# Goals

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 OASIS CTI TC</td>
<td>3</td>
</tr>
<tr>
<td>1.2 CTI TAXII Client</td>
<td>5</td>
</tr>
<tr>
<td>1.3 CTI Python STIX</td>
<td>9</td>
</tr>
<tr>
<td>1.4 ATTACK CTI Overview</td>
<td>14</td>
</tr>
<tr>
<td>1.5 Library Functions</td>
<td>17</td>
</tr>
<tr>
<td>1.6 Licenses</td>
<td>50</td>
</tr>
</tbody>
</table>
A Python module to access up to date ATT&CK content available in STIX format via its public TAXII server. This project leverages the python classes and functions of the cti-python-stix2 and cti-taxii-client libraries developed by MITRE.
CHAPTER 1

Goals

• Provide an easy way to access and interact with up to date ATT&CK content available in STIX via public TAXII server
• Allow security analysts to quickly explore ATT&CK content and apply it in their daily operations
• Allow the integration of ATT&CK content with other platforms to host up to date information from the framework
• Help security analysts during the transition from the ATT&CK MediaWiki API to the STIX/TAXII 2.0 API
• Learn STIX2 and TAXII Client Python libraries

1.1 OASIS CTI TC

The OASIS Cyber Threat Intelligence (CTI) TC (Technical Committee) was chartered to define a set of information representations and protocols to address the need to model, analyze, and share cyber threat intelligence. In the initial phase of TC work, three specifications will be transitioned from the US Department of Homeland Security (DHS) for development and standardization under the OASIS open standards process: STIX (Structured Threat Information Expression), TAXII (Trusted Automated Exchange of Indicator Information), and CybOX (Cyber Observable Expression).

The OASIS CTI Technical Committee will:

• Define composable information sharing services for peer-to-peer, hub-and-spoke, and source subscriber threat intelligence sharing models
• Develop standardized representations for campaigns, threat actors, incidents, tactics techniques and procedures (TTPs), indicators, exploit targets, observables, and courses of action
• Develop formal models that allow organizations to develop their own standards-based sharing architectures to meet specific needs
1.1.1 OASIS CTI & MITRE ATT&CK

On May 14th, 2018, the ATT&CK team announced that all of MITRE’s Adversarial Tactics, Techniques, and Common Knowledge content, including ATT&CK for Enterprise, PRE-ATT&CK™, and ATT&CK for Mobile, was going to be available via their own TAXII 2.0 server in STIX 2.0 format. This move to STIX and TAXII was a great effort by the ATT&CK team to facilitate the use of the framework in a more programmatical way and allow the integration of it with several other applications. In order to interact with the TAXII server and handle the STIX content, MITRE created the cti-taxii-client and cti-python-stix2 libraries and released them as part of the open repositories of the OASIS Technical Committee for Cyber Threat Intelligence.

1.1.2 OASIS TC Open Repositories

An OASIS TC Open Repository is a public GitHub repository supporting the activities of an associated OASIS Technical Committee. TC Open Repository contents are created through public contributions under a designated open source license, and community participants establish development priorities for assets maintained in the repository.

Repositories

- cti-documentation: GitHub Pages site for STIX and TAXII
- cti-marking-prototype: Prototype for processing granular data markings in STIX
- cti-pattern.matcher: Match STIX content against STIX patterns
- cti-pattern-validator: Validate patterns used to express Cyber Observable content in STIX Indicators
- cti-python-stix2: Python APIs for STIX 2
- cti-stix-elevator: Convert STIX 1.2 XML to STIX 2.0 JSON
- cti-stix-slider: Convert STIX 2.0 JSON to STIX 1.2 XML
- cti-stix-validator: Validator for STIX 2.0 JSON normative requirements and best practices
- cti-stix2-json-schemas: Non-normative STIX and Cyber Observable schemas and examples
- cti-taxii-client: TAXII 2 Client Library Written in Python
- cti-taxii-server: TAXII 2 Server Library Written in Python
- cti-training: Collection of CTI-related training materials

References

- https://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/attck%E2%84%A2-content-available-in-stix%E2%84%A2-20-via
- https://oasis-open.github.io/cti-documentation/resources.html#taxii-20-specification
1.2 CTI TAXII Client

The cti-taxii-client library was developed by MITRE and it is a minimal client implementation for the TAXII 2.0 server. A TAXII server is an open-source module designed to serve STIX 2.0 content in compliance with the TAXII 2.0 specification. Written in JavaScript, a TAXII server takes advantage of Node.js’s asynchronous I/O model to handle incoming connections, allowing the server to handle connections smoothly under load.

1.2.1 What is TAXII?

Trusted Automated Exchange of Intelligence Information (TAXII™) is an application protocol for exchanging CTI over HTTPS. TAXII defines a RESTful API (a set of services and message exchanges) and a set of requirements for TAXII Clients and Servers. As depicted below, TAXII defines two primary services to support a variety of common sharing models:

- **Collection** - A Collection is an interface to a logical repository of CTI objects provided by a TAXII Server that allows a producer to host a set of CTI data that can be requested by consumers: TAXII Clients and Servers exchange information in a request-response model.

- **Channel** - Maintained by a TAXII Server, a Channel allows producers to push data to many consumers and consumers to receive data from many producers: TAXII Clients exchange information with other TAXII Clients in a publish-subscribe model. Note: The TAXII 2.0 specification reserves the keywords required for Channels but does not specify Channel services. Channels and their services will be defined in a later version of TAXII.

1.2.2 TAXII Client & ATT&CK

As mentioned before, on May 14th, 2018, the ATT&CK team announced that all of MITRE’s Adversarial Tactics, Techniques, and Common Knowledge content, including ATT&CK for Enterprise, PRE-ATT&CK™, and ATT&CK for Mobile, was going to be available via their own TAXII 2.0 server in STIX 2.0 format. The following four classes are available via the taxii2-client library and can be used to interact with ATT&CK’s public TAXII server:

- Server
- ApiRoot
- Collection
- Status

1.2.3 Query ATT&CK

ATT&CK users can use the initial `Server` class to instantiate a server object pointing to the framework’s public TAXII server URL `https://cti-taxii.mitre.org/taxii/`

```python
>>> from taxii2client import Server

>>> server = Server("https://cti-taxii.mitre.org/taxii/")
```

The server variable can then be used to access initial metadata about the ATT&CK TAXII server:

```python
>>> server.title
'CTI TAXII server'

>>> server.description
'This TAXII server contains a listing of ATT&CK domain collections expressed as STIX, including PRE-ATT&CK, ATT&CK for Enterprise, and ATT&CK Mobile.'
```

(continues on next page)
In addition, available API Roots can be referenced from the server object. API Roots are logical groupings of TAXII Channels and Collections and can be thought of as instances of the TAXII API available at different URLs, where each API Root is the “root” URL of that particular instance of the TAXII API:

```python
>>> server.api_roots
[<taxii2client.ApiRoot object at 0x10519e7b8>]
>>> api_root = server.api_roots[0]
<taxii2client.ApiRoot object at 0x10519e7b8>
```

As we can see above, there is only one API root instance available, and information about it can be accessed the following way:

```python
>>> api_root.title
'stix'

>>> api_root.url
'https://cti-taxii.mitre.org/stix/'

api_root.versions
['taxii-2.0']
```

If you explore the additional attributes and methods available in the only api root instance, there is a collections attribute:

```python
>>> api_root.
api_root.close( api_root.custom_properties api_root.get_status( api_root.refresh( api_root.refresh_information( api_root.url api_root.collections api_root.description api_root.max_content_length api_root.refresh_collections( api_root.title api_root.versions

The collections attribute can then be used and get more information about them via their respective available properties:

```python
>>> api_root.collections
[<taxii2client.Collection object at 0x105ba1dd8>, <taxii2client.Collection object at 0x105b855f8>, <taxii2client.Collection object at 0x105b85908>]

>>> api_root.collections[0]
<taxii2client.Collection object at 0x105ba1dd8>

>>> api_root.collections[0].title
'Enterprise ATT&CK'

>>> api_root.collections[0].id
'95ecc380-afe9-11e4-9b6c-751b66dd541e'

>>> api_root.collections[0].description
```
'This data collection holds STIX objects from Enterprise ATT&CK'

```python
global api_root

global api_root

global api_root

global api_root

A for loop can be used to print all the collections available in the ATT&CK public TAXII server with their respective names and ids. As we can see below, there are three collections available in the TAXII server, and they are mapped to ATT&CK domains:

```python
for collection in api_root.collections:
    print(collection.title + ': ' + collection.id)

Enterprise ATT&CK: 95ecc380-afe9-11e4-9b6c-751b66dd541e
PRE-ATT&CK: 062767bd-02d2-4b72-84ba-56caef0f8658
Mobile ATT&CK: 2f669986-b40b-4423-b720-4396ca6a462b
```

We can then use the `Collection` class to instantiate TAXII2 Collection objects for each available collection:

```python
from taxii2client import Collection

ENTERPRISE_COLLECTION = Collection(api_root.collections[0].url)
PRE_COLLECTION = Collection(api_root.collections[1].url)
MOBILE_COLLECTION = Collection(api_root.collections[2].url)
```

Finally we can use the `get_object` method from the `Collection` class and retrieve a specific object from the ATT&CK Enterprise Matrix. Let's say we want to retrieve technique 1066. We will need to provide the object id `attack-pattern--00d0b012-8a03-410e-95de-5826bf542de6` that corresponds to T1066. You can use the MITRE cti GitHub repo to confirm the technique-id mapping:

```python
T1066 = ENTERPRISE_COLLECTION.get_object("attack-pattern--00d0b012-8a03-410e-95de-5826bf542de6")

T1066
```

1.2. CTI TAXII Client
As you can see above, we were able to get information about a specific technique from the ATT&CK public TAXII server. However, it would be good to filter our collection request by specific STIX objects without relying on an object ID only. This is where the next library cti-python-stix2 comes into play.

1.2.4 References

- https://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/attck%E2%84%A2-content-available-in-stix%E2%84%A2-20-via
- https://github.com/oasis-open/cti-taxii-client
- https://oasis-open.github.io/cti-documentation/taxii/intro.html
- https://github.com/mitre/cti/blob/master/enterprise-attack/attack-pattern/attack-pattern–00d0b012-8a03-410e-95de-5826bf542de6.json
1.3 CTI Python STIX

The cti-python-stix2 library was developed by MITRE to help users serialize/de-serialize, produce, consume, and process STIX 2 content.

1.3.1 What is STIX?

Structured Threat Information Expression (STIX™) is a language and serialization format used to exchange cyber threat intelligence (CTI).

1.3.2 STIX Objects

STIX Objects categorize each piece of information with specific attributes to be populated. Chaining multiple objects together through relationships allows for easy or complex representations of CTI.

- **Attack Pattern**: A type of Tactics, Techniques, and Procedures (TTP) that describes ways threat actors attempt to compromise targets.
- **Campaign**: A grouping of adversarial behaviors that describes a set of malicious activities or attacks that occur over a period of time against a specific set of targets.
- **Course of Action**: An action taken to either prevent an attack or respond to an attack.
- **Identity**: Individuals, organizations, or groups, as well as classes of individuals, organizations, or groups.
- **Indicator**: Contains a pattern that can be used to detect suspicious or malicious cyber activity.
- **Intrusion Set**: A grouped set of adversarial behaviors and resources with common properties believed to be orchestrated by a single threat actor.
- **Malware**: A type of TTP, also known as malicious code and malicious software, used to compromise the confidentiality, integrity, or availability of a victim’s data or system.
- **Observed Data**: Conveys information observed on a system or network (e.g., an IP address).
- **Report**: Collections of threat intelligence focused on one or more topics, such as a description of a threat actor, malware, or attack technique, including contextual details.
- **Threat Actor**: Individuals, groups, or organizations believed to be operating with malicious intent.
- **Tool**: Legitimate software that can be used by threat actors to perform attacks.
- **Vulnerability**: A mistake in software that can be directly used by a hacker to gain access to a system or network.

STIX also defines two relationship objects (SROs)

- **Relationship**: Used to link two SDOs and to describe how they are related to each other.
- **Sighting**: Denotes the belief that an element of CTI was seen (e.g., indicator, malware).

1.3.3 STIX & ATT&CK

The ATT&CK framework is now available on a public TAXII server, and the content is in STIX2 format. Therefore, the ATT&CK team had to map/translate ATT&CK objects and properties to STIX 2.0 objects and properties syntax.
ATT&CK Concept | STIX Object Type
---|---
Technique | attack-pattern
Group | intrusion-set
Software | malware or tool
Mitigation | course-of-action
Tactic | x-mitre-tractic
Matrix | x-mitre-matrix

You can learn more about the mapping concepts here.

Even though ATT&CK content can be retrieved from its public TAXII server via its own taxii client python library, the cti-taxii-client was developed to consume and process the STIX content in a more efficient way. The `get_object` method from the TAXII client library works well if you already have a specific id of a STIX object. However, the cti-python-stix2 library provides a more dynamic and flexible filtering capability to retrieve ATT&CK content by specific STIX objects such as `attack-patterns`, `intrusion-set`, `x-mitre-matrix`, etc.

ATT&CK users can use the cti-python-stix2 library to retrieve STIX 2.0 data, but they must first reference a STIX Data Source.

- Data Sources represent locations from which STIX data can be retrieved.
- The STIX library comes with a TAXIIICollection suite that contains TAXIICollectionStore, TAXIICollectionSource, and TAXIICollectionSink classes.
- The TAXIIICollection suite supports searching on all STIX2 common object properties. This works simply by augmenting the filtering that is done remotely at the TAXII2 server instance.
- The TAXIIICollection will separate any supplied queries into TAXII supported filters and non-supported filters.
- During a TAXIIICollection API call, TAXII2 supported filters get inserted into the TAXII2 server request (to be evaluated at the server). The rest of the filters are kept locally and then applied to the STIX2 content that is returned from the TAXII2 server, before being returned from the TAXIIICollection API call.
- The TAXIIICollectionSource class retrieves STIX content from local/remote TAXII Collection(s)
- The TAXIIICollectionSource class can be used with a `Collection` object that can be instantiated by the taxii client library.

### 1.3.4 Query ATT&CK

ATT&CK users can use the taxii client and stix libraries together to reference specific STIX objects available in the public ATT&CK TAXII server:

```
>>> from stix2 import TAXIICollectionSource, Filter
>>> from taxii2client import Collection

>>> ATTCK_STIX_COLLECTIONS = "https://cti-taxii.mitre.org/stix/collections/
>>> ENTERPRISE_ATTCK = "95ecc380-afe9-11e4-9b6c-751b66dd541e"
>>> PRE_ATTCK = "062767bd-02d2-4b72-84ba-56caef0f8658"
>>> MOBILE_ATTCK = "2f669986-b40b-4423-b720-4396ca6a462b"
>>> ENTERPRISE_COLLECTION = Collection(ATTCK_STIX_COLLECTIONS + ENTERPRISE_ATTCK + "/")
>>> TC_ENTERPRISE_SOURCE = TAXIICollectionSource(ENTERPRISE_COLLECTION)
>>> PRE_COLLECTION = Collection(ATTCK_STIX_COLLECTIONS + PRE_ATTCK + "/")
>>> TC_PRE_SOURCE = TAXIICollectionSource(PRE_COLLECTION)
```

(continues on next page)
>>> MOBILE_COLLECTION = Collection(ATTCK_STIX_COLLECTIONS + MOBILE_ATTCK + "/")
>>> TC_MOBILE_SOURCE = TAXIICollectionSource(MOBILE_COLLECTION)

We can then use the Filter class available in STIX to retrieve multiple objects:

```python
>>> enterprise_stix_objects = {}
>>> enterprise_filter_objects = {
... "techniques": Filter("type", ",", "attack-pattern"),
... "mitigations": Filter("type", ",", "course-of-action"),
... "groups": Filter("type", ",", "intrusion-set"),
... "malware": Filter("type", ",", "malware"),
... "tools": Filter("type", ",", "tool"),
... "relationships": Filter("type", ",", "relationship")
... }
>>> for key in enterprise_filter_objects:
...     enterprise_stix_objects[key] = self.TC_ENTERPRISE_SOURCE.query(enterprise_filter_objects[key])
```

You can learn more about Filters here.

We can now display the first element in the list of techniques of the enterprise_stix_objects dictionary and validate that we were able to retrieve data from ATT&CK public TAXII server:

```python
>>> enterprise_stix_objects["techniques"][0]
AttackPattern(
    type='attack-pattern',
    id='attack-pattern--cf7b3a06-8b42-4c33-bbe9-012120027925',
    created_by_ref='identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5',
    name='Compile After Delivery',
    description='Adversaries may attempt to make payloads difficult to discover and analyze by delivering files to victims as uncompiled code. Similar to [Obfuscated Files or Information](https://attack.mitre.org/techniques/T1027), text-based source code files may subvert analysis and scrutiny from protections targeting executables/binaries. These payloads will need to be compiled before execution; typically via native utilities such as `csc.exe` or GCC/MinGW. (Citation: ClearSky MuddyWater Nov 2018) Source code payloads may also be encrypted, encoded, and/or embedded within other files, such as those delivered as a [Spearphishing Attachment](https://attack.mitre.org/techniques/T1193).
Payloads may also be delivered in formats unrecognizable and inherently benign to the native OS (ex: EXEs on macOS/Linux) before later being (re)compiled into a proper executable binary with a bundled compiler and execution framework. (Citation: TrendMicro WindowsAppMac)
KillChainPhases=[KillChainPhase(kill_chain_name='mitre-attack', phase_name='defense-evasion')],
external_references=[
    ExternalReference(
        source_name='mitre-attack',
        url='https://attack.mitre.org/techniques/T1500',
        external_id='T1500'
    ),
    ExternalReference(
        source_name='ClearSky MuddyWater Nov 2018',
)]]
```

1.3. CTI Python STIX
You can also retrieve all the stix objects available for each collection without providing a filter:

```python
>>> enterprise_objects = TC_ENTERPRISE_SOURCE.query()
>>> type(enterprise_objects)
<class 'list'>
```

You can then use a similar for loop and an empty list to capture all the STIX object types and count the number of records per object type:

```python
>>> enterprise_list = []
>>> for o in enterprise_objects:
...     enterprise_list.append(o['type'])

>>> from collections import Counter

>>> Counter(enterprise_list)
Counter({
    'relationship': 4852,
    'malware': 278,
    'attack-pattern': 244,
    'course-of-action': 241,
    'intrusion-set': 88,
    'tool': 56,
    'x-mitre-tactic': 12,
})
```
{'x-mitre-matrix': 1,
'identity': 1,
'marking-definition': 1}
}

In addition, you can access object properties for each object type and get more information about what is provided:

```python
>>> object = enterprise_objects[0]
>>> object.object_properties()
[
    'type',
    'id',
    'created_by_ref',
    'created',
    'modified',
    'relationship_type',
    'description',
    'source_ref',
    'target_ref',
    'revoked',
    'labels',
    'external_references',
    'object_marking_refs',
    'granular_markings'
]
```

```python
>>> object.properties_populated()
[
    'object_marking_refs',
    'id',
    'external_references',
    'created',
    'modified',
    'type',
    'created_by_ref',
    'source_ref',
    'relationship_type',
    'target_ref',
    'revoked'
]
```

### 1.3.5 References

- [https://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/attck%E2%84%A2-content-available-in-stix%E2%84%A2-20-via](https://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/attck%E2%84%A2-content-available-in-stix%E2%84%A2-20-via)
- [https://oasis-open.github.io/cti-documentation/stix/intro](https://oasis-open.github.io/cti-documentation/stix/intro)

1.3. CTI Python STIX
1.4 ATTACK CTI Overview

The ATTACK-Python-Client project provides a python library named `attackcti` which wraps the functionality of `cti-python-stix2` and `cti-taxii-client` libraries developed by MITRE. This python wrapper allows ATT&CK users to query STIX content from the ATT&CK public TAXII server via pre-defined functions with a few lines of code.

1.4.1 Requirements

- Python3

1.4.2 Installation

The `attackcti` library can be installed via PIP:

```bash
$ pip install attackcti
```

Or you can install it from source:

```bash
$ git clone https://github.com/Cyb3rWard0g/ATTACK-Python-Client
$ cd ATTACK-Python-Client
$ pip install .
```

1.4.3 Quick Start

You can simply import the `attackcti` library and start retrieving ATT&CK content in STIX from its public TAXII server:

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> all_enterprise = lift.get_enterprise()

>>> len(all_enterprise)
10

>>> all_enterprise.keys()
dict_keys(['techniques', 'mitigations', 'groups', 'malware', 'tools', 'relationships', 'tactics', 'matrix', 'identity', 'marking-definition'])

>>> len(all_enterprise['techniques'])
244

>>> all_enterprise['techniques'][0]
AttackPattern(
```
**Compile After Delivery**

Adversaries may attempt to make payloads difficult to discover and analyze by delivering files to victims as uncompiled code. Similar to [Obfuscated Files or Information](https://attack.mitre.org/techniques/T1027), text-based source code files may subvert analysis and scrutiny from protections targeting executables/binaries. These payloads will need to be compiled before execution; typically via native utilities such as csc.exe or GCC/MinGW. (Citation: ClearSky MuddyWater Nov 2018)

Source code payloads may also be encrypted, encoded, and/or embedded within other files, such as those delivered as a [Spearphishing Attachment](https://attack.mitre.org/techniques/T1193). Payloads may also be delivered in formats unrecognizable and inherently benign to the native OS (ex: EXEs on macOS/Linux) before later being (re)compiled into a proper executable binary with a bundled compiler and execution framework. (Citation: TrendMicro WindowsAppMac)

```
kill_chain_phases=[
    KillChainPhase(
        kill_chain_name='mitre-attack',
        phase_name='defense-evasion',
    ),
];
```

**x_mitre_contributors**=['Ye Yint Min Thu Htut, Offensive Security Team, DBS Bank', 'Praetorian']

**x_mitre_data_sources**=['Process command-line parameters', 'Process monitoring', 'File monitoring']

**x_mitre_defense_bypassed**=['Static File Analysis', 'Binary Analysis', 'Anti-virus', 'Host intrusion prevention systems', 'Signature-based detection']

**x_mitre_detection**='Monitor the execution file paths and command-line arguments for common compilers, such as csc.exe and GCC/MinGW, and correlate with other suspicious behavior to reduce false positives from normal user and administrator behavior. The compilation of payloads may also generate file creation and/or file write events, look for non-native binary formats and cross-platform compiler and execution frameworks like Mono and determine if they have a legitimate purpose on the system. (Citation: TrendMicro WindowsAppMac) Typically these should only be used in specific and limited cases, like for software development.'
By default, the data returned by the available functions in the attackcti library is of type stix2:

```python
>>> type(all_enterprise['techniques'][0])
<class 'stix2.v20.sdo.AttackPattern'>
```

However, you can use the available `stix_format` parameter and set it to `False` to return a dictionary and with a more friendly field name schema as shown below:

```python
>>> all_enterprise_friendly = lift.get_enterprise(stix_format=False)
>>> type(all_enterprise_friendly['techniques'][0])
<class 'dict'>
>>> len(all_enterprise_friendly['techniques'])
244
>>> all_enterprise_friendly['techniques'][0]
{
    'external_references': [
        {
            'external_id': 'T1500',
            'source_name': 'mitre-attack',
            'url': 'https://attack.mitre.org/techniques/T1500',
        },
        {
            'source_name': 'ClearSky MuddyWater Nov 2018',
        },
        {
            'source_name': 'TrendMicro WindowsAppMac',
        }
    ],
    'kill_chain_phases': [
        {
            'phase_name': 'defense-evasion',
            'kill_chain_name': 'mitre-attack'
        }
    ],
    'x_mitre_version': '1.0',
    'url': 'https://attack.mitre.org/techniques/T1500',
    'matrix': 'mitre-attack',
    'technique_id': 'T1500',
}
```
1.4.4 Notebooks

I put together a few Jupyter notebooks for you to learn a little bit more about a few of the functions available in the attackcti library:

- Notebooks

1.5 Library Functions

1.5.1 getenterprise
get_enterprise(self, stix_format=True)

Extracts all the available STIX objects in the Enterprise ATT&CK matrix categorized in the following way:

<table>
<thead>
<tr>
<th>ATT&amp;CK Format</th>
<th>STIX Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>technique</td>
<td>attack-pattern</td>
</tr>
<tr>
<td>mitigation</td>
<td>course-of-action</td>
</tr>
<tr>
<td>group</td>
<td>intrusion-set</td>
</tr>
<tr>
<td>malware</td>
<td>malware</td>
</tr>
<tr>
<td>tool</td>
<td>tool</td>
</tr>
<tr>
<td>relationship</td>
<td>relationship</td>
</tr>
<tr>
<td>tactic</td>
<td>x-mitre-tactic</td>
</tr>
<tr>
<td>matrix</td>
<td>x-mitre-tactic</td>
</tr>
<tr>
<td>identity</td>
<td>identity</td>
</tr>
<tr>
<td>marking-definition</td>
<td>marking-definition</td>
</tr>
</tbody>
</table>

Parameters:

- **stix_format**: returns results in original STIX format or friendly syntax ('attack-pattern' or 'technique')

Returns: Dictionary

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()

>>> enterprise = lift.get_enterprise()

>>> type(enterprise)
<class 'dict'>

>>> enterprise.keys()
dict_keys(['techniques', 'mitigations', 'groups', 'malware', 'tools', 'relationships', 'tactics', 'matrix', 'identity', 'marking-definition'])

>>> type(enterprise['techniques'])
<class 'list'>

>>> type(enterprise['techniques'][0])
<class 'stix2.v20.sdo.AttackPattern'>

>>> print(enterprise['techniques'][0])
{
    "type": "attack-pattern",
    "id": "attack-pattern--cf7b3a06-8b42-4c33-bbe9-012120027925",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-25T20:53:07.719Z",
    "name": "Compile After Delivery",
    "description": "Adversaries may attempt to make payloads difficult to discover by delivering files to victims as uncompiled code. Similar to [Obfuscated Files or Information](https://attack.mitre.org/techniques/T1027), text-based source code files may subvert analysis and scrutiny from protections targeting executables/binaries. These payloads will need to be compiled before execution, typically via native utilities such as csc.exe or GCC/MinGW. (Citation: ClearSky MuddyWater Nov 2018)\n\nSource code payloads may also be encrypted, encoded, and/or embedded within other files, such as those delivered as [Spearphishing Attachment](https://attack.mitre.org/techniques/T1193). Payloads may also be delivered in formats unrecognizable and inherently benign to the native OS (ex: EXEs on macOS/Linux) before later being (re)compiled into a proper binary with a bundled compiler and execution framework. (Citation: TrendMicro WindowsAppMac)"

(continues on next page)
```
"kill_chain_phases": [
    {
        "kill_chain_name": "mitre-attack",
        "phase_name": "defense-evasion"
    }
],
"external_references": [
    {
        "source_name": "mitre-attack",
        "url": "https://attack.mitre.org/techniques/T1500",
        "external_id": "T1500"
    },
    {
        "source_name": "ClearSky MuddyWater Nov 2018",
    },
    {
        "source_name": "TrendMicro WindowsAppMac",
    }
],
"object_marking_refs": [
    "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
],
"x_mitre_contributors": [
    "Ye Yint Min Thu Htut, Offensive Security Team, DBS Bank",
    "Praetorian"
],
"x_mitre_data_sources": [
    "Process command-line parameters",
    "Process monitoring",
    "File monitoring"
],
"x_mitre_defense_bypassed": [
    "Static File Analysis",
    "Binary Analysis",
    "Anti-virus",
    "Host intrusion prevention systems",
    "Signature-based detection"
],
"x_mitre_detection": "Monitor the execution file paths and command-line arguments for common compilers, such as csc.exe and GCC/MinGW, and correlate with other suspicious behavior to reduce false positives from normal user and administrator behavior. The compilation of payloads may also generate file creation and/or file write events. Look for non-native binary formats and cross-platform compiler and execution frameworks like Mono and determine if they have a legitimate purpose on the system. (Citation: TrendMicro WindowsAppMac) Typically these should only be used in specific and limited cases, like for software development."
],
"x_mitre_permissions_required": [
    "User"
]
1.5.2 get_enterprise_techniques

get_enterprise_techniques(self, stix_format=True)

Extracts all the available techniques STIX objects in the Enterprise ATT&CK matrix.

Parameters:

- **stix_format**: returns results in original STIX format or friendly syntax ('attack-pattern' or 'technique')

Returns: List of stix2.v20.sdo.AttackPattern objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()

>>> enterprise_techniques = lift.get_enterprise_techniques()

>>> type(enterprise_techniques)
<class 'list'>

>>> type(enterprise_techniques[0])
<class 'stix2.v20.sdo.AttackPattern'>

>>> print(enterprise_techniques[0])
{
    "type": "attack-pattern",
    "id": "attack-pattern--cf7b3a06-8b42-4c33-bbe9-012120027925",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-25T20:53:07.719Z",
    "name": "Compile After Delivery",
    "description": "Adversaries may attempt to make payloads difficult to discover and analyze by delivering files to victims as uncompiled code. Similar to [Obfuscated Files or Information](https://attack.mitre.org/techniques/T1027), text-based source code files may subvert analysis and scrutiny from protections targeting executables/binaries. These payloads will need to be compiled before execution; typically via native utilities such as csc.exe or GCC/MinGW. (Citation: ClearSky MuddyWater Nov 2018)\n\nSource code payloads may also be encrypted, encoded, and/or embedded within other files, such as those delivered as a [Spearphishing Attachment](https://attack.mitre.org/techniques/T1193). Payloads may also be delivered in formats unrecognizable and inherently benign to the native OS (ex: EXEs on macOS/Linux) before later being (re)compiled into a proper executable binary with a bundled compiler and execution framework. (Citation: TrendMicro)."
}````
"kill_chain_phases": [
  {
    "kill_chain_name": "mitre-attack",
    "phase_name": "defense-evasion"
  }
],
"external_references": [
  {
    "source_name": "mitre-attack",
    "url": "https://attack.mitre.org/techniques/T1500",
    "external_id": "T1500"
  },
  {
    "source_name": "ClearSky MuddyWater Nov 2018",
  },
  {
    "source_name": "TrendMicro WindowsAppMac",
  }
],
"object_marking_refs": [
  "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
],
"x_mitre_contributors": [
  "Ye Yint Min Thu Htut, Offensive Security Team, DBS Bank",
  "Praetorian"
],
"x_mitre_data_sources": [
  "Process command-line parameters",
  "Process monitoring",
  "File monitoring"
],
"x_mitre_defense_bypassed": [
  "Static File Analysis",
  "Binary Analysis",
  "Anti-virus",
  "Host intrusion prevention systems",
  "Signature-based detection"
],
"x_mitre_detection": "Monitor the execution file paths and command-line arguments for common compilers, such as csc.exe and GCC/MinGW, and correlate with other suspicious behavior to reduce false positives from normal user and administrator behavior. The compilation of payloads may also generate file creation and/or file write events. Look for non-native binary formats and cross-platform compiler and execution frameworks like Mono and determine if they have a legitimate purpose on the system. (Citation: TrendMicro WindowsAppMac) Typically these should only be used in specific and limited cases, like for software development."
]
1.5.3 get_enterprise_mitigations

get_enterprise_mitigations(self, stix_format=True)

Extracts all the available mitigations STIX objects in the Enterprise ATT&CK matrix

Parameters:

• stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sdo.CourseOfAction objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> enterprise_mitigations = lift.get_enterprise_mitigations()
>>> type(enterprise_mitigations)
<class 'list'>
>>> type(enterprise_mitigations[0])
<class 'stix2.v20.sdo.CourseOfAction'>
>>>
>>> print(enterprise_mitigations[0])
{
    "type": "course-of-action",
    "id": "course-of-action--70886857-0f19-4caa-b081-548354a8a994",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-26T19:30:33.607Z",
    "modified": "2019-04-26T19:41:45.126Z",
    "name": "Firmware Corruption Mitigation",
    "description": "Prevent adversary access to privileged accounts or access necessary to perform this technique. Check the integrity of the existing BIOS and device firmware to determine if it is vulnerable to modification. Patch the BIOS and other firmware as necessary to prevent successful use of known vulnerabilities.",
    "external_references": [
    {
        "source_name": "mitre-attack",
        "url": "https://attack.mitre.org/techniques/T1495",
```
1.5.4 get_enterprise_groups

get_enterprise_groups(self, stix_format=True)

Extracts all the available groups STIX objects in the Enterprise ATT&CK matrix

Parameters:

• stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sdo.IntrusionSet objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> enterprise_groups = lift.get_enterprise_groups()
>>> type(enterprise_groups)
<class 'list'>
>>> type(enterprise_groups[0])
<class 'stix2.v20.sdo.IntrusionSet'>
>>> print(enterprise_groups[0])
{
    "type": "intrusion-set",
    "id": "intrusion-set--9538b1a4-4120-4e2d-bf59-3b11fcab05a4",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-16T15:14:38.533Z",
    "modified": "2019-04-29T18:59:16.079Z",
    "name": "TEMP.Veles",
    "description": "[TEMP.Veles](https://attack.mitre.org/groups/G0088) is a Russia-based threat group that has targeted critical infrastructure. The group has been observed utilizing TRITON, a malware framework designed to manipulate industrial safety systems. (Citation: FireEye TRITON 2019) (Citation: FireEye TEMP.Veles 2018) (Citation: FireEye TEMP.Veles JSON April 2019)",
    "aliases": [
        "TEMP.Veles",
        "XENOTIME"
    ],
    "external_references": [
        {
            "source_name": "mitre-attack",
            "url": "https://attack.mitre.org/groups/G0088",
        }
    ]
}
```

(continues on next page)
"external_id": "G0088"
},
{
"source_name": "TEMP.Veles",
"description": "(Citation: FireEye TRITON 2019)"
},
{
"source_name": "XENOTIME",
"description": "The activity group XENOTIME, as defined by Dragos, has overlaps with activity reported upon by FireEye about TEMP.Veles as well as the actors behind TRITON. (Citation: Dragos Xenotime 2018) (Citation: Pylos Xenotime 2019) (Citation: FireEye TRITON 2019) (Citation: FireEye TEMP.Veles 2018 )"
},
{
"source_name": "FireEye TRITON 2019",
},
{
"source_name": "FireEye TEMP.Veles 2018",
},
{
"source_name": "FireEye TEMP.Veles JSON April 2019",
"url": "https://www.fireeye.com/content/dam/fireeye-www/blog/files/TRITON_Appendix_C.html"
},
{
"source_name": "Dragos Xenotime 2018",
"url": "https://dragos.com/resource/xenotime/"
},
{
"source_name": "Pylos Xenotime 2019",
"url": "https://pylos.co/2019/04/12/a-xenotime-to-remember-veles-in-the-wild/"
},
{
"source_name": "FireEye TEMP.Veles 2018",
}
1.5.5 get_enterprise_malware

get_enterprise_malware(self, stix_format=True)

Extracts all the available malware STIX objects in the Enterprise ATT&CK matrix

Parameters:

- stix_format: returns results in original STIX format or friendly syntax ('attack-pattern' or 'technique')

Returns: List of stix2.v20.sdo.Malware objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> enterprise_malware = lift.get_enterprise_malware()
>>> type(enterprise_malware)
<class 'list'>
>>> type(enterprise_malware[0])
<class 'stix2.v20.sdo.Malware'>
>>> print(enterprise_malware[0])
{
    "type": "malware",
    "id": "malware--d1531eaa-9e17-473e-a680-3298469662c3",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-23T18:41:36.914Z",
    "modified": "2019-04-29T21:19:34.739Z",
    "name": "CoinTicker",
    "description": "[CoinTicker](https://attack.mitre.org/software/S0369) is a malicious application that poses as a cryptocurrency price ticker and installs components of the open source backdoors EvilOSX and EggShell.(Citation: CoinTicker 2019)",
    "labels": [
        "malware",
    ],
    "external_references": [
        {
            "source_name": "mitre-attack",
            "url": "https://attack.mitre.org/software/S0369",
            "external_id": "S0369"
        },
        {
            "source_name": "CoinTicker 2019",
```
1.5.6 get_enterprise_tools

get_enterprise_tools(self, stix_format=True)

Extracts all the available tools STIX objects in the Enterprise ATT&CK matrix

Parameters:

- stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sdo.Tool objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> enterprise_tools = lift.get_enterprise_tools()
>>> type(enterprise_tools)
<class 'list'>
>>> type(enterprise_tools[0])
<class 'stix2.v20.sdo.Tool'>
>>> print(enterprise_tools[0])

"type": "tool",
"id": "tool--4b57c098-f043-4da2-83ef-7588a6d426bc",
"created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
"created": "2019-04-23T12:31:58.125Z",
"modified": "2019-04-23T18:29:12.005Z",
"name": "PoshC2",
"description": "[PoshC2](https://attack.mitre.org/software/S0378) is an open source remote administration and post-exploitation framework that is publicly available on GitHub. The server-side components of the tool are primarily written in Python, while the implants are written in [PowerShell](https://attack.mitre.org/techniques/T1086). Although [PoshC2](https://attack.mitre.org/software/S0378) is primarily focused on Windows implantation, it does contain a basic Python dropper for Linux/macOS. (Citation: GitHub PoshC2)",
```
"labels": [
    "tool"
],
"external_references": [
    {
        "source_name": "mitre-attack",
        "url": "https://attack.mitre.org/software/S0378",
        "external_id": "S0378"
    },
    {
        "source_name": "GitHub PoshC2",
        "url": "https://github.com/nettitude/PoshC2"
    }
],
"object_marking_refs": [
    "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
],
"x_mitre_aliases": [
    "PoshC2"
],
"x_mitre_platforms": [
    "Windows",
    "Linux",
    "macOS"
],
"x_mitre_version": "1.0"
}

1.5.7 getEnterpriseRelationships

getEnterpriseRelationships(self, stix_format=True)

Extracts all the available relationships STIX objects in the Enterprise ATT&CK matrix

Parameters:

- stix format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sro.Relationship objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> enterprise_relationships = lift.get_enterprise_relationships()
>>> type(enterprise_relationships)
<class 'list'>
>>> type(enterprise_relationships[0])
<class 'stix2.v20.sro.Relationship'>
```
>>> print(enterprise_relationships[0])
{
    "type": "relationship",
    "id": "relationship--2a37ddb3-56ef-4c2d-bec7-d6060eb0215a",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-29T15:54:23.241Z",
    "modified": "2019-04-29T18:16:38.854Z",
    "relationship_type": "uses",
    "source_ref": "intrusion-set--44e43fad-ffcb-4210-abcf-eaaed9735f80",
    "target_ref": "malware--ecc2f65a-b452-4eaf-9689-7e181f17f7a5",
    "external_references": [
        {
            "source_name": "Symantec Chafer Dec 2015",
            "url": "https://www.symantec.com/connect/blogs/iran-based-attackers-use-back-door-threats-spy-middle-eastern-targets"
        },
        {
            "source_name": "Securelist Remexi Jan 2019",
            "url": "https://securelist.com/chafer-used-remexi-malware/89538/"
        }
    ],
    "object_marking_refs": [
        "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
    ]
}

1.5.8 get_enterprise_tactics

get_enterprise_tactics(self, stix_format=True)

Extracts all the available tactics STIX objects in the Enterprise ATT&CK matrix

Parameters:
    • stix_format: returns results in original STIX format or friendly syntax ('attack-pattern' or 'technique')

Returns: List of dictionaries

Examples

>>> from attackcti import attack_client
>>> lift = attack_client()
>>>
>>> enterprise_tactics = lift.get_enterprise_tactics()
>>> type(enterprise_tactics)
<class 'list'>
>>> type(enterprise_tactics[0])
<class 'dict'>
>>> print(enterprise_tactics[0])
{'external_references': [{'external_id': 'TA0040', 'source_name': 'mitre-attack', 'url': 'https://attack.mitre.org/tactics/TA0040'},], 'object_marking_refs': ['marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168'], 'id': 'x-mitre-tactic--5569339b-94c2-49ee-afb3-2222936582c8', 'name': 'Impact', 'created': '2019-03-14T18:44:44.639Z', 'modified': '2019-04-29T14:23:04.506Z', 'type': 'x-mitre-tactic', 'created_by_ref': 'identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5', 'description': 'The Impact tactic represents techniques whose primary objective directly reduces the availability or integrity of a system, service, or network; including manipulation of data to impact a business or operational process. These techniques may represent an adversary’s end goal, or provide cover for a breach of confidentiality.', 'x_mitre_shortname': 'impact'}

1.5.9 get_pre

get_pre(self, stix_format=True)

Extracts all the available STIX objects in the Pre ATT&CK matrix categorized in the following way:

<table>
<thead>
<tr>
<th>ATT&amp;CK Format</th>
<th>STIX Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>technique</td>
<td>attack-pattern</td>
</tr>
<tr>
<td>group</td>
<td>intrusion-set</td>
</tr>
<tr>
<td>relationship</td>
<td>relationship</td>
</tr>
<tr>
<td>tactic</td>
<td>x-mitre-tactic</td>
</tr>
<tr>
<td>matrix</td>
<td>x-mitre-tactic</td>
</tr>
<tr>
<td>identity</td>
<td>identity</td>
</tr>
<tr>
<td>marking-definition</td>
<td>marking-definition</td>
</tr>
</tbody>
</table>

Parameters:

* stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: Dictionary

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> pre = lift.get_pre()
>>> type(pre)
<class 'dict'>
>>> pre.keys()
dict_keys(['techniques', 'groups', 'relationships', 'tactics', 'matrix', 'identity', 'marking-definition'])
>>> type(pre['techniques'])
<class 'list'>
>>> type(pre['techniques'][0])
<class 'stix2.v20.sdo.AttackPattern'>
```
>>> print(pre['techniques'][0])
{
  "type": "attack-pattern",
  "id": "attack-pattern--b182f29c-2505-4b32-a000-0440ef189f59",
  "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
  "created": "2018-04-18T17:59:24.739Z",
  "modified": "2018-10-17T00:14:20.652Z",
  "name": "Spearphishing for Information",
  "description": "Spearphishing for information is a specific variant of spearphishing. Spearphishing for information is different from other forms of spearphishing in that it doesn’t leverage malicious code. All forms of spearphishing are electronically delivered social engineering targeted at a specific individual, company, or industry. Spearphishing for information is an attempt to trick targets into divulging information, frequently credentials, without involving malicious code. Spearphishing for information frequently involves masquerading as a source with a reason to collect information (such as a system administrator or a bank) and providing a user with a website link to visit. The given website often closely resembles a legitimate site in appearance and has a URL containing elements from the real site. From the fake website, information is gathered in web forms and sent to the attacker. Spearphishing for information may also try to obtain information directly through the exchange of emails, instant messengers or other electronic conversation means. (Citation: ATTACKREF GRIZZLY STEPPE JAR),
  "kill_chain_phases": [
    {
      "kill_chain_name": "mitre-pre-attack",
      "phase_name": "technical-information-gathering"
    }
  ],
  "external_references": [
    {
      "source_name": "mitre-pre-attack",
      "url": "https://attack.mitre.org/techniques/T1397",
      "external_id": "T1397"
    },
  {
    "source_name": "ATTACKREF GRIZZLY STEPPE JAR",
  }
  ],
  "object_marking_refs": [
    "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
  ],
  "x_mitre_detectable_by_common_defenses": "Partial",
  "x_mitre檢測可能な一般的な防衛手段": "Partial",
  "x_mitre_defensive_countermeasures": "Depending on the specific method of phishing, the detections can vary. For emails, filtering based on DKIP+SPF or header analysis can help detect when the email sender is spoofed. When it comes to following links, network intrusion detection systems (NIDS), firewalls, removing links, exploding shortened links, proxy monitoring, blocking uncategorized sites, and site reputation based filtering can all provide detection opportunities."
],
  "x_mitre_difficulty_for_adversary": "Yes",
  "x_mitre_difficulty_for_adversary_explanation": "Sending emails is trivial, and over time, an adversary can refine their technique to minimize detection by making their emails seem legitimate in structure and content."}
1.5.10 get_pre_techniques

get_pre_techniques(self, stix_format=True)

Extracts all the available techniques STIX objects in the Pre ATT&CK matrix

Parameters:

• stix_format: returns results in original STIX format or friendly syntax (`attack-pattern` or `technique`)

Returns: List of stix2.v20.sdo.AttackPattern objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()

>>> pre_techniques = lift.get_pre_techniques()
>>> type(pre_techniques)
<class 'list'>

>>> type(pre_techniques[0])
<class 'stix2.v20.sdo.AttackPattern'>

>>> print(pre_techniques[0])
{
    "type": "attack-pattern",
    "id": "attack-pattern--b182f29c-2505-4b32-a000-0440ef189f59",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2018-04-18T17:59:24.739Z",
    "modified": "2018-10-17T00:14:20.652Z",
    "name": "Spearphishing for Information",
    "description": "Spearphishing for information is a specific variant of spearphishing. Spearphishing for information is different from other forms of spearphishing in that it it doesn't leverage malicious code. All forms of spearphishing are electronically delivered social engineering targeted at a specific individual, company, or industry. Spearphishing for information is an attempt to trick targets into divulging information, frequently credentials, without involving malicious code. Spearphishing for information frequently involves masquerading as a source with a reason to collect information (such as a system administrator or a bank) and providing a user with a website link to visit. The given website often closely resembles a legitimate site in appearance and has a URL containing elements from the real site. From the fake website, information is gathered in web forms and sent to the attacker. Spearphishing for information may also try to obtain information directly through the exchange of emails, instant messengers or other electronic conversation means. (Citation: ATTACKREF GRIZZLY STEPPE JAR)",
    "kill_chain_phases": [
    {
        "kill_chain_name": "mitre-pre-attack",
        "phase_name": "technical-information-gathering"
    }
```
1.5.11 get_pre_groups

get_pre_groups(self, stix_format=True)

Extracts all the available groups STIX objects in the Pre ATT&CK matrix

Parameters:

- **stix_format**: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sdo.IntrusionSet objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> pre_groups = lift.get_pre_groups()
>>> type(pre_groups)
<class 'list'>
```
>>> type(pre_groups[0])
<class 'stix2.v20.sdo.IntrusionSet'>

>>> print(pre_groups[0])
{
    "type": "intrusion-set",
    "id": "intrusion-set--9538b1a4-4120-4e2d-bf59-3b11fcab05a4",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-16T14:38.533Z",
    "modified": "2019-04-29T18:59:16.079Z",
    "name": "TEMP.Veles",
    "description": "[TEMP.Veles](https://attack.mitre.org/groups/G0088) is a Russia-based threat group that has targeted critical infrastructure. The group has been observed utilizing TRITON, a malware framework designed to manipulate industrial safety systems.(Citation: FireEye TRITON 2019)(Citation: FireEye TEMP.Veles 2018)(Citation: FireEye TEMP.Veles JSON April 2019)",
    "aliases": [
        "TEMP.Veles",
        "XENOTIME"
    ],
    "external_references": [
        {
            "source_name": "mitre-attack",
            "url": "https://attack.mitre.org/groups/G0088",
            "external_id": "G0088"
        },
        {
            "source_name": "TEMP.Veles",
            "description": "(Citation: FireEye TRITON 2019)"
        },
        {
            "source_name": "XENOTIME",
            "description": "The activity group XENOTIME, as defined by Dragos, has overlaps with activity reported upon by FireEye about TEMP.Veles as well as the actors behind TRITON.(Citation: Dragos Xenotime 2018)(Citation: Pylos Xenotime 2019)(Citation: FireEye TRITON 2019)(Citation: FireEye TEMP.Veles 2018 )"
        },
        {
            "source_name": "FireEye TRITON 2019",
        },
        {
            "source_name": "FireEye TEMP.Veles 2018",
        },
        {
            "source_name": "FireEye TEMP.Veles JSON April 2019",
        }
    ]
}
1.5.12 get_pre_relationships

get_pre_relationships(self, stix_format=True)

Extracts all the available relationships STIX objects in the Pre ATT&CK matrix

Parameters:
- **stix_format**: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sro.Relationship objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()

>>> pre_relationships = lift.get_pre_relationships()
>>> type(pre_relationships)
<class 'list'>

>>> type(pre_relationships[0])
<class 'stix2.v20.sro.Relationship'>
```
>>> print(pre_relationships[0])
{
    "type": "relationship",
    "id": "relationship--21842707-0f15-43bf-bc42-2bceaf2cfa2",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-24T19:44:41.212Z",
    "modified": "2019-04-29T18:59:16.596Z",
    "relationship_type": "uses",
    "description": "[TEMP.Veles](https://attack.mitre.org/groups/G0088) has used dynamic DNS",
    "source_ref": "intrusion-set--9538b14-4120-4e2d-b59-3b1fca05a4",
    "target_ref": "attack-pattern--20a66013-8dab-4ca3-a67d-766c842c561c",
    "external_references": [
      {
        "source_name": "FireEye TRITON 2019",
      }
    ],
    "object_marking_refs": [
      "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
    ]
}

1.5.13 get_pre_tactics

get_pre_tactics(self, stix_format=True)

Extracts all the available tactics STIX objects in the Pre ATT&CK matrix

Parameters:

- **stix_format**: returns results in original STIX format or friendly syntax ('attack-pattern' or 'technique')

Returns: List of dictionaries

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> pre_tactics = lift.get_pre_tactics()
>>> type(pre_tactics)
<class 'list'>
>>> type(pre_tactics[0])
<class 'dict'>
>>> print(pre_tactics[0])
{'external_references': [{'external_id': 'TA0017', 'source_name': 'mitre-attack', 'url': 'https://attack.mitre.org/tactics/TA0017'}, 'object_marking_refs': ['marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168'], 'id': 'x-mitre-tactic--b9f8a273-6167-47cb-89e6-02774d067e24', 'name': 'Organizational Information Gathering', 'created': '2018-10-17T00:14:20.652Z', 'modified': '2018-10-17T00:14:20.652Z', 'type': 'x-mitre-tactic', 'created_by_ref': 'identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5', 'description': 'Organizational information gathering consists of the process of identifying critical organizational elements of intelligence an adversary will need about a target in order to best attack.'}"
```
1.5.14 get_mobile

get_mobile(self, stix_format=True)

Extracts all the available STIX objects in the Mobile ATT&CK matrix categorized in the following way:

<table>
<thead>
<tr>
<th>ATT&amp;CK Format</th>
<th>STIX Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>technique</td>
<td>attack-pattern</td>
</tr>
<tr>
<td>mitigation</td>
<td>course-of-action</td>
</tr>
<tr>
<td>group</td>
<td>intrusion-set</td>
</tr>
<tr>
<td>malware</td>
<td>malware</td>
</tr>
<tr>
<td>tool</td>
<td>tool</td>
</tr>
<tr>
<td>relationship</td>
<td>relationship</td>
</tr>
<tr>
<td>tactic</td>
<td>x-mitre-tactic</td>
</tr>
<tr>
<td>matrix</td>
<td>x-mitre-tactic</td>
</tr>
<tr>
<td>identity</td>
<td>identity</td>
</tr>
<tr>
<td>marking-definition</td>
<td>marking-definition</td>
</tr>
</tbody>
</table>

Parameters:

- **stix_format**: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: Dictionary

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()

>>> mobile = lift.get_mobile()

>>> type(mobile)
<class 'dict'>

>>> mobile.keys()
dict_keys(['techniques', 'mitigations', 'groups', 'malware', 'tools', 'relationships', 'tactics', 'matrix', 'identity', 'marking-definition'])

>>> type(mobile['techniques'])
<class 'list'>

>>> type(mobile['techniques'][0])
<class 'stix2.v20.sdo.AttackPattern'>

>>> print(mobile['techniques'][0])
{
    "type": "attack-pattern",
    "id": "attack-pattern--c6a146ae-9c63-4606-97ff-e261e76e8380",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8855e40b5",
    "created": "2019-02-01T17:29:43.503Z",
    "modified": "2019-02-01T17:29:43.503Z",
    "name": "Web Service",
```

"description": "Adversaries may use an existing, legitimate external Web service as a means for relaying commands to a compromised system. These commands may also include pointers to command and control (C2) infrastructure. Adversaries may post content, known as a dead drop resolver, on Web services with embedded (and often obfuscated/encoded) domains or IP addresses. Once infected, victims will reach out to and be redirected by these resolvers. Popular websites and social media acting as a mechanism for C2 may give a significant amount of cover due to the likelihood that hosts within a network are already communicating with them prior to a compromise. Using common services, such as those offered by Google or Twitter, makes it easier for adversaries to hide in expected noise. Web service providers commonly use SSL/TLS encryption, giving adversaries an added level of protection. Use of Web services may also protect back-end C2 infrastructure from discovery through malware binary analysis while also enabling operational resiliency (since this infrastructure may be dynamically changed).",

"kill_chain_phases": [
{
  "kill_chain_name": "mitre-mobile-attack",
  "phase_name": "command-and-control"
}
],
"external_references": [
{
  "source_name": "mitre-mobile-attack",
  "url": "https://attack.mitre.org/techniques/T1481",
  "external_id": "T1481"
}
],
"object_marking_refs": [
  "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
],
"x_mitrePlatforms": [
  "Android",
  "iOS"
],
"x_mitreTacticType": [
  "Post-Adversary Device Access"
],
"x_mitreVersion": "1.0"
]

1.5.15 get_mobile_techniques

get_mobile_techniques(self, stix_format=True)

Extracts all the available techniques STIX objects in the Mobile ATT&CK matrix

Parameters:

- stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sdo.AttackPattern objects
Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()

>>> mobile_techniques = lift.get_mobile_techniques()

>>> type(mobile_techniques)
<class 'list'>

>>> type(mobile_techniques[0])
<class 'stix2.v20.sdo.AttackPattern'>

>>> print(mobile_techniques[0])
{
    "type": "attack-pattern",
    "id": "attack-pattern--c6a146ae-9c63-4606-97ff-e261e76e8380",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-02-01T17:29:43.503Z",
    "modified": "2019-02-01T17:29:43.503Z",
    "name": "Web Service",
    "description": "Adversaries may use an existing, legitimate external Web service
→ as a means for relaying commands to a compromised system. These commands may
→ also include pointers to command and control (C2) infrastructure. Adversaries may
→ post content, known as a dead drop resolver, on Web services with embedded (and
→ often obfuscated/encoded) domains or IP addresses. Once infected, victims will
→ reach out to and be redirected by these resolvers. Popular websites and social
→ media acting as a mechanism for C2 may give a significant amount of cover due to
→ the likelihood that hosts within a network are already communicating with them
→ prior to a compromise. Using common services, such as those offered by Google or
→ Twitter, makes it easier for adversaries to hide in expected noise. Web service
→ providers commonly use SSL/TLS encryption, giving adversaries an added level of
→ protection. Use of Web services may also protect back-end C2 infrastructure from
→ discovery through malware binary analysis while also enabling operational
→ resiliency (since this infrastructure may be dynamically changed).",
    "kill_chain_phases": [
        {
            "kill_chain_name": "mitre-mobile-attack",
            "phase_name": "command-and-control"
        }
    ],
    "external_references": [
        {
            "source_name": "mitre-mobile-attack",
            "url": "https://attack.mitre.org/techniques/T1481",
            "external_id": "T1481"
        }
    ],
    "object_marking_refs": [
        "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
    ],
    "x_mitre_platforms": [
        "Android",
        "iOS"
    ],
    "x_mitre_tactic_type": [
        "Post-Adversary Device Access"
    ],
}
```
1.5.16 get_mobile_mitigations

get_mobile_mitigations(self, stix_format=True)

Extracts all the available mitigations STIX objects in the Mobile ATT&CK matrix

Parameters:

  • stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sdo.CourseOfAction objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()

>>> mobile_mitigations = lift.get_mobile_mitigations()

>>> type(mobile_mitigations)
<class 'list'>

>>> type(mobile_mitigations[0])
<class 'stix2.v20.sdo.CourseOfAction'>

>>> print(mobile_mitigations[0])
{
    "type": "course-of-action",
    "id": "course-of-action--25dc1ce8-eb55-4333-ae30-a7cb4f5894a1",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2017-10-25T14:48:53.732Z",
    "modified": "2018-10-17T00:14:20.652Z",
    "name": "Application Developer Guidance",
    "description": "This mitigation describes any guidance or training given to developers of applications to avoid introducing security weaknesses that an adversary may be able to take advantage of.",
    "external_references": [  
        {  
            "source_name": "mitre-mobile-attack",
            "url": "https://attack.mitre.org/mitigations/M1013",
            "external_id": "M1013"
        },
    ],
    "object_marking_refs": [  
        "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
    ],
    "x_mitre_old_attack_id": "MOB-M1013",
    "x_mitre_version": "1.0"
}
```

1.5.17 get_mobile_groups
get_mobile_groups(self, stix_format=True)

Extracts all the available groups STIX objects in the Mobile ATT&CK matrix

Parameters:

- stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sdo.IntrusionSet objects

Examples

```python
gp>>> from attackcti import attack_client
gp>>> lift = attack_client()
gp>>> mobile_groups = lift.get_mobile_groups()
gp>>> type(mobile_groups)
gp<class 'list'>
gp>>> type(mobile_groups[0])
gp<class 'stix2.v20.sdo.IntrusionSet'>
gp>>> print(mobile_groups[0])
{
    "type": "intrusion-set",
    "id": "intrusion-set--bef4c620-0787-42a8-a96d-b7eb6e85917c",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "modified": "2019-04-29T18:16:13.040Z",
    "name": "APT28",
    "description": "[APT28](https://attack.mitre.org/groups/G0007) is a threat group that has been attributed to Russia's Main Intelligence Directorate of the Russian General Staff by a July 2018 U.S. Department of Justice indictment. This group reportedly compromised the Hillary Clinton campaign, the Democratic National Committee, and the Democratic Congressional Campaign Committee in 2016 in an attempt to interfere with the U.S. presidential election. [APT28](https://attack.mitre.org/groups/G0007) has been active since at least January 2007. [Citation: DOJ GRU Indictment Jul 2018] (Citation: Ars Technica GRU indictment Jul 2018) (Citation: Crowdstrike DNC June 2016) (Citation: FireEye APT28) (Citation: SecureWorks TG-4127) (Citation: FireEye APT28 January 2017) (Citation: GRIZZLY STEPPE JAR) (Citation: Sofacy DealersChoice) (Citation: Palo Alto Sofacy 06-2018) (Citation: Symantec APT28 Oct 2018)",
    "aliases": [
        "APT28",
        "SNAKEMACKEREL",
        "Swallowtail",
        "Group 74",
        "Sednit",
        "Sofacy",
        "Pawn Storm",
        "Fancy Bear",
        "STRONTIUM",
        "Tsar Team",
        "Threat Group-4127",
        "TG-4127"
    ],
    "external_references": [
```
(continues on next page)
1.5. Library Functions
{
  "source_name": "Tsar Team",
  "description": "(Citation: ESET Sednit Part 3)(Citation: Talos Seduploader Oct 2017)(Citation: Talos Seduploader Oct 2017)"
},
{
  "source_name": "Threat Group-4127",
  "description": "(Citation: SecureWorks TG-4127)"
},
{
  "source_name": "DOJ GRU Indictment Jul 2018",
  "url": "https://www.justice.gov/file/1080281/download"
},
{
  "source_name": "Ars Technica GRU indictment Jul 2018",
},
{
  "source_name": "Crowdstrike DNC June 2016",
  "url": "https://www.crowdstrike.com/blog/bears-midst-intrusion-democratic-national-committee/
"},
{
  "source_name": "FireEye APT28",
},
{
  "source_name": "SecureWorks TG-4127",
  "url": "https://www.secureworks.com/research/threat-group-4127-targets-hillary-clinton-presidential-campaign"
},
{
  "source_name": "FireEye APT28 January 2017",
},
{
  "source_name": "GRIZZLY STEPPE JAR",
}
(continues on next page)
},

"source_name": "Sofacy DealersChoice",
},

"source_name": "Palo Alto Sofacy 06-2018",
url": "https://researchcenter.paloaltonetworks.com/2018/06/unit42-sofacy-groups-parallel-attacks/"
},

"source_name": "Symantec APT28 Oct 2018",
},

"source_name": "Kaspersky Sofacy",
url": "https://securelist.com/sofacy-apt-hits-high-profile-targets-with-updated-toolset/72924/"
},

"source_name": "ESET Sednit Part 3",
},

"source_name": "Talos Seduploader Oct 2017",
},

"source_name": "Securelist Sofacy Feb 2018",
}
1.5.18 get_mobile_malware

get_mobile_malware(self, stix_format=True)

Extracts all the available malware STIX objects in the Mobile ATT&CK matrix

Parameters:

• stix_format: returns results in original STIX format or friendly syntax (`attack-pattern` or `technique`)

Returns: List of stix2.v20.sdo.Malware objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> mobile_malware = lift.get_mobile_malware()
>>> type(mobile_malware)
<class 'list'>
>>> type(mobile_malware[0])
<class 'stix2.v20.sdo.Malware'>
>>> print(mobile_malware[0])
{
    "type": "malware",
    "id": "malware--08784a9d-09e9-4dce-a839-9612398214e8",
    "created_by_ref": "identity--c78cb6e5-0c4b-4b4b-8297-8b5e40b5",
    "created": "2018-10-17T00:14:20.652Z",
    "modified": "2018-10-17T00:14:20.652Z",
    "name": "Allwinner",
    "description": "[Allwinner](https://attack.mitre.org/software/S0319) is a company that supplies processors used in Android tablets and other devices. A Linux kernel distributed by [Allwinner](https://attack.mitre.org/software/S0319) for use on these devices reportedly contained a backdoor. (Citation: HackerNews-Allwinner)",
    "labels": [  
```
"malware",
],
"external_references": [
    {
        "source_name": "mitre-mobile-attack",
        "url": "https://attack.mitre.org/software/S0319",
        "external_id": "S0319"
    },
    {
        "source_name": "Allwinner",
        "description": "(Citation: HackerNews-Allwinner)"
    },
    {
        "source_name": "HackerNews-Allwinner",
        "url": "https://thehackernews.com/2016/05/android-kernal-exploit.html"
    }
],
"object_marking_refs": [
    "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
],
"x_mitre_aliases": [
    "Allwinner"
],
"x_mitre_old_attack_id": "MOB-S0035",
"x_mitre_platforms": [
    "Android"
],
"x_mitre_version": "1.1"
}

1.5.19 get_mobile_tools

get_mobile_tools(self, stix_format=True)

Extracts all the available tools STIX objects in the Mobile ATT&CK matrix

Parameters:

• stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sdo.Tool objects

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()

>>> mobile_tools = lift.get_mobile_tools()
>>> type(mobile_tools)
<class 'list'>

>>> type(mobile_tools[0])
```

(continues on next page)
<class 'stix2.v20.sdo.Tool'>

>>> print(mobile_tools[0])

```
{
    "type": "tool",
    "id": "tool--da21929e-40c0-443d-bdf4-6b60d15448b4",
    "created_by_ref": "identity--c78cb6e5-04b-4611-8297-d1b8b55e40b5",
    "created": "2017-10-25T14:48:48.609Z",
    "modified": "2018-12-11T20:40:31.461Z",
    "name": "Xbot",
    "description": "[Xbot](https://attack.mitre.org/software/S0298) is an Android-malware family that was observed in 2016 primarily targeting Android users in Russia and Australia. (Citation: PaloAlto-Xbot)",
    "labels": [
        "tool"
    ],
    "external_references": [
        {
            "source_name": "mitre-mobile-attack",
            "url": "https://attack.mitre.org/software/S0298",
            "external_id": "S0298"
        },
        {
            "source_name": "Xbot",
            "description": "(Citation: PaloAlto-Xbot)"
        },
        {
            "source_name": "PaloAlto-Xbot",
        }
    ],
    "object_marking_refs": [
        "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
    ],
    "x_mitre_aliases": [
        "Xbot"
    ],
    "x_mitre_old_attack_id": "MOB-S0014",
    "x_mitre_platforms": [
        "Android"
    ],
    "x_mitre_version": "1.1"
}
```

1.5.20 get_mobile_relationships

get_mobile_relationships(self, stix_format=True)

Extracts all the available relationships STIX objects in the Mobile ATT&CK matrix

Parameters:
• **stix_format**: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of stix2.v20.sro.Relationship objects

**Examples**

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> mobile_relationships = lift.get_mobile_relationships()
>>> type(mobile_relationships)
<class 'list'>
>>> type(mobile_relationships[0])
<class 'stix2.v20.sro.Relationship'>
>>> print(mobile_relationships[0])
{
    "type": "relationship",
    "id": "relationship--6186ed87-69a1-43e7-bb60-76527d287e31",
    "created_by_ref": "identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5",
    "created": "2019-04-29T19:31:31.074Z",
    "modified": "2019-04-29T19:31:31.074Z",
    "relationship_type": "revoked-by",
    "source_ref": "attack-pattern--0bcc4ec1-a897-49a9-a9ff-c00df1d1209d",
    "target_ref": "attack-pattern--2d646840-f6f5-4619-a5a8-29c8316bbac5",
    "object_marking_refs": [
        "marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168"
    ]
}
```

1.5.21 **get_mobile_tactics**

**get_mobile_tactics(self, stix_format=True)**

Extracts all the available tactics STIX objects in the Mobile ATT&CK matrix

Parameters:
  • **stix_format**: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List of dictionaries

**Examples**

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> mobile_tactics = lift.get_mobile_tactics()
>>> type(mobile_tactics)
<class 'list'>
>>> type(mobile_tactics[0])
<class 'dict'>
```
>>> print(mobile_tactics[0])
{'external_references': [{'external_id': 'TA0030', 'source_name': 'mitre-attack', 'url': 'https://attack.mitre.org/tactics/TA0030'},], 'object_marking_refs': ['marking-definition--fa42a846-8d90-4e51-bc29-71d5b4802168'], 'id': 'x-mitre-tactic--987cda6d-eb77-406b-bf68-bcb5f3d2e1df', 'name': 'Defense Evasion', 'created': '2018-10-17T00:14:20.652Z', 'modified': '2018-10-17T00:14:20.652Z', 'type': 'x-mitre-tactic', 'created_by_ref': 'identity--c78cb6e5-0c4b-4611-8297-d1b8b55e40b5', 'description': 'Defense evasion consists of techniques an adversary may use to evade detection or avoid other defenses. Sometimes these actions are the same as or variations of techniques in other categories that have the added benefit of subverting a particular defense or mitigation. Defense evasion may be considered a set of attributes the adversary applies to all other phases of the operation.', 'x_mitre_shortname': 'defense-evasion'}

1.5.22 get_data_sources

get_data_sources(self)

Extracts all data sources mapped to techniques across all matrices.

Returns: List

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> lift.get_data_sources()
```

1.5.23 get_techniques_by_datasources
get_techniques_by_datasources(self, *args, stix_format=True)

Extracts all techniques mapped to one or multiple data sources.

Parameters:
- *args: one or more data sources ("datasource1", "datasource2")
- stix_format: returns results in original STIX format or friendly syntax (‘attack-pattern’ or ‘technique’)

Returns: List

Examples

```python
>>> from attackcti import attack_client
>>> lift = attack_client()
>>> techniques = lift.get_techniques_by_datasources("windows event logs")
>>> len(techniques)
22
>>> for t in techniques:
...    print(t['name'],t['x_mitre_data_sources'])
...  Inhibit System Recovery ['Windows Registry', 'Services', 'Windows event logs',
  'Process command-line parameters', 'Process monitoring']
  Group Policy Modification ['Windows event logs']
  File Permissions Modification ['File monitoring', 'Process monitoring', 'Process,
  command-line parameters', 'Windows event logs']
  BITS Jobs ['API monitoring', 'Packet capture', 'Windows event logs']
  CMSTP ['Process monitoring', 'Process command-line parameters', 'Process use of
  network', 'Windows event logs']
  Control Panel Items ['API monitoring', 'Binary file metadata', 'DLL monitoring',
  'Windows Registry', 'Windows event logs', 'Process command-line parameters',
  'Process monitoring']
  Indirect Command Execution ['File monitoring', 'Process monitoring', 'Process command-
  line parameters', 'Windows event logs']
  Kerberoasting ['Windows event logs']
  SIP and Trust Provider Hijacking ['API monitoring', 'Application logs', 'DLL
  monitoring', 'Loaded DLLs', 'Process monitoring', 'Windows Registry', 'Windows,
  event logs']
  Distributed Component Object Model ['API monitoring', 'Authentication logs', 'DLL,
  monitoring', 'Packet capture', 'Process monitoring', 'Windows Registry', 'Windows,
  event logs']
  Dynamic Data Exchange ['API monitoring', 'DLL monitoring', 'Process monitoring',
  'Windows Registry', 'Windows event logs']
  Hooking ['API monitoring', 'Binary file metadata', 'DLL monitoring', 'Loaded DLLs',
  'Process monitoring', 'Windows event logs']
  Image File Execution Options Injection ['Process monitoring', 'Windows Registry',
  'Windows event logs']
  LLINR/NBT-NS Poisoning and Relay ['Windows event logs', 'Windows Registry', 'Packet,
  capture', 'Netflow/Enclave netflow']
  SID-History Injection ['API monitoring', 'Authentication logs', 'Windows event logs']
  Create Account ['Process monitoring', 'Process command-line parameters',
  'Authentication logs', 'Windows event logs']
  Modify Registry ['Windows Registry', 'File monitoring', 'Process monitoring',
  'Process command-line parameters', 'Windows event logs']
  Account Manipulation ['Authentication logs', 'API monitoring', 'Windows event logs',
  'Packet capture']
```

(continues on next page)
1.6 Licenses

GNU GENERAL PUBLIC LICENSE  Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/> Everyone is permitted to copy and
_distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended
to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all
its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies
also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free
software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can
change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies
of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that
they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute
and/or modify it.

For the developers’ and authors’ protection, the GPL clearly explains
that there is no warranty for this free software. For both users’ and authors’ sake, the GPL requires that modified
versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run

Indicator Removal on Host ['File monitoring', 'Process monitoring', 'Process command-line parameters', 'API monitoring', 'Windows event logs']
Scheduled Task ['File monitoring', 'Process monitoring', 'Process command-line parameters', 'Windows event logs']
New Service ['Windows Registry', 'Process monitoring', 'Process command-line parameters', 'Windows event logs']
Obfuscated Files or Information ['Network protocol analysis', 'Process use of network', 'File monitoring', 'Malware reverse engineering', 'Binary file metadata', 'Process command-line parameters', 'Environment variable', 'Process monitoring', 'Windows event logs', 'Network intrusion detection system', 'Email gateway', 'SSL/TLS inspection']
modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users’ freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.


The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official
standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming
language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other
than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is
not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to
implement a Standard Interface for which an implementation is available to the public in source code form. A “Major
Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific
operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object
code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all
the source code needed to generate, install, and (for an executable work) run the object code and to modify the work,
including scripts to control those activities. However, it does not include the work’s System Libraries, or general-
purpose tools or generally available free programs which are used unmodified in performing those activities but which
are not part of the work. For example, Corresponding Source includes interface definition files associated with source
files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is
specifically designed to require, such as by intimate data communication or control flow between those subprograms
and other parts of the work.

The Corresponding Source need not include anything that users
can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that
same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of
copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms
your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this
License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of
fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not
convey, without conditions so long as your license otherwise remains in force. You may convey covered works to
others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for
running those works, provided that you comply with the terms of this License in conveying all material for which you
do not control copyright. Those thus making or running the covered works for you must do so exclusively on your
behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted
material outside their relationship with you.

Conveying under any other circumstances is permitted solely under
the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.


No covered work shall be deemed part of an effective technological
measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20
December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid
circumvention of technological measures to the extent such circumvention is effected by exercising rights under this
License with respect to the covered work, and you disclaim any intention to limit operation or modification of the
work as a means of enforcing, against the work’s users, your or third parties’ legal rights to forbid circumvention of
technological measures.


You may convey verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software
in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided,
in accord with this section must be in a format that is publicly documented (and with an implementation available to
the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this
License by making exceptions from one or more of its conditions. Additional permissions that are applicable to
the entire Program shall be treated as though they were included in this License, to the extent that they are valid
under applicable law. If additional permissions apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by this License without regard to the additional
permissions.

When you convey a copy of a covered work, you may at your option
remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written
to require their own removal in certain cases when you modify the work.) You may place additional permissions on
material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you
add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16
of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that
material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified ver-
sions of such material be marked in reasonable ways as different from the original version;

or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or
service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys
the material (or modified versions of it) with contractual assumptions of liability to the recipi-
ent, for any liability that these contractual assumptions directly impose on those licensors and
authors.

All other non-permissive additional terms are considered “further
restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice
stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a
license document contains a further restriction but permits relicensing or conveying under this License, you may add
to a covered work material governed by the terms of that license document, provided that the further restriction does
not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you
must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating
where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the
form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly
provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.


You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this
definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.


If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue

1.6. Licenses
to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the program’s name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

<program> Copyright (C) <year> <name of author> This program comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’. This is free software, and you are welcome to redistribute it under certain conditions; type ‘show c’ for details.

The hypothetical commands ‘show w’ and ‘show c’ should show the appropriate parts of the General Public License. Of course, your program’s commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>. 

1.6. Licenses