To remove multiple users at once:

1. Create a CSV file with one username per line.
2. Save the file to your current working directory.
3. Use the `bulk remove` command. For example:

```
code42 high-risk-employee bulk remove /Users/matt.allen/remove_high_risk_employee.csv
```

Learn more about the *Departing Employee* and *High Risk Employee* commands.

- *Get started with the Code42 command-line interface (CLI)*
- *Configure a profile*
- *Enable V2 File Events*
- *Ingest data into a SIEM*
- *Manage legal hold users*
- *Clean up your environment by deactivating devices*
- *Write custom extension scripts using the Code42 CLI and Py42*
- *Manage users*
- *Configure trusted activities*
- *Configure alert rules*
- *Add and manage cases*

- *Perform bulk actions*
- *Manage watchlist members*
- *(DEPRECATED) Manage detection list users*

CHAPTER

TWO

COMMANDS

2.1 alert-rules

Manage users associated with alert rules.

```
alert-rules [OPTIONS] COMMAND [ARGS]...
```

2.1.1 add-user

Add a user to an alert rule.

```
alert-rules add-user [OPTIONS]
```

Options

--rule-id <rule_id>
: **Required** Identification number of the alert rule.

-u, --username <username>
: **Required** The username of the user to add to the alert rule.

-d, --debug
: Turn on debug logging.

--totp <totp>
: TOTP token for multi-factor authentication.

--profile <profile>
: The name of the Code42 CLI profile to use when executing this command.

2.1.2 bulk

Tools for executing bulk alert rule actions.

```
alert-rules bulk [OPTIONS] COMMAND [ARGS]...
```

add

Bulk add users to alert rules from a CSV file. CSV file format: rule_id,username

```
alert-rules bulk add [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk adding/removing users.

```
alert-rules bulk generate-template [OPTIONS] {add|remove}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

remove

Bulk remove users from alert rules using a CSV file. CSV file format: {','.join(ALERT_RULES_CSV_HEADERS)}

```
alert-rules bulk remove [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.1.3 list

Fetch existing alert rules.

```
alert-rules list [OPTIONS]
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.1.4 remove-user

Remove a user from an alert rule.

```
alert-rules remove-user [OPTIONS]
```

Options

--rule-id <rule_id>
Required Identification number of the alert rule.

-u, --username <username>
Required The username of the user to remove from the alert rule.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.1.5 show

Print out detailed alert rule criteria.

```
alert-rules show [OPTIONS] RULE_ID
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

RULE_ID
Required argument

2.2 alerts

Get and send alert data.

```
alerts [OPTIONS] COMMAND [ARGS]...
```

2.2.1 bulk

Tools for executing bulk alert actions.

```
alerts bulk [OPTIONS] COMMAND [ARGS]...
```

generate-template

Generate the CSV template needed for bulk alert commands.

```
alerts bulk generate-template [OPTIONS] {update}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

update

Bulk update alerts using a CSV file with format: id,state,note

```
alerts bulk update [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.2.2 clear-checkpoint

Remove the saved alert checkpoint from *–use-checkpoint/-c* mode.

```
alerts clear-checkpoint [OPTIONS] CHECKPOINT_NAME
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CHECKPOINT_NAME
Required argument

2.2.3 search

Search for alerts.

```
alerts search [OPTIONS]
```

Options

--state <state>
Filter alerts by status. Defaults to returning all statuses.

Options RESOLVED | IN_PROGRESS | OPEN | PENDING

--severity <severity>
Filter alerts by severity. Defaults to returning all severities.

Options CRITICAL | HIGH | LOW | MODERATE | MODERATE

--description <description>
Filter alerts by description. Does fuzzy search by default.

--exclude-rule-type <exclude_rule_type>
Filter alerts by excluding the given rule type(s).

--rule-type <rule_type>
Filter alerts by including the given rule type(s).

Options FedCloudSharePermissions | FedEndpointExfiltration | FedFileTypeMismatch

--exclude-rule-id <exclude_rule_id>
Filter alerts by excluding the given rule id(s).

--rule-id <rule_id>
Filter alerts by including the given rule id(s).

--exclude-rule-name <exclude_rule_name>
Filter alerts by excluding the given rule name(s).

--rule-name <rule_name>
Filter alerts by including the given rule name(s).

--exclude-actor-contains <exclude_actor_contains>
Filter alerts by excluding actor(s) whose cloud alias contains the given string.

--exclude-actor <exclude_actor>
Filter alerts by excluding the given actor(s) who triggered the alert. Arguments must match actor's cloud alias exactly.

--actor-contains <actor_contains>
Filter alerts by including actor(s) whose cloud alias contains the given string.

--actor <actor>
Filter alerts by including the given actor(s) who triggered the alert. Arguments must match the actor's cloud alias exactly.

-b, --begin <begin>
The beginning of the date range in which to look for alerts. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time. [required unless –use-checkpoint option used]

-e, --end <end>
The end of the date range in which to look for alerts, argument format options are the same as *–begin*.

--advanced-query <QUERY_JSON>
A raw JSON alerts query. Useful for when the provided query parameters do not satisfy your requirements. Argument can be passed as a string, read from stdin by passing '-', or from a filename if prefixed with '@', e.g. '–advanced-query @query.json'. WARNING: Using advanced queries is incompatible with other query-building arguments.

-c, --use-checkpoint <use_checkpoint>
Use a checkpoint with the given name to only get alerts that were not previously retrieved.If a checkpoint for alerts with the given name doesn't exist, it will be created on the first run.Subsequent CLI runs with this flag and the same name will use the stored checkpoint to modify the search query and then update the stored checkpoint

--or-query

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

--include-all
Display simple properties of the primary level of the nested response.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

2.2.4 send-to

Send alerts to the given server address.

HOSTNAME format: address:port where port is optional and defaults to 514.

```
alerts send-to [OPTIONS] HOSTNAME
```

Options

--state <state>
Filter alerts by status. Defaults to returning all statuses.

Options RESOLVED | IN_PROGRESS | OPEN | PENDING

--severity <severity>
Filter alerts by severity. Defaults to returning all severities.

Options CRITICAL | HIGH | LOW | MODERATE | MODERATE

--description <description>
Filter alerts by description. Does fuzzy search by default.

--exclude-rule-type <exclude_rule_type>
Filter alerts by excluding the given rule type(s).

--rule-type <rule_type>
Filter alerts by including the given rule type(s).

Options FedCloudSharePermissions | FedEndpointExfiltration | FedFileTypeMismatch

--exclude-rule-id <exclude_rule_id>
Filter alerts by excluding the given rule id(s).

--rule-id <rule_id>
Filter alerts by including the given rule id(s).

--exclude-rule-name <exclude_rule_name>
Filter alerts by excluding the given rule name(s).

--rule-name <rule_name>
Filter alerts by including the given rule name(s).

--exclude-actor-contains <exclude_actor_contains>
Filter alerts by excluding actor(s) whose cloud alias contains the given string.

--exclude-actor <exclude_actor>
Filter alerts by excluding the given actor(s) who triggered the alert. Arguments must match actor's cloud alias exactly.

--actor-contains <actor_contains>
Filter alerts by including actor(s) whose cloud alias contains the given string.

--actor <actor>
Filter alerts by including the given actor(s) who triggered the alert. Arguments must match the actor's cloud alias exactly.

-b, --begin <begin>
The beginning of the date range in which to look for alerts. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g.

'2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time. [required unless –use-checkpoint option used]

-e, --end <end>
The end of the date range in which to look for alerts, argument format options are the same as *–begin*.

--advanced-query <QUERY_JSON>
A raw JSON alerts query. Useful for when the provided query parameters do not satisfy your requirements. Argument can be passed as a string, read from stdin by passing '-', or from a filename if prefixed with '@', e.g. '–advanced-query @query.json'. WARNING: Using advanced queries is incompatible with other query-building arguments.

-c, --use-checkpoint <use_checkpoint>
Use a checkpoint with the given name to only get alerts that were not previously retrieved.If a checkpoint for alerts with the given name doesn't exist, it will be created on the first run.Subsequent CLI runs with this flag and the same name will use the stored checkpoint to modify the search query and then update the stored checkpoint

--or-query

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

--ignore-cert-validation
Set to skip CA certificate validation. Incompatible with the 'certs' option.

--certs <certs>
A CA certificates-chain file for the TCP-TLS protocol.

-p, --protocol <protocol>
Protocol used to send logs to server. Use TCP-TLS for additional security. Defaults to UDP.

Options TCP | UDP | TLS-TCP

--include-all
Display simple properties of the primary level of the nested response.

-f, --format <format>
The output format of the result. Defaults to json format.

Options JSON | RAW-JSON

Arguments

HOSTNAME
Required argument

2.2.5 show

Display the details of a single alert.

```
alerts show [OPTIONS] ALERT_ID
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

--include-observations
View observations of the alert.

Arguments

ALERT_ID
Required argument

2.2.6 update

Update alert information.

```
alerts update [OPTIONS] ALERT_ID
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

--state <state>
The state to give to the alert.

Options RESOLVED | IN_PROGRESS | OPEN | PENDING

--note <note>
A note to attach to the alert.

Arguments

ALERT_ID
Required argument

2.3 audit-logs

Get and send audit log event data.

```
audit-logs [OPTIONS] COMMAND [ARGS]...
```

2.3.1 clear-checkpoint

Remove the saved audit log checkpoint from *–use-checkpoint/-c* mode.

```
audit-logs clear-checkpoint [OPTIONS] CHECKPOINT_NAME
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CHECKPOINT_NAME
Required argument

2.3.2 search

Search audit log events.

```
audit-logs search [OPTIONS]
```

Options

-b, --begin <begin>
The beginning of the date range in which to look for audit-logs. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time. [required unless –use-checkpoint option used]

-e, --end <end>
The end of the date range in which to look for audit-logs, argument format options are the same as *–begin*.

--affected-username <affected_username>
Filter results by affected usernames.

--affected-user-id <affected_user_id>
Filter results by affected user IDs.

--actor-ip <actor_ip>
Filter results by user IP addresses.

--actor-user-id <actor_user_id>
Filter results by actor user IDs.

--actor-username <actor_username>
Filter results by actor usernames.

--event-type <event_type>
Filter results by event types (e.g. search_issued, user_registered, user_deactivated).

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-c, --use-checkpoint <use_checkpoint>
Use a checkpoint with the given name to only get audit-logs that were not previously retrieved.If a checkpoint for audit-logs with the given name doesn't exist, it will be created on the first run.Subsequent CLI runs with this flag and the same name will use the stored checkpoint to modify the search query and then update the stored checkpoint

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.3.3 send-to

Send audit log events to the given server address in JSON format.

HOSTNAME format: address:port where port is optional and defaults to 514.

```
audit-logs send-to [OPTIONS] HOSTNAME
```

Options

-b, --begin <begin>
The beginning of the date range in which to look for audit-logs. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time. [required unless –use-checkpoint option used]

-e, --end <end>
The end of the date range in which to look for audit-logs, argument format options are the same as *–begin*.

--affected-username <affected_username>
Filter results by affected usernames.

--affected-user-id <affected_user_id>
Filter results by affected user IDs.

--actor-ip <actor_ip>
Filter results by user IP addresses.

--actor-user-id <actor_user_id>
Filter results by actor user IDs.

--actor-username <actor_username>
Filter results by actor usernames.

--event-type <event_type>
Filter results by event types (e.g. search_issued, user_registered, user_deactivated).

-c, --use-checkpoint <use_checkpoint>
Use a checkpoint with the given name to only get audit-logs that were not previously retrieved.If a checkpoint for audit-logs with the given name doesn't exist, it will be created on the first run.Subsequent CLI runs with this flag and the same name will use the stored checkpoint to modify the search query and then update the stored checkpoint

--ignore-cert-validation
Set to skip CA certificate validation. Incompatible with the 'certs' option.

--certs <certs>
A CA certificates-chain file for the TCP-TLS protocol.

-p, --protocol <protocol>
Protocol used to send logs to server. Use TCP-TLS for additional security. Defaults to UDP.

Options TCP | UDP | TLS-TCP

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

HOSTNAME
Required argument

2.4 cases

Manage cases and events associated with cases.

```
cases [OPTIONS] COMMAND [ARGS]...
```

2.4.1 create

Create a new case.

```
cases create [OPTIONS] NAME
```

Options

--assignee <assignee>
The UID of the user to assign to the case.

--description <description>
The description of the case.

--findings <findings>
Any findings for the case.

--subject <subject>
The user UID of the subject of the case.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

NAME
Required argument

2.4.2 export

Download a case detail summary as a PDF file at the given path with name <case_number>_case_summary.pdf.

```
cases export [OPTIONS] CASE_NUMBER
```

Options

--path <path>
The file path where to save the PDF. Defaults to the current directory.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CASE_NUMBER
Required argument

2.4.3 file-events

Fetch file events associated with the case.

```
cases file-events [OPTIONS] COMMAND [ARGS]...
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

add

Associate a file event to a case, by event ID.

```
cases file-events add [OPTIONS]
```

Options

--case-number <case_number>
Required The number assigned to the case.

--event-id <event_id>
Required The file event ID associated with the case.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

bulk

Tools for executing bulk case file-event actions.

```
cases file-events bulk [OPTIONS] COMMAND [ARGS]...
```

add

Bulk associate file events to cases using a CSV file with format: number,event_id.

```
cases file-events bulk add [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk adding/removing users.

```
cases file-events bulk generate-template [OPTIONS] {add|remove}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

remove

Bulk remove the file event association from cases using a CSV file with format: number,event_id.

```
cases file-events bulk remove [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

list

List all the file events associated with the case.

```
cases file-events list [OPTIONS] CASE_NUMBER
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

Arguments

CASE_NUMBER
Required argument

remove

Remove the associated file event from the case, by event ID.

```
cases file-events remove [OPTIONS]
```

Options

--case-number <case_number>
Required The number assigned to the case.

--event-id <event_id>
Required The file event ID associated with the case.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.4.4 list

List all the cases.

```
cases list [OPTIONS]
```

Options

--name <name>
Filter by name of a case. Supports partial name matches.

--subject <subject>
Filter by the user UID of the subject of a case.

--assignee <assignee>
Filter by the user UID of an assignee.

--begin-create-time <begin_create_time>
The beginning of the date range in which to look for cases. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time.

--end-create-time <end_create_time>
The end of the date range in which to look for cases, argument format options are the same as *–begin*.

--begin-update-time <begin_update_time>
The beginning of the date range in which to look for cases. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time.

--end-update-time <end_update_time>
The end of the date range in which to look for cases, argument format options are the same as *–begin*.

--status <status>
Filter cases by case status.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.4.5 show

Show case details.

```
cases show [OPTIONS] CASE_NUMBER
```

Options

--include-file-events
View file events associated to the case.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

Arguments

CASE_NUMBER
Required argument

2.4.6 update

Update case details for the given case.

```
cases update [OPTIONS] CASE_NUMBER
```

Options

--name <name>
The name of the case.

--assignee <assignee>
The UID of the user to assign to the case.

--description <description>
The description of the case.

--findings <findings>
Any findings for the case.

--subject <subject>
The user UID of the subject of the case.

--status <status>
Status of the case. *OPEN* or *CLOSED*.

Options CLOSED | OPEN

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CASE_NUMBER
Required argument

2.5 devices

Manage devices within your Code42 environment.

```
devices [OPTIONS] COMMAND [ARGS]...
```

2.5.1 bulk

Tools for managing devices in bulk.

```
devices bulk [OPTIONS] COMMAND [ARGS]...
```

deactivate

Deactivate all devices from the provided CSV containing a 'guid' column.

```
devices bulk deactivate [OPTIONS] CSV_FILE
```

Options

--change-device-name
Prepend 'deactivated_<current_date>' to the name of any successfully deactivated devices.

--purge-date <purge_date>
The date on which the archive should be purged from cold storage in yyyy-MM-dd format. If not provided, the date will be set according to the appropriate organization settings.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk device commands.

```
devices bulk generate-template [OPTIONS] {reactivate|deactivate|rename}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

reactivate

Reactivate all devices from the provided CSV containing a 'guid' column.

```
devices bulk reactivate [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

rename

Rename all devices from the provided CSV containing a 'guid' and a 'name' column.

```
devices bulk rename [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.5.2 deactivate

Deactivate a device within Code42. Requires the device GUID to deactivate.

```
devices deactivate [OPTIONS] DEVICE_GUID
```

Options

--change-device-name
Prepend 'deactivated_<current_date>' to the name of the device if deactivation is successful.

--purge-date <purge_date>
The date on which the archive should be purged from cold storage in yyyy-MM-dd format. If not provided, the date will be set according to the appropriate organization settings.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

DEVICE_GUID
Required argument

2.5.3 list

Get information about many devices.

```
devices list [OPTIONS]
```

Options

--active
Limits results to only active devices.

--inactive
Limits results to only deactivated devices.

--org-uid <org_uid>
Limit devices to only those in the organization you specify. Note that child organizations will be included.

--include-backup-usage
Return backup usage information for each device (may significantly lengthen the size of the return).

--include-usernames
Add the username associated with a device to the output.

--include-settings
Include device settings in output.

--include-legal-hold-membership
Include legal hold membership in output.

--include-total-storage
Include backup archive count and total storage in output.

--exclude-most-recently-connected <exclude_most_recently_connected>
Filter out the N most recently connected devices per user. Useful for identifying duplicate and/or replaced devices that are no longer needed across an environment. If a user has 2 devices and N=1, the one device with the most recent 'lastConnected' date will not show up in the result list.

--last-connected-before <last_connected_before>
Include devices only when the 'lastConnected' field is after the provided value. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time.

--last-connected-after <last_connected_after>
Include devices only when 'lastConnected' field is after the provided value. Argument format options are the same as –last-connected-before.

--created-before <created_before>
Include devices only when 'creationDate' field is less than the provided value. Argument format options are the same as –last-connected-before.

--created-after <created_after>
Include devices only when 'creationDate' field is greater than the provided value. Argument format options are the same as –last-connected-before.

--page-size <page_size>
Number of devices to retrieve per API call. Lower this value if you are getting timeouts when retrieving devices with backup info. Default: 100

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.5.4 list-backup-sets

Get information about many devices and their backup sets.

```
devices list-backup-sets [OPTIONS]
```

Options

--active
Limits results to only active devices.

--inactive
Limits results to only deactivated devices.

--org-uid <org_uid>
Limit devices to only those in the organization you specify. Note that child organizations will be included.

--include-usernames
Add the username associated with a device to the output.

--page-size <page_size>
Number of devices to retrieve per API call. Lower this value if you are getting timeouts when retrieving devices with backup info. Default: 100

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.5.5 reactivate

Reactivate a device within Code42. Requires the device GUID to reactivate.

```
devices reactivate [OPTIONS] DEVICE_GUID
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

DEVICE_GUID
Required argument

2.5.6 rename

Rename a device with Code42. Requires the device GUID to rename.

```
devices rename [OPTIONS] DEVICE_GUID
```

Options

-n, --new-device-name <new_device_name>
Required The new name for the device.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

DEVICE_GUID
Required argument

2.5.7 show

Print individual device details. Requires device GUID.

```
devices show [OPTIONS] DEVICE_GUID
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

DEVICE_GUID
Required argument

2.6 legal-hold

Add and remove custodians from legal hold matters.

```
legal-hold [OPTIONS] COMMAND [ARGS]...
```

2.6.1 add-user

Add a custodian to a legal hold matter.

```
legal-hold add-user [OPTIONS]
```

Options

-m, --matter-id <matter_id>
Required Identification number of the legal hold matter the custodian will be added to.

-u, --username <username>
Required The username of the custodian to add to the matter.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.6.2 bulk

Tools for executing bulk legal hold actions.

```
legal-hold bulk [OPTIONS] COMMAND [ARGS]...
```

add

Bulk add custodians to legal hold matters using a CSV file. CSV file format: matter_id,username

```
legal-hold bulk add [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk adding/removing users.

```
legal-hold bulk generate-template [OPTIONS] {add|remove}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

remove

Bulk release custodians from legal hold matters using a CSV file. CSV file format: matter_id,username

```
legal-hold bulk remove [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.6.3 list

Fetch existing legal hold matters.

```
legal-hold list [OPTIONS]
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.6.4 remove-user

Release a custodian from a legal hold matter.

```
legal-hold remove-user [OPTIONS]
```

Options

-m, --matter-id <matter_id>
Required Identification number of the legal hold matter the custodian will be removed from.

-u, --username <username>
Required The username of the custodian to add to the matter.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.6.5 search-events

Tools for getting legal hold event data.

```
legal-hold search-events [OPTIONS]
```

Options

-m, --matter-id <matter_id>
Filter results by legal hold UID.

--event-type <event_type>
Filter results by event types.

Options MembershipCreated | MembershipReactivated | MembershipDeactivated | HoldCreated | HoldDeactivated | HoldReactivated | Restore

--begin <begin>
The beginning of the date range in which to look for legal hold events. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time.

--end <end>
The end of the date range in which to look for legal hold events, argument format options are the same as *–begin*.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.6.6 show

Display details of a given legal hold matter.

```
legal-hold show [OPTIONS] MATTER_ID
```

Options

--include-inactive
View all custodians associated with the legal hold matter, including inactive custodians.

--include-policy
View details of the preservation policy associated with the legal hold matter.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

MATTER_ID
Required argument

2.7 profile

Manage Code42 connection settings.

```
profile [OPTIONS] COMMAND [ARGS]...
```

2.7.1 create

Create a profile with username/password authentication. The first profile created will be the default.

```
profile create [OPTIONS]
```

Options

-n, --name <name>
Required The name of the Code42 CLI profile to use when executing this command.

-s, --server <server>
Required The URL you use to sign into Code42.

-u, --username <username>
Required The username of the Code42 API user.

--password <password>
The password for the Code42 API user. If this option is omitted, interactive prompts will be used to obtain the password.

--totp <totp>
TOTP token for multi-factor authentication.

--disable-ssl-errors <disable_ssl_errors>
For development purposes, do not validate the SSL certificates of Code42 servers. This is not recommended, except for specific scenarios like testing. Attach this flag to the update command to toggle the setting.

--use-v2-file-events <use_v2_file_events>
Opts to use the V2 file event data model. Attach this flag to the update command to toggle the setting

-d, --debug
Turn on debug logging.

2.7.2 create-api-client

Create a profile with Code42 API client authentication. The first profile created will be the default.

```
profile create-api-client [OPTIONS]
```

Options

-n, --name <name>
Required The name of the Code42 CLI profile to use when executing this command.

-s, --server <server>
Required The URL you use to sign into Code42.

--api-client-id <api_client_id>
Required The API client key for API client authentication. Used with the *–secret* option.

--secret <secret>
Required The API secret for API client authentication. Used with the *–api-client* option.

--disable-ssl-errors <disable_ssl_errors>
For development purposes, do not validate the SSL certificates of Code42 servers. This is not recommended, except for specific scenarios like testing. Attach this flag to the update command to toggle the setting.

--use-v2-file-events <use_v2_file_events>
Opts to use the V2 file event data model. Attach this flag to the update command to toggle the setting

-d, --debug
Turn on debug logging.

2.7.3 delete

Deletes a profile and its stored password (if any).

```
profile delete [OPTIONS] PROFILE_NAME
```

Options

-y, --assume-yes
Assume "yes" as the answer to all prompts and run non-interactively.

Arguments

PROFILE_NAME
Required argument

2.7.4 delete-all

Deletes all profiles and saved passwords (if any).

```
profile delete-all [OPTIONS]
```

Options

-y, --assume-yes
Assume "yes" as the answer to all prompts and run non-interactively.

2.7.5 list

Show all existing stored profiles.

```
profile list [OPTIONS]
```

2.7.6 reset-pw

Change the stored password for a profile. Only affects what's stored in the local profile,
does not make any changes to the Code42 user account.

```
profile reset-pw [OPTIONS] [PROFILE_NAME]
```

Options

-d, --debug
Turn on debug logging.

Arguments

PROFILE_NAME
Optional argument

2.7.7 show

Print the details of a profile.

```
profile show [OPTIONS] [PROFILE_NAME]
```

Arguments

PROFILE_NAME
Optional argument

2.7.8 update

Update an existing profile.

```
profile update [OPTIONS]
```

Options

-n, --name <name>
The name of the Code42 CLI profile to use when executing this command.

-s, --server <server>
The URL you use to sign into Code42.

--api-client-id <api_client_id>
The API client key for API client authentication. Used with the *–secret* option.

--secret <secret>
The API secret for API client authentication. Used with the *–api-client* option.

-u, --username <username>
The username of the Code42 API user.

--password <password>
The password for the Code42 API user. If this option is omitted, interactive prompts will be used to obtain the password.

--totp <totp>
TOTP token for multi-factor authentication.

--disable-ssl-errors <disable_ssl_errors>
For development purposes, do not validate the SSL certificates of Code42 servers. This is not recommended, except for specific scenarios like testing. Attach this flag to the update command to toggle the setting.

--use-v2-file-events <use_v2_file_events>
Opts to use the V2 file event data model. Attach this flag to the update command to toggle the setting

-d, --debug
Turn on debug logging.

2.7.9 use

Set a profile as the default. If not providing a profile-name,
prompts for a choice from a list of all profiles.

```
profile use [OPTIONS] [PROFILE_NAME]
```

Arguments

PROFILE_NAME
Optional argument

2.8 Security Data

> **Warning:** V1 file events, saved searches, and queries are **deprecated**.

See more information in the Enable V2 File Events User Guide.

2.8.1 security-data

Get and send file event data.

```
security-data [OPTIONS] COMMAND [ARGS]...
```

clear-checkpoint

Remove the saved file event checkpoint from *–use-checkpoint/-c* mode.

```
security-data clear-checkpoint [OPTIONS] CHECKPOINT_NAME
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CHECKPOINT_NAME
Required argument

saved-search

Search for file events using saved searches.

```
security-data saved-search [OPTIONS] COMMAND [ARGS]...
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

list

List available saved searches.

```
security-data saved-search list [OPTIONS]
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

show

Get the details of a saved search.

```
security-data saved-search show [OPTIONS] SEARCH_ID
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

SEARCH_ID
Required argument

search

Search for file events.

```
security-data search [OPTIONS]
```

Options

--saved-search <saved_search>
Get events from a saved search filter with the given ID.WARNING: Using a saved search is incompatible with other query-building arguments.

--risk-severity <risk_severity>
Limits events to those classified by the given risk severity.

Options CRITICAL | HIGH | LOW | MODERATE | NO_RISK_INDICATED

--risk-indicator <risk_indicator>
Limits events to those classified by the given risk indicator categories.

Options PUBLIC_CORPORATE_BOX | PUBLIC_CORPORATE_GOOGLE | PUBLIC_CORPORATE_ONEDRIVE | SENT_CORPORATE_GMAIL | SHARED_CORPORATE_BOX | SHARED_CORPORATE_GOOGLE_DRIVE | SHARED_CORPORATE_ONEDRIVE | AMAZON_DRIVE | BOX | DROPBOX | GOOGLE_DRIVE | ICLOUD | MEGA | ONEDRIVE | ZOHO | BITBUCKET | GITHUB | GITLAB | SOURCEFORGE | STASH | 163.COM | 126.COM | AOL | COMCAST | GMAIL | ICLOUD_MAIL | MAIL.COM | OUTLOOK | PROTONMAIL | QQMAIL | SINA_MAIL | SOHU_MAIL | YAHOO | ZOHO_MAIL | AIRDROP | REMOVABLE_MEDIA | AUDIO | DOCUMENT | EXECUTABLE | IMAGE | PDF | PRESENTATION | SCRIPT | SOURCE_CODE | SPREADSHEET | VIDEO | VIRTUAL_DISK_IMAGE | ZIP | FACEBOOK_MESSENGER | MICROSOFT_TEAMS | SLACK | WHATSAPP | OTHER | UNKNOWN | FACEBOOK

| LINKEDIN | REDDIT | TWITTER | FILE_MISMATCH | OFF_HOURS | REMOTE | FIRST_DESTINATION_USE | RARE_DESTINATION_USE

--include-non-exposure
Get all events including non-exposure events.

--tab-url <tab_url>
Limits events to be exposure events with one of the specified destination tab URLs.

--process-owner <process_owner>
Limits exposure events by process owner, as reported by the device's operating system. Applies only to *Printed* and *Browser or app read* events.

--file-category <file_category>
Limits events to file events where the file can be classified by one of these categories.

Options Audio | Document | Executable | Image | Pdf | Presentation | Script | SourceCode | Spreadsheet | Video | VirtualDiskImage | Archive

--file-path <file_path>
Limits events to file events where the file is located at one of these paths. Applies to endpoint file events only.

--file-name <file_name>
Limits events to file events where the file has one of these names.

--source <source>
Limits events to only those from one of these sources. For example, Gmail, Box, or Endpoint.

--sha256 <sha256>
Limits events to file events where the file has one of these SHA256 hashes.

--md5 <md5>
Limits events to file events where the file has one of these MD5 hashes.

--actor <actor>
Limits events to only those enacted by the cloud service user of the person who caused the event.

--c42-username <c42_username>
Limits events to endpoint events for these Code42 users.

--event-action <event_action>
Limits events to those with given event action. Only compatible with V2 file events.

Options application-read | file-created | file-deleted | file-downloaded | file-emailed | file-modified | file-printed | file-shared | removable-media-created | removable-media-deleted | removable-media-modified | sync-app-created | sync-app-deleted | sync-app-modified

-t, --type <type>
Limits events to those with given exposure types. Only compatible with V1 file events.

Options ApplicationRead | CloudStorage | IsPublic | OutsideTrustedDomains | RemovableMedia | SharedToDomain | SharedViaLink

-b, --begin <begin>
The beginning of the date range in which to look for file events. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time. [required unless –use-checkpoint option used]

-e, --end <end>
The end of the date range in which to look for file events, argument format options are the same as *–begin*.

--or-query
Combine query filter options with 'OR' logic instead of the default 'AND'.

--advanced-query <QUERY_JSON>
A raw JSON file events query. Useful for when the provided query parameters do not satisfy your requirements. Argument can be passed as a string, read from stdin by passing '-', or from a filename if prefixed with '@', e.g. '–advanced-query @query.json'. WARNING: Using advanced queries is incompatible with other query-building arguments.

-c, --use-checkpoint <use_checkpoint>
Use a checkpoint with the given name to only get file events that were not previously retrieved.If a checkpoint for file events with the given name doesn't exist, it will be created on the first run.Subsequent CLI runs with this flag and the same name will use the stored checkpoint to modify the search query and then update the stored checkpoint

--columns <columns>
Filter output to include only specified columns. Accepts comma-separated list of column names (case-insensitive).

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

--columns <columns>
Filter output to include only specified columns. Accepts comma-separated list of column names (case-insensitive).

--include-all
Display simple properties of the primary level of the nested response.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON | CEF

send-to

Send events to the given server address.

HOSTNAME format: address:port where port is optional and defaults to 514.

```
security-data send-to [OPTIONS] HOSTNAME
```

Options

--saved-search <saved_search>
Get events from a saved search filter with the given ID.WARNING: Using a saved search is incompatible with other query-building arguments.

--risk-severity <risk_severity>
Limits events to those classified by the given risk severity.

Options CRITICAL | HIGH | LOW | MODERATE | NO_RISK_INDICATED

--risk-indicator <risk_indicator>
Limits events to those classified by the given risk indicator categories.

Options PUBLIC_CORPORATE_BOX | PUBLIC_CORPORATE_GOOGLE | PUBLIC_CORPORATE_ONEDRIVE | SENT_CORPORATE_GMAIL | SHARED_CORPORATE_BOX | SHARED_CORPORATE_GOOGLE_DRIVE | SHARED_CORPORATE_ONEDRIVE | AMAZON_DRIVE | BOX | DROPBOX | GOOGLE_DRIVE | ICLOUD | MEGA | ONEDRIVE | ZOHO | BITBUCKET | GITHUB | GITLAB | SOURCEFORGE | STASH | 163.COM | 126.COM | AOL | COMCAST | GMAIL | ICLOUD_MAIL | MAIL.COM | OUTLOOK | PROTONMAIL | QQMAIL | SINA_MAIL | SOHU_MAIL | YAHOO | ZOHO_MAIL | AIRDROP | REMOVABLE_MEDIA | AUDIO | DOCUMENT | EXECUTABLE | IMAGE | PDF | PRESENTATION | SCRIPT | SOURCE_CODE | SPREADSHEET | VIDEO | VIRTUAL_DISK_IMAGE | ZIP | FACEBOOK_MESSENGER | MICROSOFT_TEAMS | SLACK | WHATSAPP | OTHER | UNKNOWN | FACEBOOK | LINKEDIN | REDDIT | TWITTER | FILE_MISMATCH | OFF_HOURS | REMOTE | FIRST_DESTINATION_USE | RARE_DESTINATION_USE

--include-non-exposure
Get all events including non-exposure events.

--tab-url <tab_url>
Limits events to be exposure events with one of the specified destination tab URLs.

--process-owner <process_owner>
Limits exposure events by process owner, as reported by the device's operating system. Applies only to *Printed* and *Browser or app read* events.

--file-category <file_category>
Limits events to file events where the file can be classified by one of these categories.

Options Audio | Document | Executable | Image | Pdf | Presentation | Script | SourceCode | Spreadsheet | Video | VirtualDiskImage | Archive

--file-path <file_path>
Limits events to file events where the file is located at one of these paths. Applies to endpoint file events only.

--file-name <file_name>
Limits events to file events where the file has one of these names.

--source <source>
Limits events to only those from one of these sources. For example, Gmail, Box, or Endpoint.

--sha256 <sha256>
Limits events to file events where the file has one of these SHA256 hashes.

--md5 <md5>
Limits events to file events where the file has one of these MD5 hashes.

--actor <actor>
Limits events to only those enacted by the cloud service user of the person who caused the event.

--c42-username <c42_username>
Limits events to endpoint events for these Code42 users.

--event-action <event_action>
Limits events to those with given event action. Only compatible with V2 file events.

Options application-read | file-created | file-deleted | file-downloaded | file-emailed | file-modified | file-printed | file-shared | removable-media-created | removable-media-deleted | removable-media-modified | sync-app-created | sync-app-deleted | sync-app-modified

-t, --type <type>
Limits events to those with given exposure types. Only compatible with V1 file events.

Options ApplicationRead | CloudStorage | IsPublic | OutsideTrustedDomains | RemovableMedia | SharedToDomain | SharedViaLink

-b, --begin <begin>
The beginning of the date range in which to look for file events. Accepts a date/time in yyyy-MM-dd (UTC) or yyyy-MM-dd HH:MM:SS (UTC+24-hr time) format where the 'time' portion of the string can be partial (e.g. '2020-01-01 12' or '2020-01-01 01:15') or a 'short time' value representing days (30d), hours (24h) or minutes (15m) from the current time. [required unless –use-checkpoint option used]

-e, --end <end>
The end of the date range in which to look for file events, argument format options are the same as *–begin*.

--or-query
Combine query filter options with 'OR' logic instead of the default 'AND'.

--advanced-query <QUERY_JSON>
A raw JSON file events query. Useful for when the provided query parameters do not satisfy your requirements. Argument can be passed as a string, read from stdin by passing '-', or from a filename if prefixed with '@', e.g. '–advanced-query @query.json'. WARNING: Using advanced queries is incompatible with other query-building arguments.

-c, --use-checkpoint <use_checkpoint>
Use a checkpoint with the given name to only get file events that were not previously retrieved.If a checkpoint for file events with the given name doesn't exist, it will be created on the first run.Subsequent CLI runs with this flag and the same name will use the stored checkpoint to modify the search query and then update the stored checkpoint

--columns <columns>
Filter output to include only specified columns. Accepts comma-separated list of column names (case-insensitive).

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

--ignore-cert-validation
Set to skip CA certificate validation. Incompatible with the 'certs' option.

--certs <certs>
A CA certificates-chain file for the TCP-TLS protocol.

-p, --protocol <protocol>
Protocol used to send logs to server. Use TCP-TLS for additional security. Defaults to UDP.

Options TCP | UDP | TLS-TCP

-f, --format <format>
The output format of the result. Defaults to RAW-JSON format.

Options CEF | JSON | RAW-JSON

Arguments

HOSTNAME
Required argument

2.9 trusted-activities

Manage trusted activities and resources.

```
trusted-activities [OPTIONS] COMMAND [ARGS]...
```

2.9.1 bulk

Tools for executing bulk trusted activity actions.

```
trusted-activities bulk [OPTIONS] COMMAND [ARGS]...
```

create

Bulk create trusted activities using a CSV file with format: type,value,description.

Available *type* values are: DOMAIN|SLACK

```
trusted-activities bulk create [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk trusted-activities commands

```
trusted-activities bulk generate-template [OPTIONS] {create|update|remove}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

remove

Bulk remove trusted activities using a CSV file with format: resource_id.

```
trusted-activities bulk remove [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

update

Bulk update trusted activities using a CSV file with format: resource_id,value,description.

```
trusted-activities bulk update [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.9.2 create

Create a trusted activity.

VALUE is the name of the domain or Slack workspace.

```
trusted-activities create [OPTIONS] {DOMAIN|SLACK} VALUE
```

Options

--description <description>
The description of the trusted activity.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

TYPE
Required argument

VALUE
Required argument

2.9.3 list

List all trusted activities.

```
trusted-activities list [OPTIONS]
```

Options

--type <type>

> **Options** DOMAIN | SLACK

-f, --format <format>
The output format of the result. Defaults to table format.

> **Options** TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.9.4 remove

Remove a trusted activity. Requires the activity's resource ID.

```
trusted-activities remove [OPTIONS] RESOURCE_ID
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

RESOURCE_ID
Required argument

2.9.5 update

Update a trusted activity. Requires the activity's resource ID.

```
trusted-activities update [OPTIONS] RESOURCE_ID
```

Options

--value <value>
The value of the trusted activity, such as the domain or Slack workspace name.

--description <description>
The description of the trusted activity.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

RESOURCE_ID
Required argument

2.10 users

Manage users within your Code42 environment.

```
users [OPTIONS] COMMAND [ARGS]...
```

2.10.1 add-alias

Add a cloud alias for a given user.

A cloud alias is the username an employee uses to access cloud services such as Google Drive or Box. Adding a cloud alias allows Incydr to link a user's cloud activity with their Code42 username. Each user has a default cloud alias of their Code42 username. You can add one additional alias.

```
users add-alias [OPTIONS] USERNAME ALIAS
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

ALIAS
Required argument

2.10.2 add-role

Add the specified role to the user with the specified username.

```
users add-role [OPTIONS]
```

Options

--username <username>
Username of the target user.

--role-name <role_name>
Name of role to add.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.10.3 bulk

Tools for managing users in bulk.

```
users bulk [OPTIONS] COMMAND [ARGS]...
```

add-alias

Add aliases to a list of users from the provided CSV in format: username,alias.

A cloud alias is the username an employee uses to access cloud services such as Google Drive or Box. Adding a cloud alias allows Incydr to link a user's cloud activity with their Code42 username. Each user has a default cloud alias of their Code42 username. You can add one additional alias.

```
users bulk add-alias [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

add-roles

Add roles to a list of users from the provided CSV in format: username,role_name

```
users bulk add-roles [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

deactivate

Deactivate a list of users from the provided CSV in format: username

```
users bulk deactivate [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk user commands.

```
users bulk generate-template [OPTIONS] {update|move|add-alias|remove-
                             alias|update-risk-profile}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

move

Change the organization of the list of users from the provided CSV in format: username,org_id

```
users bulk move [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

reactivate

Reactivate a list of users from the provided CSV in format: username

```
users bulk reactivate [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

remove-alias

Remove aliases from a list of users from the provided CSV in format: username,alias

```
users bulk remove-alias [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

remove-roles

Remove roles from a list of users from the provided CSV in format: username,role_name

```
users bulk remove-roles [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

update

Update a list of users from the provided CSV in format: user_id,username,email,password,first_name,last_name,notes,archive_size_quota

```
users bulk update [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

update-risk-profile

Update user risk profile data from the provided CSV in format: username,start_date,end_date,notes

To clear a value, set column item to the string: 'null'.

```
users bulk update-risk-profile [OPTIONS] CSV_FILE
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

--append-notes
Append provided note value to already existing note on a new line. Defaults to overwrite.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.10.4 deactivate

Deactivate a user.

```
users deactivate [OPTIONS] USERNAME
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.10.5 list

List users in your Code42 environment.

```
users list [OPTIONS]
```

Options

--org-uid <org_uid>
Limit users to only those in the organization you specify. Note that child orgs are included.

--role-name <role_name>
Limit results to only users having the specified role.

--active
Limits results to only active users.

--inactive
Limits results to only deactivated users.

--include-legal-hold-membership
Include legal hold membership in output.

--include-roles
Include user roles.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.10.6 list-aliases

List the cloud aliases for a given user.

Each user has a default cloud alias of their Code42 username with up to one additional alias.

```
users list-aliases [OPTIONS] USERNAME
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.10.7 list-risk-profiles

List users in your Code42 environment.

```
users list-risk-profiles [OPTIONS]
```

Options

--active
Limits results to only active users.

--inactive
Limits results to only deactivated users.

--manager-id <manager_id>
Matches users whose manager has the given Code42 user ID.

--department <department>
Matches users in the given department.

--employment-type <employment_type>
Matches users with the given employment type.

-r, --region <region>
Matches users the given region (state).

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.10.8 move

Change the organization of the user with the given username to the org with the given org UID.

```
users move [OPTIONS]
```

Options

--username <username>
Required The username of the user to move.

--org-id <org_id>
Required The unique identifier (UID) for the organization to which the user will be moved.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.10.9 orgs

Tools for viewing user orgs.

```
users orgs [OPTIONS] COMMAND [ARGS]...
```

list

List all orgs.

```
users orgs list [OPTIONS]
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

show

Show org details.

```
users orgs show [OPTIONS] ORG_UID
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

ORG_UID
Required argument

2.10.10 reactivate

Reactivate a user.

```
users reactivate [OPTIONS] USERNAME
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.10.11 remove-alias

Remove a cloud alias for a given user.

```
users remove-alias [OPTIONS] USERNAME ALIAS
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

ALIAS
Required argument

2.10.12 remove-role

Remove the specified role to the user with the specified username.

```
users remove-role [OPTIONS]
```

Options

--role-name <role_name>
Name of role to remove.

--username <username>
Username of the target user.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.10.13 show

Show user details.

```
users show [OPTIONS] USERNAME
```

Options

--include-legal-hold-membership
Include legal hold membership in output.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.10.14 show-risk-profile

Show user risk profile details.

```
users show-risk-profile [OPTIONS] USERNAME
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.10.15 update

Update a user with the specified unique identifier.

```
users update [OPTIONS]
```

Options

--user-id <user_id>
Required The unique identifier of the user to be modified.

--username <username>
The new username for the user.

--password <password>
The new password for the user.

--email <email>
The new email for the user.

--first-name <first_name>
The new first name for the user.

--last-name <last_name>
The new last name for the user.

--notes <notes>
Notes about this user.

--archive-size-quota <archive_size_quota>
The total size (in bytes) allowed for this user.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.10.16 update-departure-date

Sets the *end_date* on a User's risk profile (useful for users on the Departing Watchlist). Date format: %Y-%m-%d

```
users update-departure-date [OPTIONS] USERNAME [[%Y-%m-%d]]
```

Options

--clear
Clears the current *end_date* value.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

DATE
Optional argument

2.10.17 update-risk-profile-notes

Sets the *notes* value of a User's risk profile.

WARNING: Overwrites any existing note value.

```
users update-risk-profile-notes [OPTIONS] USERNAME [NOTE]
```

Options

--clear
Clears the current *notes* value.

--append
Appends provided note to existing note text as a new line.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

NOTE
Optional argument

2.10.18 update-start-date

Sets the *start_date* on a User's risk profile (useful for users on the New Hire Watchlist). Date format: %Y-%m-%d

```
users update-start-date [OPTIONS] USERNAME DATE
```

Options

--clear
Clears the current *start_date* value.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

DATE
Optional argument

2.11 watchlists

Manage watchlist user memberships.

```
watchlists [OPTIONS] COMMAND [ARGS]...
```

2.11.1 add

Add a user to a watchlist.

```
watchlists add [OPTIONS] [USER_ID|USERNAME]
```

Options

--watchlist-id <watchlist_id>
ID of the watchlist.

--watchlist-type <watchlist_type>
Type of watchlist to add user to.

Options CONTRACT_EMPLOYEE | CUSTOM | DEPARTING_EMPLOYEE | ELEVATED_ACCESS_PRIVILEGES | FLIGHT_RISK | HIGH_IMPACT_EMPLOYEE | NEW_EMPLOYEE | PERFORMANCE_CONCERNS | POOR_SECURITY_PRACTICES | SUSPICIOUS_SYSTEM_ACTIVITY

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

[USER_ID|USERNAME]
Required argument

2.11.2 bulk

Tools for executing bulk watchlist actions.

```
watchlists bulk [OPTIONS] COMMAND [ARGS]...
```

add

Bulk add users to watchlists using a CSV file. Requires either a *watchlist_id* or *watchlist_type* column header to identify the watchlist, and either a *user_id* or *username* column header to identify the user to add.

```
watchlists bulk add [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk adding/removing users.

```
watchlists bulk generate-template [OPTIONS] {add|remove}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

remove

Bulk remove users from watchlists using a CSV file. Requires either a *watchlist_id* or *watchlist_type* column header to identify the watchlist, and either a *user_id* or *username* header to identify the user to remove.

```
watchlists bulk remove [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.11.3 list

List all watchlists.

```
watchlists list [OPTIONS]
```

Options

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.11.4 list-members

List all members on a given watchlist.

```
watchlists list-members [OPTIONS]
```

Options

--watchlist-id <watchlist_id>
ID of the watchlist.

--watchlist-type <watchlist_type>
Type of watchlist to list.

Options CONTRACT_EMPLOYEE | CUSTOM | DEPARTING_EMPLOYEE | ELEVATED_ACCESS_PRIVILEGES | FLIGHT_RISK | HIGH_IMPACT_EMPLOYEE | NEW_EMPLOYEE | PERFORMANCE_CONCERNS | POOR_SECURITY_PRACTICES | SUSPICIOUS_SYSTEM_ACTIVITY

--only-included-users
Restrict results to users explicitly added to watchlist via API or Console. Users added implicitly via group membership or other dynamic rule will not be listed.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

2.11.5 remove

Remove a user from a watchlist.

```
watchlists remove [OPTIONS] [USER_ID|USERNAME]
```

Options

--watchlist-id <watchlist_id>
ID of the watchlist.

--watchlist-type <watchlist_type>
Type of watchlist to remove user from.

Options CONTRACT_EMPLOYEE | CUSTOM | DEPARTING_EMPLOYEE | ELEVATED_ACCESS_PRIVILEGES | FLIGHT_RISK | HIGH_IMPACT_EMPLOYEE | NEW_EMPLOYEE | PERFORMANCE_CONCERNS | POOR_SECURITY_PRACTICES | SUSPICIOUS_SYSTEM_ACTIVITY

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

[USER_ID|USERNAME]
Required argument

2.12 departing-employee

(DEPRECATED): Use *code42 watchlists* commands instead.

Add and remove employees from the Departing Employees detection list.

```
departing-employee [OPTIONS] COMMAND [ARGS]...
```

2.12.1 add

(DEPRECATED): Use *code42 watchlists* commands instead.

Add a user to the Departing Employees detection list.

```
departing-employee add [OPTIONS] USERNAME
```

Options

--departure-date <departure_date>
The date the employee is departing. Format: yyyy-MM-dd.

--cloud-alias <cloud_alias>
If the employee has an email alias other than their Code42 username that they use for cloud services such as Google Drive, OneDrive, or Box, add and monitor the alias. WARNING: Adding a cloud alias will override any existing cloud alias for this user.

--notes <notes>
Optional notes about the employee.

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.12.2 bulk

(DEPRECATED): Use *code42 watchlists* commands instead.

Tools for executing bulk departing employee actions.

```
departing-employee bulk [OPTIONS] COMMAND [ARGS]...
```

add

(DEPRECATED): Use *code42 watchlists* commands instead.

Bulk add users to the departing employees detection list using a CSV file with format: username,cloud_alias,departure_date,notes.

```
departing-employee bulk add [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk adding/removing users.

```
departing-employee bulk generate-template [OPTIONS] {add|remove}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

remove

(DEPRECATED): Use *code42 watchlists* commands instead.

Bulk remove users from the departing employees detection list using a CSV file with format username.

```
departing-employee bulk remove [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.12.3 list

(DEPRECATED): Use *code42 watchlists* commands instead.

Lists the users on the Departing Employees list.

```
departing-employee list [OPTIONS]
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

--filter <filter>
Departing employee filter options. Defaults to ALL.

Options EXFILTRATION_24_HOURS | EXFILTRATION_30_DAYS | LEAVING_TODAY | ALL

2.12.4 remove

(DEPRECATED): Use *code42 watchlists* commands instead.

Remove a user from the Departing Employees detection list.

```
departing-employee remove [OPTIONS] USERNAME
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.13 high-risk-employee

(DEPRECATED): Use *code42 watchlists* commands instead.

Add and remove employees from the High Risk Employees detection list.

```
high-risk-employee [OPTIONS] COMMAND [ARGS]...
```

2.13.1 add

(DEPRECATED): Use *code42 watchlists* commands instead.

Add a user to the high risk employees detection list.

```
high-risk-employee add [OPTIONS] USERNAME
```

Options

--cloud-alias <cloud_alias>
If the employee has an email alias other than their Code42 username that they use for cloud services such as Google Drive, OneDrive, or Box, add and monitor the alias. WARNING: Adding a cloud alias will override any existing cloud alias for this user.

--notes <notes>
Optional notes about the employee.

-t, --risk-tag <risk_tag>
Risk tags associated with the employee.

Options CONTRACT_EMPLOYEE | ELEVATED_ACCESS_PRIVILEGES | FLIGHT_RISK | HIGH_IMPACT_EMPLOYEE | PERFORMANCE_CONCERNS | POOR_SECURITY_PRACTICES | SUSPICIOUS_SYSTEM_ACTIVITY

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.13.2 add-risk-tags

(DEPRECATED): Use *code42 watchlists* commands instead.

Associates risk tags with a user.

```
high-risk-employee add-risk-tags [OPTIONS] USERNAME
```

Options

-t, --risk-tag <risk_tag>
Risk tags associated with the employee.

Options CONTRACT_EMPLOYEE | ELEVATED_ACCESS_PRIVILEGES | FLIGHT_RISK | HIGH_IMPACT_EMPLOYEE | PERFORMANCE_CONCERNS | POOR_SECURITY_PRACTICES | SUSPICIOUS_SYSTEM_ACTIVITY

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.13.3 bulk

(DEPRECATED): Use *code42 watchlists* commands instead.

Tools for executing high risk employee actions in bulk.

```
high-risk-employee bulk [OPTIONS] COMMAND [ARGS]...
```

add

(DEPRECATED): Use *code42 watchlists* commands instead.

Bulk add users to the high risk employees detection list using a CSV file with format: username,cloud_alias,risk_tag,notes.

```
high-risk-employee bulk add [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

add-risk-tags

(DEPRECATED): Use *code42 watchlists* commands instead.

Adds risk tags to users in bulk using a CSV file with format: username,tag.

```
high-risk-employee bulk add-risk-tags [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

generate-template

Generate the CSV template needed for bulk adding/removing users.

```
high-risk-employee bulk generate-template [OPTIONS] {add|remove|add-risk-
                                          tags|remove-risk-tags}
```

Options

-p, --path <path>
Write template file to specific file path/name.

Arguments

CMD
Required argument

remove

(DEPRECATED): Use *code42 watchlists* commands instead.

Bulk remove users from the high risk employees detection list using a CSV file with format username.

```
high-risk-employee bulk remove [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

remove-risk-tags

(DEPRECATED): Use *code42 watchlists* commands instead.

Removes risk tags from users in bulk using a CSV file with format: username,tag.

```
high-risk-employee bulk remove-risk-tags [OPTIONS] CSV_FILE
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

CSV_FILE
Required argument

2.13.4 list

(DEPRECATED): Use *code42 watchlists* commands instead.

Lists the employees on the High Risk Employee list.

```
high-risk-employee list [OPTIONS]
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

-f, --format <format>
The output format of the result. Defaults to table format.

Options TABLE | CSV | JSON | RAW-JSON

--filter <filter>
High risk employee filter options. Defaults to ALL.

Options EXFILTRATION_24_HOURS | EXFILTRATION_30_DAYS | ALL

2.13.5 remove

(DEPRECATED): Use *code42 watchlists* commands instead.

Remove a user from the high risk employees detection list.

```
high-risk-employee remove [OPTIONS] USERNAME
```

Options

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

2.13.6 remove-risk-tags

(DEPRECATED): Use *code42 watchlists* commands instead.

Disassociates risk tags from a user.

```
high-risk-employee remove-risk-tags [OPTIONS] USERNAME
```

Options

-t, --risk-tag <risk_tag>
Risk tags associated with the employee.

Options CONTRACT_EMPLOYEE | ELEVATED_ACCESS_PRIVILEGES | FLIGHT_RISK | HIGH_IMPACT_EMPLOYEE | PERFORMANCE_CONCERNS | POOR_SECURITY_PRACTICES | SUSPICIOUS_SYSTEM_ACTIVITY

-d, --debug
Turn on debug logging.

--totp <totp>
TOTP token for multi-factor authentication.

--profile <profile>
The name of the Code42 CLI profile to use when executing this command.

Arguments

USERNAME
Required argument

- *Alert Rules*
- *Alerts*
- *Audit Logs*
- *Cases*
- *Devices*
- *Legal Hold*
- *Profile*
- *Security Data*
- *Trusted Activities*
- *Users*
- *Watchlists*
- *(DEPRECATED) Departing Employee*

- *(DEPRECATED) High Risk Employee*

The Code42 command-line interface (CLI) tool offers a way to interact with your Code42 environment without using the Code42 console or making API calls directly. For example, you can use it to extract Code42 data for use in a security information and event management (SIEM) tool or manage users on the High Risk Employees list or Departing Employees list.

CHAPTER

THREE

REQUIREMENTS

To use the Code42 CLI, you must have:

- A Code42 product plan that supports the feature or functionality for your use case
- Endpoint monitoring enabled in the Code42 console
- Python version 3.6 and later installed

CHAPTER

FOUR

CONTENT

INDEX

Symbols

A

C

D

H

L

M

N

O

P

R

S

T

U

V

W