Table of Contents

Overview .......................................................................................................................................................... 4

1 API Introduction ......................................................................................................................................... 4

2 Generating API Keys .................................................................................................................................. 5

  2.1 Enterprise .............................................................................................................................................. 6

  2.2 FTK Central .......................................................................................................................................... 8

3 Authentication .............................................................................................................................................. 9

4 Identification .............................................................................................................................................. 9

5 Pagination .................................................................................................................................................. 10

6 Filtering .................................................................................................................................................... 11

7 Cases .......................................................................................................................................................... 13

  7.1 Create a Case ....................................................................................................................................... 13

  7.2 List Cases ............................................................................................................................................ 14

  7.3 Creating a Portable Case .................................................................................................................... 15

8 Evidence ..................................................................................................................................................... 16

  8.1 Process Evidence ............................................................................................................................... 16

  8.2 Browsing Evidence ............................................................................................................................ 17

  8.3 Listing Evidence ............................................................................................................................... 18

  8.4 Listing Processed Evidence ............................................................................................................... 19

9 Labels ......................................................................................................................................................... 20

  9.1 Creating Labels .................................................................................................................................... 20

  9.2 Applying/Removing Labels ............................................................................................................... 21

  9.3 Listing Labels .................................................................................................................................... 22
Overview

Exterro was founded with the simple vision that applying the concepts of process optimization and data science to how companies manage digital information and respond to litigation would drive more successful outcomes at a lower cost. We remain committed to this vision today. We deliver a fully integrated Legal GRC platform that enables our clients to address their privacy, regulatory, compliance, digital forensics, and litigation risks more effectively and at lower costs. We provide software solutions that help some of the world’s largest organizations, law enforcement and government agencies work smarter, more efficiently, and support the Rule of Law.

1 API Introduction

FTK Connect is a RESTful interface with the ability to access functional procedures and objects within AccessData products. Within the API you will find standard REST methodology used throughout and a wide-range of parameters are available allowing for modifying any of the attributes seen within other software’s. Modules exported as endpoints currently provide access to case management, evidence processing and browsing, label management, exporting files; from filters and IDs, in multiple formats, and agent control. All this access allows for a large, varied arrangement of production automations to be implemented simply and quickly.
2 Generating API Keys

The REST interface acts similar to many other interfaces, enforcing an API key to be valid on every sent request. To generate an API key that will be used to authenticate with the Quin-C service, you will require a CodeMeter License dongle with the ADAPI tag. This tag can be seen within the AccessData License Manager inside the Enterprise section of your dongle licenses. An example dongle and the tag can be seen in the License Manager image below:
Two methods are available to generate API keys. One method entails using the Enterprise interface and the other via the web interface (provided the browser has valid cookies cached). See the relevant section below that corresponds to how you wish to generate API keys and this will provide a walkthrough of the procedure.

Notes:

- During this procedure, a user ID or Name will have to be provided. The API key generated will inherit any permissions that the user has available. Thus, typically you will want to create an API key from the Administrator account to provide access to all endpoints.
- The user that is used to logged in can only create API keys that inherit the permissions of the users that this user has permission over.

2.1 Enterprise

When starting the FTK.exe you will be able to generate API keys through the application provided the dongle available to FTK contains the key mentioned previously. The generate API key function can be found by selecting the "Access API Key" option from the Tools menu.
In the new window that is created by selecting this option, you can select the user to generate an API key for. The user selection is used to define the permissions that the bound API key should obtain. Typically, you will want to bind this to the Administrator so that the API key has access to all of the API calls. Then, by clicking "Generate Key", the text box above will populate with the newly generated API key.
2.2 FTK Central

Once the dongle is plugged in and CodeMeter recognizes the device, the Quin-C service will be able to recognize the key. Simply logging into the Quin-C platform as normal will push validated cookies into your web browser, which is how the API key generation authenticates that it should release them. Upon logging into the Quin-C interface, go to the URL bar and we will need to manipulate the URL that Quin-C has redirected you too.

We must replace the "ftkclogin.html" with "api/security/{userid}/getenterpriseapiguid" where the {userid} must be the applicable userid for the account we logged in with. So to generate the Administrator's API key, the userid would be replaced with 1000. An example is shown below:

```
localhost:4443/api/security/1000
localhost:4443/api/security/1000/getenterpriseapiguid
```

This XML file does not appear to have any style information associated with it. The document:

```
<string xmlns="http://schemas.microsoft.com/2003/10/Serialization/">cdb916cd-494a
```
3 Authentication

Each HTTP request must contain authentication for the procedure to be successfully run by the service. The authentication trigger is provided to the service inside the HTTP request through the headers of the request. The headers of the request should include a key named "EnterpriseAPIKey" and this key will define a value of the API key. So the HTTP headers will appear similar to below when correctly applied:

```json
{
    "EnterpriseAPIKey": "7e1bf9de-86ef-40b0-a62d-96de52259939"
}
```

Note: All API keys generated by Enterprise or FTK Central will be produced in the same format as shown above.

4 Identification

Every object within the service contains an identifying integer that the API will use to target specific objects. These identifiers are supplied within the path and body of the HTTP request to denote which object will be targeted. Objects with identifiers include cases, evidence objects, custodians, labels, and jobs. An example of the usage of case and job identifiers in an URI is below:

```
https://localhost:4443/api/v2/enterpriseapi/core/{caseid}/getjobstatus/{jobid}
```
5 Pagination

The Enterprise API makes use of pagination in certain endpoints. Pagination is the method of producing a paging system and determining which page to pass back upon a HTTP request being received. The number of objects per page (pageSize) and the target page are supplied to the API which then translates these values into the objects that will be returned.

Upon the return JSON from the API when using an API call with pagination, the return JSON will contain keys totalCount and entities to provide information on the total amount of objects that are available. Thus, the maximum page is the totalCount/pageSize floored (rounded down).

```json
GET /api/v2/enterpriseapi/core/{caseid}/getobjectlist/5/1

{
  "totalCount": 10000,
  "entities": [
    {},
    {},
    {},
    {},
    {},
    {}  
  ]
}
```
6 Filtering

Filters are available in multiple API calls and are used to restrict a dataset for the procedure. Example places of use can be found in the agent, evidence browsing and export sections. Each filter is comprised of a column type, an attribute to compare against, a comparison type, and values to compare against the attribute. Alongside this, these filters can be combined with AND/OR tools to create more complex filtering options.

The filter JSON will be used by the API to construct an filter in the service for use when obtaining data.

A list of the possible filtering attributes can be obtained by making a GET request to `getallattributes` which will return all attributes and their features. Find documentation about this call in the Utilities section. Examples of file category and file name filters are seen below:

```json
{
    "staticAttributeName": "ObjectFiles_FileCategory",
    "mode": "Includes",
    "values": [0, ] // All Categories
}
```

```json
{
    "staticAttributeName": "ObjectName",
    "operator": "StartsWith",
    "operand": "S",
}
```
Each of the filter types allows for either string or integer input for the operator. The operators available for each filter are shown below:

- **GridColumnFilter Operators**
  - 0 - Includes
  - 1 - Excludes

- **GridColumnComparisonFilter Operators**
  - 0 - GreaterThan
  - 1 - GreaterThanEqualTo
  - 2 - LessThan
  - 3 - LessThanEqualTo
  - 4 - EqualTo
  - 5 - NotEqualTo

- **StringColumnFilter Operators**
  - 0 - EqualTo
  - 1 - NotEqualTo
  - 2 - Contains
  - 3 - StartsWith
  - 4 - EndsWith
7 Cases

7.1 Create a Case

Creates a case within the database supplying the parameters directly as case attributes. The case ID is returned in the request's response for use in parameters for other calls.

**POST /api/v2/enterpriseapi/core/createcase**

**Parameters:**

```json
{
    "name": "string", ## The name to provide the case.
    "description": "string", ## The description to provide the case.
    "comments": "string", ## To add any user comments to the case.
    "attorney": "string", ## Supply an attorney to the case?
    "ftkCaseFolderPath": "string", ## The folder to create to store case data.
    "responsiveFilePath": "string", ## The folder to create to store job data.
    "projectType": "string",## The type of the case.
    "processingMode": int64, ## The mode of processing the case will default to.
    "effectiveEndDate": "string", ## End date for the case to close on.
    "effectiveStartDate": "string", ## Start date for the case to be available.
    "entropyEnabled": false, ## Predefinition for should detect Entropy of files.
    "extractCreditCardNumbers": false, ## Predefinition for Entity Extraction.
    "extractEmailAddresses": false, ## Predefinition for Entity Extraction.
    "extractLocations": false, ## Predefinition for Entity Extraction.
    "extractOrganizations": false, ## Predefinition for Entity Extraction.
    "extractPeople": false, ## Predefinition for Entity Extraction.
    "extractPhoneNumbers": false, ## Predefinition for Entity Extraction.
    "extractSSNs": false, ## Predefinition for Entity Extraction.
}
```

**Response:**

`int64`
7.2 List Cases

Lists all cases available to the user; who created the API key. The response is a singular list of JSON objects each containing a case and its attributes.

**GET /api/v2/enterpriseapi/core/getcaselist**

**Response:**

```json
[
  {
    "name": "string", ## The name to provide the case.
    "description": "string", ## The description to provide the case.
    "comments": "string", ## To add any user comments to the case.
    "attorney": "string", ## Supply an attorney to the case?
    "ftkCaseFolderPath": "string", ## The folder to create to store case data.
    "responsiveFilePath": "string", ## The folder to create to store job data.
    "projectType": "string",## The type of the case.
    "processingMode": int64, ## The mode of processing the case will default to.
    "effectiveEndDate": "string", ## End date for the case to close on.
    "effectiveStartDate": "string", ## Start date for the case to be available.
    "entropyEnabled": false, ## Predefinition for should detect Entropy of files.
    "extractCreditCardNumbers": false, ## Predefinition for Entity Extraction.
    "extractEmailAddresses": false, ## Predefinition for Entity Extraction.
    "extractLocations": false, ## Predefinition for Entity Extraction.
    "extractOrganizations": false, ## Predefinition for Entity Extraction.
    "extractPeople": false, ## Predefinition for Entity Extraction.
    "extractPhoneNumbers": false, ## Predefinition for Entity Extraction.
    "extractSSNs": false, ## Predefinition for Entity Extraction.
  }
]
```
7.3 Creating a Portable Case

Creates a portable case within the output folder specified.

POST /api/v2/enterpriseapi/core/{caseid}/createportablecase

Parameters:

```
[
  {
    "outputpath": "string", ## The path to the portable case output.
    "foldername": "PortableCase", ## The name of the folder the portable case would be stored in.
    "copyqview": true, "filter": {}, ## Copy FTK+ to boot with.
    "filter": {}, ## A JSON filter OR set to none.
    "uifilter": {}, ## A normal review filter.
    "columnsToCopy": [{"attributeUniqueName": "ObjectName", "isColumnSetMember": false, "description": ",
    "displayName": "ObjectName", "dataType": 11, "productType": 64, "isSystemField": true, "isReadOnly":
    true, "isGridEditable": false, "isCustomField": false, "isMappable": true, "isFilterable": true, "isBurnable":
    false, "isAutoTextField": false, "uiLayout": 0, "isDefault": false, "multiLine": null, "maxLength": 1024,
    "isCategory": false, "hasMetaDataValue": false, "isVirtual": false, "isLookup": false, "isViewable": true,
    "columnID": 9001, "isProducible": false, "state": 3, "id": 265, "lastModifiedDate": null, "lastModifiedBy":
    null, "hasSeqID": false, "columnGUID": "00000000-0001-0000-40da-41e02923000", "columnClass": 0 }], ##
    ## Maintain columns for offline view.
  }
]
```
8 Evidence

8.1 Process Evidence

Starts a processing job on the evidence file/directory provided. The processing parameters can be supplied within the JSON as either keyvalues or as an XML string. To indicate the type of evidence that is being provided, the `evidencetype` attribute can be set to either 0, 1, 2, or 3.

The request will return a list of integers corresponding to the job IDs created from the processing parameters that were specified.

### POST /api/v2/enterpriseapi/core/{caseid}/processdata

**Parameters:**

```
{
   "comments": "string", ## Comments to add to evidence object.
   "completeprocessingoptions": "string", ## An XML string of processing options.
   "evidencetocreate": {
      "evidencepath": "string", ## The path to the evidence file or directory.
      "evidencetype": "string", ## The type of evidence provided.
      ## 0 = File, 1 = Directory, 2 = Image, 3 = Auto.
      "nameNumber": "string"## The identification of the evidence.
      "timezone": "string"## The timezone to adjust for.
   },
   "processingmanager": "string", ## The processing manager to use for processing.
}
```

**Response:**

```
int64
```
8.2 Browsing Evidence

Enables the end-user to browse evidence in a "pageview" using Pagination. This call allows the use of filters and columns to restrict and alter the viewable information returned. This will then return a page of objects with the columns supplied in the relevant size provided within the path. The page to target can be supplied within the path as well to iterate through an entire page set.

To determine the total objects and therefore the total pages, the request will always provide the total objects found in the page cycle. Read Pagination for more information.

GET /api/v2/enterpriseapi/core/{caseid}/getobjectlist/{pageSize}/{page}

Parameters:

{
  "columns": [
    {
      "attribute": {
        "attributeUniqueName": "string", ## An object attribute.
        "columnID": int64 ## The object attribute ID.
      },
      "sort": "enum 0-2" ## The type of sort. 0 = None, 1 = Ascending, 2 = Descending
    }
  ],
  "filter": {
    ## A JSON filter.
  }
}

Response:

{
  "totalCount": int64,
  "entities": [] ## A list of the objects with the attributes requested.
}
8.3 Listing Evidence

Lists all evidence objects added to the case. The response is a singular list of JSON objects each containing an evidence object and its attributes.

GET /api/v2/enterpriseapi/core/{caseid}/getevidencelist

Response:

```json
[
{
  "dataSourceName": "string", ## Custodian/Person.
  "evidenceId": 0, ## The internal ID of the evidence object.
  "evidenceName": "string", ## The name of the evidence.
  "evidenceTime": "2021-05-26T09:15:01.120Z", ## The time when evidence was added.
  "evidenceSize": 0, ## Size of the evidence in KB.
  "filePath": "string", ## The path to the evidence file.
  "evidenceType": "string", ## The type of the evidence file.
  "comments": "string" ## Any comments added to the evidence during processing.
},
]
```
8.4 Listing Processed Evidence

Lists all processed evidence objects in a case. This response is a singular list of JSON objects each containing an evidence objects and its attributes.

```
GET /api/v2/enterpriseapi/core/{caseid}/getprocessedevidencelist
```

**Response:**

```
[
{
"dataSourceName": "string", ## Custodian/Person.
"evidenceId": 0, ## The internal ID of the evidence object.
"evidenceName": "string", ## The name of the evidence.
"evidenceTime": "2021-05-26T09:15:01.120Z", ## The time when evidence was added.
"evidenceSize": 0, ## Size of the evidence in KB.
"filePath": "string", ## The path to the evidence file.
"evidenceType": "string", ## The type of the evidence file.
"comments": "string" ## Any comments added to the evidence during processing.
},
]
```
9 Labels

9.1 Creating Labels

Creates a new label within the case using the name and color provided.

The request returns the label ID of the newly created label.

**POST /api/v2/enterpriseapi/core/{caseid}/createlabel**

**Parameters:**

```json
{
    "name": "string",  ## Name of the label.
    "color": "string",  ## The color of the label in HEX.
}
```

**Response:**

`int64`
9.2 Applying/Removing Labels

Labels evidence child objects with the specified label. This call allows the use of filters as to not have to know the target object IDs in advance, performing a "batch" label job.

POST /api/v2/enterpriseapi/jobs/{caseid}/labelobjects

Parameters:

```
{
  "folderAssignmentOptions": {
    "filterForFolder": {
      "columnDefinition": {
        "attribute": {
          "attributeUniqueName": "string", ## An object attribute.
          "columnID": int64 ## The object attribute ID.
        }
      }
    },
    "searchSessionID": None,
    "folderIDsForAssign": [ int64, ] ## The label IDs to assign.
    "folderIDsForUnAssign": [ int64, ], ## The label IDs to un-assign.
  },
  "filter": {
    ## A JSON filter.
  }
}
```

Response:

```
int64
```
9.3 Listing Labels

The request simply returns a list of available labels as objects.

**POST /api/v2/enterpriseapi/core/{caseid}/getlabellist**

Response:

```
[
{
    "name": "string", ## Name of the label.
    "color": "string", ## The color of the label in HEX.
},
]
```

This request simply returns a list of object IDs for any file object labelled by the supplied label.

**GET /api/v2/enterpriseapi/core/{caseid}/label/{labelid}/evidenceObjects**

Response:

```
int64
```
10 Jobs

10.1 Status Retrieval

Polls for the status of a job and retrieves data about the running job.

**GET /api/v2/enterpriseapi/jobs/{caseid}/getjobstatus/{jobid}**

**Response:**

```
{
    "resultData": "string" ## A JSON string of information about the job.
    "state": "string", ## The state of the job.
    "startTime": "string" ## A date and time of the job start.
    "endTime": "string" ## A date and time of the job end. NULL if not ended.
}
```
10.2 Exporting Evidence Objects

Evidence objects can be exported through the API in native format. Using `dumpnativeobject` will provide the access to export files directly from the evidence via a list of file object IDs or a filter.

The request returns the job ID for the export job that had been started.

**POST /api/v2/enterpriseapi/jobs/{caseid}/dumpnativeobjects**

**Parameters:**

```json
{
   "objectIds": [ int64, ], ## A list of file object IDs.
   "dumpUsingObjectId": true, ## Dump files by the object IDs provided?
   "runParser": true, ## Should run message parser if processing?
   "checkProcessedObjectFlag": true, ## Check if object is processed before dump.
   "isInputFolder": true, ## Is the input folder containing evidence?
   "uiFilter": {}, ## A normal review filter. OR
   "bscFilter": { ## A custom filter include search capabilities.
      "columnDefinition": {
         "attribute": {
            "attributeUniqueName": "string", ## A object attribute.
            "columnID": 0 ## The object attribute’s ID.
         },
         "relatedDataMustExist": true ## Check related data?
      },
      "operator": 0, ## Equals, NotEquals, StartsWith, etc
      "values": [], ## Comparison values or single value.
      "isDtSearchFilter": true, ## Is searching dtSearch index?
      "isDatabaseSearchFilter": true, ## Is searching database columns?
   },
   "completeProcessingOptions": "string", ## Processing options in XML string.
   "inputFolder": "string", ## Dump folder or evidence input folder if isInputFolder is set.
   "outputFolder": "string" ## Dump output folder if providing evidence.
}
```

**Response:**

`int64`
10.3 Generating Search Reports

Runs a search process against the dtSearch index to locate the search term provided. This search is capable of all systems available in Enterprise, e.g. Regex, Word Stemming, etc. The positive objects detected by the search may also be labelled during the job and the parameters of which labels to apply is available in the parameters of the call.

The request will return an job ID indicating which job had been started.

**POST /api/v2/enterpriseapi/jobs/{caseid}/createsearchcountreport**

**Parameters:**

```
{
  "assignLabel": true, ## Should assign the label to results?
  "criteria": {
    "terms": [  
      "string" ## A list of terms to search for.
    ],
    "searchParameters": { ## Searching options.
      "fuzziness": 0, ## The amount of fuzziness if "fuzzy" is true.
      "maxItemsToRetrieve": 0, ## Maximum items that should be retrieved.
      "stemming": true, ## Part of a word. e.g. run -> running
      "fuzzy": true, ## Lookalike words. e.g. runs -> runz
      "phonic": true, ## Sounding the same words. piece -> peace
      "synonyms": true, ## Words that mean the same. kill -> murder
      "userSynonyms": true, ## Use user-defined synonyms?
      "wordNetSynonyms": true, ## Use wordNet synonyms?
      "wordNetRelated": true, ## Use wordNet related words?
      "allSynonyms": true, ## Wider range of synonyms.
      "wantHitsByWord": true, ## Export hits from word searches.
      "wantHitsArray": true, ## An array of the hits.
      "regularExpressions": true, ## Enable regular expressions.
      "wantHitDetails": true, ## Further details about the hits.
    }
  }
}
"autoTermWeight": true, ## Weigh the term likeliness.
"applyLanguageAnalyzer": true, ## Perform language analysis.
"languageAnalyzerSynonyms": true, ## Synonyms for other languages.
"includeFamily": true,
"includeRelated": true,
"includeDuplicates": true,
"includeFolders": true,
"includeUnimportantOleStreams": true,
"includeEmailCluster": true,
"includeLinkedDocuments": true
}
},
"fullTextOnlySearch": true, ## Whether is for literal text only.
"name": "string", ## The name of the search report.
"searchTerms": [
{
    "label": "string", ## The label to apply to the corresponding term.
    "term": "string" ## The term to apply the label too.
}
]
}

Response:

int64
11 Agents & Site Server

11.1 Deployment

Pushes an agent to the target machine via the Site Server.
The request returns a job ID referencing the job that has been started.

**POST /api/v2/enterpriseapi/agent/{caseid}/runagentpush**

**Parameters:**

```json
{
    "agentPush": {
        "credentials": [
            {
                "domain": "string", ## The name of the domain.
                "username": "string", ## The username on the domain to deploy the agent.
                "password": "string" ## The password for the user on the domain.
            }
        ],
        "serviceName": "string", ## The name of the service if deployed as service.
        "executableName": "string", ## The name of the executable when deployed.
        "makeAvailableToAllUsers": true, ## Should install for all users?
        "updateAgent": true, ## Should update agent if already existent?
        "port": 0, ## The port to listen on.
        "paPort": 0, ## The username on the domain to deploy the agent.
        "hideAgent": true, ## Should the agent executable hide itself?
        "pushPublicInstanceConfig": true, ## Apply the public config?
        "pushHeartbeatConfig": true, ## Apply the default heartbeat config?
        "installTransient": true, ## Install as transient?
        "serviceLess": true, ## Deploy as a serviceless object but stil run?
        "removeAgent": true, ## Uninstall the agent on these targets?
        "useRemComPush": true, ## To use RemCom, enable this.
    }
}
```
```json
}

"ips": {
    "targets": [
        "string" ## A list of target IP addresses.
    ]
}
}

Response:

int64
```
11.2 Disk Collection

Performs a full disk collection on the targets specified. The evidence gather will be found inside the case folder prior to the collection under the "Jobs" folder.

The request returns a job ID referencing the job that has been started.

**POST /api/v2/enterpriseapi/agent/{caseid}/diskacquistion**

**Parameters:**

```
{
    "driveImage": {
        "imageBaseFilename": "string", ## The filename of the generated image.
        "interruptTimeoutMinutes": 0, ## Minutes before timeout.
        "driveId": 0, ## The ID of the drive to image.
        "driveType": 0, ## The type of the drive, e.g. network, etc.
        "imageType": 0, ## The image type, e.g. AD1, E01, etc.
        "calculateSha1": true, ## Calculate the SHA1 hash?
        "verifyIntegrity": true, ## Should verify upon completion?
        "segmentSizeInMBytes": 0, ## Split the image into segments? if so, size?
        "driveCompression": 0, ## The level of compression? 1-10.
        "caseNumber": "string", ## The case number to add to metadata.
        "evidenceNumber": "string", ## The evidence number to add to metadata.
        "description": "string", ## The description to add to metadata.
        "examiner": "string", ## The examiner name to add to metadata.
        "notes": "string", ## The notes to add to metadata.
        "startSector": 0, ## The sector to start at imaging.
        "endSector": 0 ## The sector to end imaging at.
    },
    "ips": {
        "targets": [
            "string" ## A list of target IP addresses.
        ]
    }
}
```

**Response:**

`int64`
11.3 Memory Collection

Captures a full memory copy from the targets specified. The memory image will be found inside the case folder prior to the collection under the "Jobs" folder. This will provide the option to gather a page file alongside the memory too.

The request returns a job ID referencing the job that has been started.

**POST /api/v2/enterpriseapi/agent/{caseid}/memoryacquistion**

**Parameters:**

```json
{
    "memoryAcquistion": {
        "includePagefile": true, ## Add the page file to the memory collection?
    },
    "createAD1": true ## Wrap the memory image in an AD1?
},
"ips": {
    "targets": [
        "string" ## A list of target IP addresses.
    ]
}
```  

**Response:**

```
int64
```
11.4 Memory Analysis

Performs a analysis of the processes, connections, services running on the operating system using Volatility. The resultant XML files generated from Volatility can be found within the case folder.

The request returns a job ID referencing the job that has been started.

```
POST /api/v2/enterpriseapi/agent/{caseid}/volatile

Parameters:

{
    "volatile": {
        "includeProcessTree": true, ## Analyse processes?
        "processTreeOptions": {
            "detectHiddenProcesses": true, ## Rebuild hidden processes?
            "includeDlls": true, ## Analyse any available DLLs?
            "dllOptions": {
                "detectInjectedDlls": true ## Route to injected DLLs?
            },
            "includeSockets": true, ## Analyse any open sockets?
            "includeHandles": true, ## Analyse handles?
            "includeJamScore": true,
            "includeStaticAnalysis": true
        },
        "includeServices": true, ## Analyse services?
        "includeJamServices": true, ## Analyse JAM services?
        "includeDrivers": true, ## Analyse drivers?
        "includeJamDrivers": true, ## Analyse JAM drivers?
        "includeUsers": true, ## Gather users?
        "includeNICs": true, ## Gather NIC interfaces?
        "includeSMBSessions": true, ## Gather SMB sessions?
        "includeArp": true, ## Analyse ARP tables?
    }
}
```
"includeRouting": true, ## Analyse web routing data?
"includeDNSCache": true, ## Gather DNS cache data?
"includePrefetch": true, ## Gather prefetch files?
"includeVolume": true, ## Analyse available volumes?
"includeUsb": true, ## Analyse USB devices?
"includeLiveRegistry": true, ## Gather Live Registry?
"includeTasks": true, ## Gather Tasks from Scheduler?
"includeJamTasks": true, ## Gather Tasks from JAM Scheduler?
},

"ips": {
  "targets": [
    "string" ## A list of target IP addresses.
  ]
}

### Response:

`int64`

These XML files can then be imported into the case with the following call.

```
GET /api/v2/enterpriseapi/agent/{caseid}/importvolatile/{jobid}
```

### Response

`int64`
11.5 Remediation

Performs a remediation task on the targets specified. Remediation tasks are supplied within a list. Each object corresponds to a single task to be run. The types of remediation tasks are documented below.

The request returns a job ID referencing the task that has been started.

```
POST /api/v2/enterpriseapi/agent/{caseid}/remediate
```

**Parameters:**

```json
{
    "agentRemediation": [  
    ],  ## A remediation task object.
    "ips": {  
    "targets": [  
    "string" ## A list of target IP addresses.
    ]
    }
}
```

**Response:**

`int64`

Available remediation tasks are as follows:

- **Disable Service**

```json
{
    "commandType": "DisableService",
    "serviceName": "AccessData Agent" ## The name of the service to disable.
}
```
• **Deploy and Execute Script**

```json
{
    "commandType": "ExecuteSupplied",
    "fileToExecute": "C:\server\C$\Scripts\toRun.ps1", ## The path of the script to deploy.
    "workingDirectory": "C:\" ## The directory to run script from.
}
```

• **Erase File**

```json
{
    "commandType": "EraseFile",
    "remoteFilePath": "C:\Badfile.txt" ## The path of the file to delete.
}
```

• **Execute Script**

```json
{
    "commandType": "ExecuteRemote",
    "remoteFilePath": "C:\Scripts\toRun.ps1", ## The path of the script to run.
    "workingDirectory": "C:\" ## The directory to run script from.
}
```

• **Kill Process**

```json
{
    "commandType": "KillProcessByName",
    "processName": "notepad.exe" ## The name of the process to kill.
}
```

• **Replace File**

```json
{
    "commandType": "ReplaceFile",
    "remoteFilePath": "C:\Badfile.txt" ## The path of the file to replace.
    "replacementFile": "C:\Goodfile.txt" ## The path of the file to copy.
}
```
• **Restart Agent**

```
{
   "commandType": "RebootAgent"
}
```

• **Start Service**

```
{
   "commandType": "StartService",
   "serviceName": "AccessData Agent" ## The name of the service to start.
}
```

• **Stop Service**

```
{
   "commandType": "StopService",
   "serviceName": "AccessData Agent" ## The name of the service to stop.
}
```
11.6 Software Inventory

Runs an inventory on the software installed on the target system/s.

The request returns a job ID referencing the job that has been started.

```
POST /api/v2/enterpriseapi/agent/{caseid}/softwareinventory
```

**Parameters:**

```
{
    "ips": {
        "targets": [
            "string" ## A list of target IP addresses.
        ]
    }
}
```

**Response:**

```
int64
```
# Utilities

## 12.1 Status Check

Checks the status of the API service.

The request returns a string containing the status.

**GET /api/v2/enterpriseapi/statuscheck**

**Response:**

```
"string"
```

## 12.2 Site Server Status Check

Checks the status of the Site Server.

The request returns a string containing the status.

**GET /api/v2/enterpriseapi/agent/getsiteserverstatus**

**Response:**

```
"string"
```
12.3 Expand Child Categories

Gathers all child categories for the parent category ID number provided.

**Note:** Each category and file type is represented by a enum value. Thus, this will expand a enum value of a parent into the constituent enum values of all children.

The request returns a list of integers for the expanded categories.

**POST /api/v2/enterpriseapi/core/getchildrenfilecategories**

**Parameters:**

\[ int64, \]\# A list of parent categories to expand.

**Response:**

int64
12.4 Object Attributes

Gathers all possible attributes that a file object may have.

The request returns a list of objects each containing an attribute and its type information.

GET /api/v2/enterpriseapi/core/getallattributes

Response:

```json
[
  {
    "attributeUniqueName": "string",
    "isColumnSetMember": true,
    "description": "string",
    "displayName": "string",
    "dataType": 0,
    "productType": 0,
    "isSystemField": true,
    "isReadOnly": true,
    "isGridEditable": true,
    "isCustomField": true,
    "isMappable": true,
    "isFilterable": true,
    "isBurnable": true,
    "isAutoTextField": true,
    "uiLayout": 0,
    "isDefault": true,
    "multiLine": true,
    "maxLength": 0,
    "isCategory": true,
    "hasMetaDataValue": true,
    "isVirtual": true,
    "isLookup": true,
  }
]```
12.5 Check Processing

Recognizes any case that is currently processing evidence and returns its ID.

**GET /api/v2/enterpriseapi/processingcaseid**

**Response:**

`int64`
13FTK Connect

13.1 Triggering API Workflows

Triggers a FTK Connect workflow by calling the workflow ID.

The request returns a job ID referencing the job that has been started.

**POST /api/v2/enterpriseapi/workflow/triggerworkflow/{workflowid=workflowid}**

**Parameters:**

```
{
  "ips": {
    "targets": [   "string" A list of target IP addresses.
  ]
}
```

**Response:**

`Boolean`
Contact Exterro

If you have any questions, please refer to this document, or any other related materials provided to you by Exterro. For usage questions, please check with your organization’s internal application administrator. Alternatively, you may contact your Exterro Training Manager or other Exterro account contact directly.

For technical difficulties, support is available through support@exterro.com.

Contact:
Exterro, Inc.
4145 SW Watson Ave., Suite 400
Beaverton, OR 97005.
Telephone: 503-501-5100
Toll Free: 1-877-EXTERRO (1-877-398-3776)
Fax: 1-866-408-7310
General E-mail: info@exterro.com
Website: www.exterro.com

Information in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Exterro, Inc. The trademarks, service marks, logos or other intellectual property rights of Exterro, Inc and others used in this documentation (“Trademarks”) are the property of Exterro, Inc and their respective owners. The furnishing of this document does not give you license to these patents, trademarks, copyrights or other intellectual property except as expressly provided in any written agreement from Exterro, Inc.

The United States export control laws and regulations, including the Export Administration Regulations of the U.S. Department of Commerce, and other applicable laws and regulations apply to this documentation which prohibits the export or re-export of content, products, services, and technology to certain countries and persons. You agree to comply with all export laws, regulations and restrictions of the United States and any foreign agency or authority and assume sole responsibility for any such unauthorized exportation.

You may not use this documentation if you are a competitor of Exterro, Inc, except with Exterro Inc’s prior written consent. In addition, you may not use the