Introduction

What is the Collective Intelligence Framework?

CIF is a cyber threat intelligence management system. CIF allows you to combine known malicious threat information from many sources and use that information for identification (incident response), detection (IDS) and mitigation (null route). The most common types of threat intelligence warehoused in CIF are IP addresses, domains and urls that are observed to be related to malicious activity.

This framework pulls in various data-observations from any source and creates a series of observations "over time" (eg: reputation). When you query for the data, you'll get back a series of observations chronologically and can help you make decisions much as you would look at an email thread.

CIF helps you to parse, normalize, store, post process, query, share and produce data sets of threat intelligence.

The Process

Parse

CIF supports ingesting many different sources of data of the same type; for example data sets or “feeds” of malicious domains. Each similar dataset can be marked with different attributes like source and confidence to name a few.

Normalize

Threat intelligence datasets often have subtle differences between them. CIF normalizes these data sets which gives you a predictable experience when leveraging the threat intelligence in other applications or processes.

Post Process

CIF has many post processors that derive additional intelligence from a single piece of threat intelligence. A simple example would be that a domain and an IP address can be derived from a URL ingested into CIF.

Store

CIF has a database schema that is highly optimized to store millions of records of threat intelligence. CIF v2 uses ElasticSearch as it's datastore.

Query

CIF can be queried via a web browser, native client or directly using the API. CIF has a database schema that is highly optimized to perform queries against a database of millions of records.

Share

CIF supports users, groups and api keys. Each threat intelligence record can be tagged to be shared with specific group of users. This allows the sharing of threat intelligence among federations.

Produce

CIF supports creating new data sets from the stored threat intelligence. These data sets can be created by type and confidence. CIF also supports whitelisting during the feed generation process.

CIF-Community

Mailing list

The primary place to interact with the CIF community is on the CIF Users (https://groups.google.com/forum/?fromgroups#!members/ci-framework) group within Google Groups (https://groups.google.com/forum/#!overview). The CIF Users group has nearly 450 participants.

IRC

You can also find a handful of people hanging out in the #cif channel (http://webchat.freenode.net/?channels=cif) on Freenode.

Community rules

These rules aim to set the expectations of the CIF community; it's not a set of restrictions but a set of expectations enabling the sharing of good ideas.

- Topics should be focused around the development of, integration of CIF and CIF like technologies
- Vendors are more than welcome, ultimately we want integration, this is how we facilitate the growth of sharing threat intelligence
- NO Sales Pitches for non-open source, freely available software unless it's directly relevant to CIF or CIF integration
- Vendors SHALL NOT compare their products to another in this forum (you have a website; you may do it there)
- Vendors that leverage the CIF community as a means for "cold-calling" it's members, at the discretion of the moderators will be banned and publicly cited
- we reserve the right to kick anyone off the list for these reasons or any other reason deemed by the moderators / community at-large

Who-supports-CIF?

CIF is supported by many organizations:

- CSIRT Gadgets Foundation (http://csirtgadgets.org/)
- REN-ISAC (http://www.ren-isac.net)
- Indiana University (http://www.indiana.edu/)
- National Science Foundation (http://www.nsf.gov/)
- Internet2 (http://www.internet2.edu/)

This material is partially-based upon work supported by the National Science Foundation under Grant No. 1127425

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

FAQ

Table of Contents

Also see our FAQ-History (https://github.com/csirtgadgets/massive-octo-spice/issues?labels=faq)

- Troubleshooting CIF
- Asking for help
- Upgrade from v1
- Can I add custom otypes?
- DNS Warnings
- Where's the CIF Client?
- Purge the ElasticSearch database

Troubleshooting CIF

See the wiki page Troubleshooting CIF (https://github.com/csirtgadgets/massive-octo-spice/wiki/Troubleshooting-CIF).

Asking for help

if what you're looking for doesn't appear in the FAQ, here's what info we'll probably need when you log an issue (https://github.com/csirtgadgets/massive-octo-spice/issues/new) or ask the list (https://groups.google.com/forum/?fromgroups#!forum/ci-framework), be sure to post the relevant information:

- [SEARCH THE MAILING LIST] (https://groups.google.com/forum/?fromgroups#!forum/ci-framework), there's a lot of good info in there.

- steps to reproduce the problem

- release version of your OS, and of CIF

- your *obfuscated* config

- recent apache logs as a result of the problem

- a list of running processes that might be useful:

  ```
  $ sudo ps aux | grep cif
  ```

- use something like Github Gists (https://gist.github.com/) to paste the relevant information

- *BE SURE TO OBFUSCATE SENSITIVE DATA*

Upgrade from v1

You cannot upgrade a v1 instance to a v2 instance but you can migrate your data from v1 to v2 (https://github.com/csirtgadgets/massive-octo-spice/wiki/Migration)

Custom otypes

See cif-users thread titled CIF custom data types (https://groups.google.com/forum/#!topic/ci-framework/trpPVxcRqbM)

DNS Warnings

- this can be an EC2-like instance, but be ware of the network activity coming from the box, it could be flagged as malicious, check with your provider's policies
- with post processing, these boxes make a lot of threaded DNS resolution requests, make sure you understand your operating environment and work with your network team to address high volume dns queries

DNS References

- http://www.spamhaus.org/zen
- http://www.spamhaus.org/dbl
- http://www.spamhaus.org/faq/answers.lasso?section=DNSBL%20Usage
- http://www.team-cymru.org/Services/ip-to-asn.html
- http://www.bind9.net/BIND-FAQ

CIF Client

The 'CIF Client' `bin/cif` is now provided by an SDK (SDK) of your choice!

Purge Database

How can I delete all the data in the ElasticSearch database but preserve my API keys?

1. Find the massive-octo-spice git repo on your CIF server
2. `$ cd massive-octo-spice/elasticsearch`
3. `$ make reload-data`

Building-a-CIF-Server

All-in-one

The CIF installation [EasyButton (https://github.com/csirtgadgets/massive-octo-spice/wiki/PlatformUbuntu)](https://github.com/csirtgadgets/massive-octo-spice/wiki/PlatformUbuntu) creates a all-in-one installation of CIF. The means following CIF components are installed on a single host:

- cif-smrt - download, parse, normalize and ingest threat intelligence
- cif-worker - extract additional intelligence from downloaded threat intelligence
- cif-starman - HTTP API
- cif-router - zmq message broker
- ElasticSearch - data warehouse

CPU

A minimum of 8 cores is recommended, technically you can get away with fewer cores but there will be many times the CIF server will be CPU constrained.

Memory

A minimum of 16 GB of memory is recommended, you can expect a idle CIF server to use between 3-6 GB of memory at any given time. We estimate 16 GB of memory will let a single user query ~225K records from ElasticSearch. If you want to support larger queries or multiple users, you will need to allocate more memory.

Disk

The OSINT configurations shipped with CIF use ~400 MB of disk daily. Using nothing but the default data sets you would be using ~146 GB of disk after the first year.

All-in-one sizing recommendations

Small Instance

- an x86-64bit platform
- at-least 16GB ram
- at-least 8 cores
- at-least 250GB of free (after OS install) disk space

Large Instance

- an x86-64bit platform
- at-least 32GB ram
- at-least 16 cores
- at-least 500GB of free (after OS install) disk space
- RAID + LVM knowledge

xLarge Instance

- an x86-64bit platform
- at-least 64GB ram
- at-least 32 cores
- at-least 500GB of free (after OS install) disk space
- RAID + LVM knowledge

Distributed architecture

(To be completed)

Home

Getting Started

Ubuntu LTS is the operating system in which CIF is developed against and is the most commonly used. RHEL and CentOS is the second most common platform used by the community, but lags in community support. If you run into a problem, be sure to first checkout:

- Known Issues (https://github.com/csirtgadgets/massive-octo-spice/issues?labels=bug&state=open)
- FAQ (FAQ)

and as always, contributions welcome! (https://github.com/csirtgadgets/massive-octo-spice/issues/new).

Installation Guides

- (stable) Ubuntu 14 LTS (PlatformUbuntu)
- (stable) AWS Guide

Hardware

Small Instance

- an x86-64bit platform
- at-least 16GB ram
- at-least 8 cores
- at-least 250GB of free (after OS install) disk space

Large Instance

- an x86-64bit platform
- at-least 32GB ram
- at-least 16 cores
- at-least 500GB of free (after OS install) disk space
- RAID + LVM knowledge

xLarge Instance

- an x86-64bit platform
- at-least 64GB ram
- at-least 32 cores
- at-least 500GB of free (after OS install) disk space
- RAID + LVM knowledge

What's new in v2?

- Made the install process significantly easier (two lines (https://github.com/csirtgadgets/massive-octo-spice/wiki/PlatformUbuntu))
- Data is stored as JSON, IODEF and Protocol buffers have been removed
- The datastore is ElasticSearch (http://www.elasticsearch.org/overview/elasticsearch), you can access your data with Kibana (http://www.elasticsearch.org/overview/kibana/)
- Added support for Tags
- see Spamhaus config (https://github.com/csirtgadgets/massive-octo-spice/blob/master/src/rules/default/spamhaus.yml) as an example
- Perl, Python and Ruby SDKs (https://github.com/csirtgadgets/massive-octo-spice/wiki/SDK)

Full change log (https://github.com/csirtgadgets/massive-octo-spice/releases)

Fine Print

bleeding-edge style distro's (eg: release cycles less than 18-24months, Fedora, non-LTS-release ubuntu, etc...) are highly discouraged and are generally not supported

Exploring-the-file-system

This page will help you understand where the important files are for your CIF installation.

Find the CIF binaries on the system

```
$ ls -l /usr/local/bin/ | grep cif
-r-xr-xr-x 1 root root  6672 Nov 29 16:14 cif
```

```
$ ls -l /opt/cif/bin/
-r-xr-xr-x 1 root root 1090 Nov 29 16:17 cif.psgi
-r-xr-xr-x 1 root root 4762 Nov 29 16:17 cif-router
-r-xr-xr-x 1 root root 9478 Nov 29 16:17 cif-smrt
-r-xr-xr-x 1 root root 5396 Nov 29 16:17 cif-tokens
-r-xr-xr-x 1 root root 6770 Nov 29 16:17 cif-worker
```

Find the CIF init.d scripts

```
$ ls /etc/init.d/ | grep cif
cif-router
cif-services
cif-smrt
cif-starman
cif-worker
```

Explore the CIF configuration files on the system

[/etc/cif/]

```
$ ls -l /etc/cif/
-rw-rw---- 1 cif  cif   144 Jul  9 12:35 cif-smrt.yml
-rw-r--r-- 1 root root  190 Jul  8 17:23 cif-starman.conf
-rw-rw---- 1 cif  cif   117 Jul  8 17:23 cif-worker.yml
drwxrwx--- 5 cif  cif  4096 Jul  8 17:23 rules
```

```
$ cat /etc/cif/cif-smrt.yml
---
client:
  remote: http://localhost:5000
  token: <value>
```

```
$ cat /etc/cif/cif-worker.yml
---
client:
  remote: tcp://localhost:4961
  token: <value>

```

[/etc/default/]

```
$ ls -al /etc/default/ | grep cif
-rw-r--r--   1 root root  377 Mar  4 12:22 cif
```

```
$ cat /etc/default/cif
# Directory where the binary distribution resides
CIF_HOME=/opt/cif

PATH=$CIF_HOME/bin:$PATH

if [ -d /opt/cif/lib/perl5 ]; then
    export PERL5LIB=/opt/cif/lib/perl5
fi

# Run as this user ID and group ID
CIF_USER=cif
CIF_GROUP=cif

# data directory
DATA_DIR=/var
LOG_DIR=/var/log

# configuration directory
CONF_DIR=/etc/cif

# add -d to turn on debugging
CIF_DEBUGGING=""
```

[/home/<user>/]

```
$ ls -al /home/<user>/ | grep cif
-rw-rw---- 1 <user> <user>  133 Nov 29 16:19 .cif.yml
```

```
$ cat /home/<user>/.cif.yml
---
client:
  no_verify_ssl: 1
  remote: https://localhost
  token: <value>
```

[/home/cif/]

```
$ ls -l /home/cif/.profile
-rw-r--r-- 1 cif cif 746 Nov 29 16:19 /home/cif/.profile
```

List the preconfigured OSINT rules

```
$ ls -l /etc/cif/rules/default/
-rw-rw---- 1 cif cif  589 Nov 29 16:19 00_whitelist.yml
-rw-rw---- 1 cif cif  266 Nov 29 16:19 1d4_us.yml
-rw-rw---- 1 cif cif  615 Nov 29 16:19 alexa.yml
-rw-rw---- 1 cif cif  721 Nov 29 16:19 alienvault.yml
-rw-rw---- 1 cif cif  479 Nov 29 16:19 aper.yml
-rw-rw---- 1 cif cif  294 Nov 29 16:19 arbor.yml
-rw-rw---- 1 cif cif  441 Nov 29 16:19 bambenekconsulting_com.yml
-rw-rw---- 1 cif cif  309 Nov 29 16:19 botscout.yml
-rw-rw---- 1 cif cif  321 Nov 29 16:19 bruteforceblocker.yml
-rw-rw---- 1 cif cif  903 Nov 29 16:19 cleanmx.cfg
-rw-rw---- 1 cif cif  260 Nov 29 16:19 crimetracker_net.yml
-rw-rw---- 1 cif cif  449 Nov 29 16:19 drg.yml
-rw-rw---- 1 cif cif  482 Nov 29 16:19 feodotracker.yml
-rw-rw---- 1 cif cif  333 Nov 29 16:19 haleys_org.yml
-rw-rw---- 1 cif cif  444 Nov 29 16:19 isc_sans_edu.yml
-rw-rw---- 1 cif cif  602 Nov 29 16:19 malc0de.yml
-rw-rw---- 1 cif cif  261 Nov 29 16:19 malekal.yml
-rw-rw---- 1 cif cif 1309 Nov 29 16:19 malwaredomainlist.cfg
-rw-rw---- 1 cif cif  813 Nov 29 16:19 malwaredomains.yml
-rw-rw---- 1 cif cif  330 Nov 29 16:19 mirc.yml
-rw-rw---- 1 cif cif  279 Nov 29 16:19 nothink_org.yml
-rw-rw---- 1 cif cif  216 Nov 29 16:19 openphish.yml
-rw-rw---- 1 cif cif  469 Nov 29 16:19 phishtank.yml
-rw-rw---- 1 cif cif  805 Nov 29 16:19 shadowserver.cfg
-rw-rw---- 1 cif cif  390 Nov 29 16:19 spamhaus.yml
-rw-rw---- 1 cif cif 1072 Nov 29 16:19 spyeyetracker.yml
-rw-rw---- 1 cif cif  266 Nov 29 16:19 sshbl.yml
-rw-rw---- 1 cif cif  489 Nov 29 16:19 threatexpert.cfg
-rw-rw---- 1 cif cif 1068 Nov 29 16:19 zeustracker.yml
```

```
$ sudo cat /etc/cif/rules/default/drg.yml

parser: pipe
defaults:
  tags: scanner
  protocol: tcp
  provider: dragonresearchgroup.org
  altid_tlp: green
  tlp: amber
  confidence: 85
  values:
    - asn
    - asn_desc
    - observable
    - lasttime
    - null
feeds:
  ssh:
    remote: http://dragonresearchgroup.org/insight/sshpwauth.txt
    application: ssh
    portlist: 22
  vnc:
    remote: http://dragonresearchgroup.org/insight/vncprobe.txt
    application: vnc
    portlist: 5900-5904
```

```
$ ls -l /etc/cif/rules/example/
-rw-rw---- 1 cif cif 453 Nov 29 16:19 freeform.yml
-rw-rw---- 1 cif cif 212 Nov 29 16:19 garwarn.yml
-rw-rw---- 1 cif cif 889 Nov 29 16:19 malware_patrol.yml
-rw-rw---- 1 cif cif 376 Nov 29 16:19 passivedns.yml
-rw-rw---- 1 cif cif 287 Nov 29 16:19 pastebin.yml
```

Explore the Apache config files

```
$ cat /etc/apache2/cif.conf
<Proxy *>
    Order deny,allow
    Allow from all
</Proxy>
ProxyRequests Off
ProxyPreserveHost On
ProxyPass / http://localhost:5000/ keepalive=Off
ProxyPassReverse / http://localhost:5000/
```

```
$ cat /etc/apache2/sites-available/default-ssl.conf
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
        ServerAdmin webmaster@localhost
        Include /etc/apache2/cif.conf

        ErrorLog ${APACHE_LOG_DIR}/error.log
...
```

Explore the Bind config files

```
$ cat /etc/bind/named.conf.options | grep -v '//'
options {
        directory "/var/cache/bind";
        dnssec-validation auto;
        auth-nxdomain no;    # conform to RFC1035
        listen-on-v6 { any; };
    forward only;
    forwarders {
        8.8.8.8;
        8.8.4.4;
    };
};
```

```
$ cat /etc/bind/named.conf.local | grep -v '//'
zone "cymru.com" {
    forward only;
    type forward;
    forwarders { };
};

zone "zen.spamhaus.org" {
    forward only;
    type forward;
    forwarders { };
};

zone "dbl.spamhaus.org" {
    forward only;
    type forward;
    forwarders { };
};
```

Explore the Monit configuration files

```
$ ls -l /etc/monit/conf.d/

-rw-r--r-- 1 root root 846 Mar 28 13:49 cif
-rw-r--r-- 1 root root 355 Mar 28 13:49 elasticsearch
```

Explore the weekly crontab

```
$ ls -l cif* /etc/cron.weekly/

-rwxr-xr-x 1 root root   49 Mar 28 13:49 cif-router
-rwxr-xr-x 1 root root   50 Mar 28 13:49 cif-worker
```

Explore the cache files

```
ls -l /var/smrt/cache/
-rw-r--r-- 1 cif cif      684 Aug 25 14:00 1d4.us-ssh
-rw-r--r-- 1 cif cif  7985835 Aug 25 14:24 20150825.log
-rw-r--r-- 1 cif cif 10068838 Aug 25 13:20 alexa.com-top10
...
```

Exploring-the-CIF-binaries

Manpages for the primary CIF binaries:

- CIF Client (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Manpage)
- CIF-Router (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Router-Manpage)
- CIF-Smrt (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Smrt-Manpage)
- CIF-Tokens (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Tokens-Manpage)
- CIF-Worker (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Worker-Manpage)

Exploring-the-listening-network-services

Listening network services

A clean installation of CIF on Ubuntu 14.04 should create a network profile similar to this:

```
sudo netstat -lnptu

Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address      Foreign Address   State    PID/Program
name
tcp        0      0 192.168.1.12:53    0.0.0.0:*         LISTEN   898/named
(Bind9)
tcp        0      0 127.0.0.1:53       0.0.0.0:*         LISTEN   898/named
(Bind9)
tcp        0      0 0.0.0.0:22         0.0.0.0:*         LISTEN   818/sshd
tcp        0      0 0.0.0.0:25         0.0.0.0:*         LISTEN   1030/master
(Postfix)
tcp        0      0 127.0.0.1:953      0.0.0.0:*         LISTEN   898/named
(Bind9)
tcp        0      0 0.0.0.0:4961       0.0.0.0:*         LISTEN   1548/perl
tcp        0      0 0.0.0.0:4963       0.0.0.0:*         LISTEN   1548/perl
tcp6       0      0 :::9200            :::*              LISTEN   1264/java
tcp6       0      0 :::80              :::*              LISTEN   1108/apache2
tcp6       0      0 :::9300            :::*              LISTEN   1264/java
tcp6       0      0 :::53              :::*              LISTEN   898/named
(Bind9)
tcp6       0      0 :::22              :::*              LISTEN   818/sshd
tcp6       0      0 :::25              :::*              LISTEN   1030/master
(Postfix)
tcp6       0      0 ::1:953            :::*              LISTEN   898/named
(Bind9)
tcp6       0      0 :::443             :::*              LISTEN   1108/apache2
tcp6       0      0 :::5000            :::*              LISTEN   1624/cif.psgi
```

Internet facing Ports:

- 25/tcp -> smtp
- 53/tcp -> dns
- 80/tcp -> http
- 443/tcp -> https
- 5000/tcp -> wsgi
- 9200/tcp -> elastic search

Firewall

If no firewall rules are detected, the firewall is enabled and the two ports open to the world are 22 (ssh) and 443 (https).

Exploring-the-software-packages-installed

Major Software Packages installed

- Apache2
- Bind9
- CIF
- Elasticsearch
- Java
- Monit
- Openjdk-7-jre-headless
- Postfix

Tags

CIF uses tags to describe observables, an single observable can have one tag or many tags. These tags are defined on ingest to CIF. Tags are not predefined by the CIF, a new tag can be created at any time by inserting a new observable with a newly created tag.

Default tags shipped with CIF:

- botnet
- exploit
- feodo
- gozi
- hijacked
- malware
- phishing
- rdata
- scanner
- search
- suspicious
- whitelist
- zeus

You can see an example on how to search by tags with this command:

```
$ cif --tags malware -f csv
amber,everyone,2015-03-20T05:04:16Z,withfx.com,,,60.764,malware,,malc0de.com,
...
```

For definitions for many of the tags shipped by default see this page (https://github.com/csirtgadgets/massive-octo-spice/wiki/Tag-Definitions).

Confidence

Introduction

Confidence details the degree of certainty of a given observation. For instance:

- "I am 85% confident that on `2015-03-20T00:00:01Z` example.com is dropping malware"
- "I am 95% confident that partner-1's observation that `http://example.com/1.html` on `2015-03-20T00:01:01Z` was being used as a phishing url"

One of the primary use cases for confidence is in the generation of threat intellignece feeds. For example, You may want to generate a de-duplicated feed of observables seen within the last seven days with a confidence of 85% or higher to be used in a network sensor.

Details

(95 - 99) Certain

- highly vetted data by known, trusted security professionals
- vetting relationship has been consistent for more than 2 years
- very specific data (eg: ip+port+protocol, or a specific url, or malware hash)
- can typically be used via traffic mitigation processes (null-routing, firewall DROP, etc) with very little risk in collateral damage.

(85 - 94) Very Confident

- vetted data by known, trusted security professionals
- data that has been vetted by a human or set of known and proven processes
- vetting relationship has been consistent and in-place for at-least 1 year
- data feed has been observed for at-least a year
- data should be highly specific (eg: port/protocols, prefixes should be as narrow as possible)
- can typically be used via traffic mitigation processes (null-routing, firewall DROP, etc) with very little risk in collateral damage.

(75 - 84) Somewhat Confident

- semi-vetted data by a security professional or trusted analytics process
- data that has under-gone *some* either machine or human vetting (eg: checked against a whitelist automatically)
- could be leveraged in traffic mitigation processes (eg: dns sink-holing), contains slight risk of collateral damage, but still severely mitigated by native whitelisting process.

(50 - 74) Not Confident

- searches (50)
- machine generated data or enumerated data
- some feeds might fall in the category if the author is lazy, or trying to cram too much into

the feed

- examples might include a domains list where the author is simply taking a botnet urls list and posting just the domains as a feed (65)
- carries risk when used in automatic mitigation processes

(00 - 49) Unknown

- machine generated / enumerated data
- examples include:
- auto-enumerated name-servers from domains
- infrastructure resolved from domain data
- carries significant risk when used in automatic mitigation processes

Timestamp

Timestamps

CIF supports three separate timestamps per record or observation: (reporttime, lasttime, firsttime). A record should have at least one timestamp associated with it and could have up to three timestamps.

Definitions

reporttime

This is the timestamp of when the record or observation was given to you.

lasttime

This is a machine generated timestamp of the last time the source observed the behavior. This would be the most recent timestamp found in machine generated logs where the host is leveraging clock synchronization (NTP).

firsttime

This is a machine generated timestamp of the first time the source observed the behavior. This would be the earliest timestamp found in machine generated logs where the host is leveraging clock synchronization (NTP).

Example

An information sharing partner may give you the following intelligence:

```
address      portlist  protocol  firsttime             lasttime
description
192.168.1.1  22        tcp       2016-06-18T00:00:00Z  2016-06-18T10:10:00Z
scanner
```

If you were to ingest this record into CIF at `2016-06-18T12:00:00Z`, you could associate these three timestamps with this single observation:

```
 firsttime: 2016-06-18T00:00:00Z
  lasttime: 2016-06-18T10:10:00Z
reporttime: 2016-06-18T12:00:00Z
```

CIF-SMRT

When using [cif-smrt (https://github.com/csirtgadgets/massive-octo-spice/wiki/ParsingFeeds#cif-smrt)](https://github.com/csirtgadgets/massive-octo-spice/wiki/ParsingFeeds#cif-smrt) to ingest intelligence into CIF, cif-smrt will automatically fill in lastime, reporttime and firsttime if those values are not specified. It's not uncommon to see the exact same timestamp when dealing with a feed that does not give any timestamps. Here's an example of the alienvault feed:

```
{
    "lasttime" : "2016-05-24T13:01:52Z",
    "firsttime" : "2016-05-24T13:01:52Z",
    "reporttime" : "2016-05-24T13:01:51Z",
    "tlp" : "white",
    "tags" : ["suspicious"],
    "altid" : "https://reputation.alienvault.com/reputation.data",
    "description" : "Scanning Host",
    "altid_tlp" : "white",
    "asn" : "8075",
    "confidence" : 65,
    "group" : ["everyone"],
    "provider" : "reputation.alienvault.com",
    "observable" : "13.84.219.191",
    "otype" : "ipv4",
}
```

Query Examples

A typical CIF query is to return x data over y period. The "y period" can be nuanced as you have two common choices:

1. lasttime
2. reporttime

If the set of records have `lasttime` and `reporttime` specified and the delta between those values is large, the data returned could be rather different when choosing to filter on `lasttime` vs `reporttime`.

The CIFv2 CLI clients default timestamp choice is almost always `reporttime`. When the `lasttime` and `reporttime` values are the same as the alienvault example above, the returned results are very likely what you expect. In the scenario where you know there is a large delta between `lasttime` and `reporttime` and you know you want the period to be based on `lasttime` you'll want to make sure you are being specific in your queries.

Perl CLI examples

Queries using `reporttime`

Under the hood (https://github.com/csirtgadgets/p5-cif-sdk/blob/ed0288f3c33cf12e6c654d472fe76d7721a1329e/bin/cif#L154) these queries use the API parameters (https://github.com/csirtgadgets/massive-octo-spice/wiki/API) `reporttime` and `reporttimeend`

1. `--today` return results for the current day starting at `T00:00:00Z`

```
$ cif --otype fqdn -c 85 --provider osint.bambenekconsulting.com --today
```

1. `--last-hour` return results for the current day and the current hour between `00:00Z` - `59:59Z`

```
$ cif --otype fqdn -c 85 --provider osint.bambenekconsulting.com --last-hour
```

1. `--last-day` return results for the previous 24 hours from the current time.

```
$ cif --otype fqdn -c 85 --provider osint.bambenekconsulting.com --last-day
```

1. `--days [int]` return results for the previous two days from the current time.

```
$ cif --otype fqdn -c 85 --provider osint.bambenekconsulting.com --days 2
```

Queries using `lasttime` and `firsttime`

When you know that you want to query on the machine generated timestamp (`lasttime`) you will need to leverage the filters `lasttime` and `firsttime`.

1. return results for the current day starting at `T00:00:00Z`

```
$ cif --otype fqdn -c 85 --provider osint.bambenekconsulting.com --firsttime
2016-05-24T00:00:00Z --lasttime 2016-05-24T23:59:59Z
```

1. return results for the current day and the current hour between 00:00Z - 59:59Z

```
$ cif --otype fqdn -c 85 --provider osint.bambenekconsulting.com --firsttime
2016-05-24T15:00:00Z --lasttime 2016-05-24T15:59:59Z
```

1. return results for the previous 24 hours from the current time.

```
$ cif --otype fqdn -c 85 --provider osint.bambenekconsulting.com --firsttime
2016-05-23T15:13:59Z --lasttime 2016-05-24T15:14:00Z
```

1. return results for the previous two days from the current time.

```
$ cif --otype fqdn -c 85 --provider osint.bambenekconsulting.com --firsttime
2016-05-22T15:13:59Z --lasttime 2016-05-24T15:14:00Z
```

Whitelist

Whitelisting

CIF has the capability to whitelist observations from entering a feed during the feed generation process.

How does whitelisting work in CIF?

Any observation (IP, domain, URL) with the following will be whitelisted during feed generation:

- tag == whitelist
- Confidence >= 25

How does an observation get an assessment of "whitelist" and a confidece >= 25?

By default CIF is configured with the following whitelists:

- 00_whitelist.yml (https://github.com/csirtgadgets/massive-octo-spice/blob/develop/src/rules/default/00_whitelist.yml)
- alexa.yml (https://github.com/csirtgadgets/massive-octo-spice/blob/develop/src/rules/default/alexa.yml)
- mirc.yml (https://github.com/csirtgadgets/massive-octo-spice/blob/develop/src/rules/default/mirc.yml)

Looking at the 00_whitelist.yml file you'll see there are additional configuration files that contribute to whitelisting. When these feeds are processed, the CIF API applies the following logic:

- resolve all domains to their ip's, slightly degrade the confidence value, whitelist the ip's
- resolve all ip's to their bgp prefix, slightly degrade the confidence value, whitelist the prefix (/16, /18, /22, /24, etc).

For example:

1. google.com is given the assessment 'whitelist' with a confidence value of 95%
2. google.com resolves to: 173.194.46.64-78, which are whitelisted at ~ 69% confidence
3. 173.194.46.64-78 resolves to 173.194.46.0/24 (bgp prefix lookup)
4. 173.194.46.0/24 is whitelisted 47% confidence

When a feed is generated, a whitelist data-set is pre-populated with these values and the feed items are checked against them (sub-domains included).

CIF-tokens

Tokens are used for authorization. Tokens are managed by the `/opt/cif/bin/cif-tokens` tool.

List tokens

Run the cif-tokens tool with no command line switches

```
$ /opt/cif/bin/cif-tokens
username       groups   admin read write acl expires revoked token
cif-smrt       everyone            yes
cbe063846786db05ebe494475f65efde533749ba516206c17c65580218b96a7b
cif-worker     everyone       yes  yes
7d9a03a682f76e6bc486d0aacc230370a4fe362dc9417bd5f48ffbe9c0f09209
root@localhost everyone       yes  yes
ab284e119df6e40f55681d854a76dc4dc1c09b65ea8689d02d993e939c408460
```

Add a user

```
$ /opt/cif/bin/cif-tokens --new --user john.smith@example.com
username               groups   admin read write acl expires revoked token
john.smith@example.com everyone       yes
b76b0ac05393936c34aa3151f3d0a123f822e6c83f73c887fd0f3de96c15797b
```

Delete a user

```
/opt/cif/bin/cif-tokens --delete --username john.smith@example.com
[2015-03-25T11:54:22,932Z][INFO]: 1 tokens deleted...
```

Modify a user

The only things you can modify to an existing user are:

- generate a new token
- remove a token
- revoke a user / token

If you want to change the following properties you have delete the user and create a new user:

- username
- admin flag
- expires date

Usage text

```
/opt/cif/bin/cif-tokens -h

Usage: /opt/cif/bin/cif-tokens [OPTION]

Options:

    -h, --help      this message

    --username      specify a username
    --admin         set the admin flag for the user
    --read          set read permissions for a token
    --write         set write permissions for a token
    --expires       set an expiration date for the token
    --groups        specify the groups for the user (default: everyone)

Actions:

    --new           generate a new token
    --delete        remove token
    --revoke        revoke a user / token
    --import        import tokens list from v1 instance
(bin/cif_apikeys_export) using STDIN
    --import-path   specify a path to read for importing tokens (aka: apikeys
in v1)

    --write-enable  enable write access for a specified user / token
    --write-disable disable write access for a specified user / token

Advanced:

    --generate-config-path      generate a new config with token
    --generate-config-remote    default: https://localhost
    --generate-config-tls       default: true

Storage:

    --storage       default: elasticsearch
    --storage-host  default: localhost:9200

 Examples:
    /opt/cif/bin/cif-tokens --new --user me@example.com --expires 2016-07-01 --
admin
    /opt/cif/bin/cif-tokens --new --user root --groups everyone,groupA,groupB
```

CIF-Groups

CIF supports the creation of groups (buckets) to segment observable's, by default CIF ships with the default user(s) in the *everyone* group and the default OSINT is placed in the *everyone* group.

Default users and groups

Example of default users and their group membership:

```
$ /opt/cif/bin/cif-tokens
username       description groups   admin read write acl expires revoked token
root@localhost             everyone       yes  yes
058f...
cif-smrt                   everyone            yes
c2fa...
cif-worker                 everyone       yes  yes
08b3...
```

Example of OSINT and it's group membership:

```
$ cif --otype ipv4 --provider spamhaus.org --limit 1
tlp  |group   |reporttime          |observable    |cc|asn   |confidence|tags
|description                                                        |rdata
|provider    |altid_tlp|altid
amber|everyone|2015-07-03T18:51:05Z|185.25.150.210|PL|198414|95
|exploit|CBL + customised NJABL. 3rd party exploits (proxies, trojans,
etc.)|185.25.150.210|spamhaus.org|green    |http://www.spamhaus.org/query/bl?
ip=185.25.150.210
```

Adding a user with different group membership

Group membership must be specified when the user is created, you cannot modify a users group membership after the user has been created. Here is an example of creating a user, adding an observable and querying the observable.

1. Add user with membership in group01

```
$ /opt/cif/bin/cif-tokens --new --username john.smith@example.com --read --
write --groups group01
```

1. Add an observable with group01. Note: The user (API token) has read and write permissions to the group group01

```
$ echo
'{"observable":"test.example.com","tlp":"amber","confidence":"25","tags":"malwa
re","provider":"example.com","group":"group01"}' | cif -s --token ba3b...
```

1. Query the observable with a user (API token) with membership in group01

```
$ cif --token ba3b... -q test.example.com
tlp  |group  |reporttime          |observable      |cc|asn|confidence|tags
|description|rdata|provider   |altid_tlp|altid
amber|group01|2015-07-03T19:54:25Z|test.example.com|  |   |25        |malware|
|     |example.com|         |
```

Exploring-the-OSINT-pre-configured

To demonstrate the capabilities of CIF and provide some usefulness out of the box, CIF ships with many Open-source Intelligence (OSINT) feeds preconfigured. You can find explore the default OSINT via the github repo (https://github.com/csirtgadgets/massive-octo-spice/tree/master/src/rules/default) or by listing the configuration files on your CIF server:

```
$ sudo ls -l /etc/cif/rules/default/

-rw-rw---- 1 cif cif  589 Mar 28 13:49 00_whitelist.yml
-rw-rw---- 1 cif cif  268 Mar 28 13:49 1d4_us.yml
-rw-rw---- 1 cif cif  616 Mar 28 13:49 alexa.yml
...
```

CIF was designed to be a data warehouse for all of the threat intelligence availabe to you; it is expected that you will add additional public, private or organic threat intelligence to your CIF server.

Introducing-the-CIF-client

The primary way you will interact with your CIF installation is CIF CLI client [usually] installed at `/usr/local/bin/cif`.

Note: an [SSH server (https://help.ubuntu.com/14.04/serverguide/openssh-server.html)](https://help.ubuntu.com/14.04/serverguide/openssh-server.html) is not installed by default by the CIF installer. You may want install an SSH server to allow you to interact with your CIF server remotely.

This first thing you'll want to do is get familiar with the CIF client by reading the help:

```
$ /usr/local/bin/cif -h
...
```

Examples

Here are many examples on how to use the CIF client:

IP Based Queries

```
$ cif -q 130.201.0.2
$ cif -q 130.201.0.0/16
$ cif -q 2001:4860:4860::8888
```

FQDNs

```
$ cif -q google.com
$ cif -q plus.google.com
```

URLs

```
$ cif -q 'http://www.google.com'
$ cif -q 'https://www.google.com/search?12345.html'
```

Hashes

```
$ cif -q de305d54-75b4-431b-adb2-eb6b9e546013                                 #
uuid
$ cif -q 3b6a927c890f067ad524baac9d751480                                     #
md5
$ cif -q 57c64d62e79a5b9829e5a902e4a3fb22ff618d89                             #
sha1
$ cif -q b712dfc617a327ce948e3341fa4d3f759988c299fcdbc80630f8b3c2c5408be2  #
sha256
```

by Observable Type

Query or filter by observable type

```
$ cif --otype ipv4   # ipv4 address
$ cif --otype ipv6   # ipv6 address
$ cif --otype fqdn   # fully qualified domain address
$ cif --otype url    # url address
$ cif --otype email  # email address

$ cif --otype md5    # md5 hash
$ cif --otype sha1   # sha1 hash
$ cif --otype sha256 # sha256 hash
$ cif --otype sha512 # sha512 hash
$ cif --otype uuid   # uuid hash
```

Tags

Query or filter by tags (https://github.com/csirtgadgets/massive-octo-spice/wiki/Tags)

CIF ships with a handful of tags but you can add your own to any data you ingest in CIF. A few examples:

```
$ cif --tags malware
$ cif --tags botnet
$ cif --tags phishing
$ cif --tags scanner
$ cif --tags zeus
$ cif --tags hijacked
```

Country Code

Query or filter by country code. A few examples:

```
$ cif --cc US
$ cif --cc CN
$ cif --cc JP
```

ASN

Query or filter by ASN. A few examples:

```
$ cif --asn 36351
$ cif --asn 199789
```

Provider

Query of filter by provider, providers are specified at ingest. A few examples:

```
$ cif --provider spamhaus.org
$ cif --provider dshield.org
$ cif --provider dragonresearchgroup.org
```

Confidence

Query of filter by confidence, confidence is specified at ingest. A few examples:

```
$ cif --otype ipv4 -c 95
$ cif --otype fqdn -c 85
$ cif --otype url -c 65
```

Application

Query of filter by application, application is specified at ingest. A few examples:

```
$ cif --otype ipv4 --application ssh
$ cif --otype fqdn --application http
```

Related data

Query of filter by rdata. A few examples:

```
$ cif --rdata ns1.pixelshouse.com
$ cif --rdata ns577.hostgator.com
$ cif --rdata google.com
```

Group

Query of filter by group, groups are specified at ingest. Example:

```
$ cif --otype fqdn --group everyone
$ cif --otype url --group group1,group2,everyone
```

Format

The CIF client can supports several different output formats:

```
$ cif -q google.com -f table
$ cif -q google.com -f json
$ cif -q google.com -f csv
$ cif -q google.com -f snort
$ cif -q google.com -f bro
$ cif -q google.com -f bind
$ cif -q google.com -f html
```

Limit

Limit the number of results returned by CIF. A few examples:

```
$ cif --cc us --limit 5
$ cif --application http -l 5
$ cif --otype fqdn -l 3
```

Time

CIF has many filters that allow you to filter your queries by time.

1. Lasttime - specify filter based on lasttime timestmap (less than)

```
cif --otype url --lasttime 2015-04-07T00:00Z
```

1. Firsttime - specify filter based on firsttime timestmap (greater than)

```
cif --otype url --firsttime 2015-04-07T00:00Z
```

1. Reporttime - specify filter based on reporttime timestmap (greater than)

```
cif --otype url --reporttime 2015-04-07T00:00Z
```

1. Reporttime-end - specify filter based on reporttime timestmap (less than)

```
cif --otype url --reporttime-end 2015-04-07T00:00Z
```

1. Today - auto-sets reporttime to today, 00:00:00Z (UTC)

```
cif --otype url --today
```

1. Last hour - auto-sets reporttime to the beginning of the previous full hour and reporttimeend to end of previous full hour

```
cif --otype url --last-hour
```

ParsingFeeds

Introduction

CIF ships with many [Open-source Intelligence (OSINT) feeds preconfigured](with many Open-source Intelligence (OSINT) feeds preconfigured.). It is expected that additional feeds will be added to the pre-configured OSINT feeds. Additionally, read the tutorial (https://github.com/csirtgadgets/massive-octo-spice/wiki/Parsing-Feeds-Tutorial) on how to create a new feed config file.

Cif-smrt

CIF ships with a utility named cif-smrt. cif-smrt has two primary capabilities; fetching and parsing. cif-smrt has the ability to fetch files using http(s) and from the local file system. cif-smrt has the ability to parse files using the following built-in parsers: regex, json, xml, rss, html, text, cif.

cif-smrt is a service that processes any configuration files found in */etc/cif/rules/default/* with a file extension .yml. cif-smrt is configured to run hourly with a random 30 minute offset.

File Syntax

YAML (http://en.wikipedia.org/wiki/YAML) is the syntax (http://www.yaml.org/refcard.html) used to generate CIF feed configuration files for cif-smrt.

File Format

All parameters can be a Global parameter or a Feed parameter. If the parameter is specified twice, the Feed parameter will supersede the Global parameter.

```
# this is a template cif-smrt configuration file. the purpose of this file
# is to copy it to a newly named file and edit it as needed
#
# cp /etc/cif/rules/example/regex_example.yml
/etc/cif/rules/default/filename.yml

# parser: instruct cif-smrt to use which type of parser
#   values: csv, pipe, regex, json, delim, rss, xml, html, text
parser: regex

# values within default apply to all feeds
defaults:

  # provider: short name of the source, normally the fqdn of the source URL
  provider: feeds.example.com

  # altid_tlp: traffic light protocol (TLP) of the alternet id
  #   (red, amber, green, white)
  altid_tlp: amber

  # tlp: traffic light protocol (TLP) of the observable
  #   (red, amber, green, white)
  tlp: amber

  # confidence: confidence in the observable (65,75,85,95)
  confidence: 75

# values within the friendly name apply only to that feed
feeds:
  # friendly name for feed
  regex_example:

    # remote: URL or filepath on host to feed source
    remote: https://feeds.example.com/scanners.csv

    # pattern: regex pattern to parse and capture the feed data
    pattern: '^(\S+),(\S+)

    # values: captured groups in the regex
    values:
      - observable
      - lasttime

    # tags: tag(s) describing the data (https://goo.gl/OCK8yc)
    tags:
      - scanner
      - suspicious

    # application: application associated with the identified port
    #  (ssh, smtp, http, imap, ftp, sip, vnc, irc)
    application: ssh

    # portlist: Port or a hyphen seperated range of ports
    #  (22, 25, 6667-7000)
    portlist: 22

    # protocol: (tcp, udp)
    protocol: tcp

    # description: text description of the observable
    description: 'hosts seen scanning ssh servers'
```

Common Parameters

Parameter Name	Values	Description	Required
parser	<string>	regex, csv, html, pipe, rss, delim, json, rss, text	no [default: regex]
pattern	<string>	Perl regex with capturing	no
values	<string>	Used with pattern, map;	no
provider	<string>	Friendly name of entity providing the feed	yes
remote	<string>	http(s) URL of feed	yes
confidence	<int>	See Confidence (https://github.com/csirtgadgets/massive-octo-spice/wiki/Confidence)	yes
tags	<string>	See Tags (https://github.com/csirtgadgets/massive-octo-spice/wiki/Tags)	yes
description	<string>	Text description	no
group	<string>	everyone,staff,admin	yes
tlp	<string>	white, green, amber, red	no
altid	<string>	usually a url pointing to the original data point (as a reference id)	no
altid_tlp	<string>	white, green, amber, red	no

Text Files

```
parser: regex
defaults:
  tlp: amber
  provider: 'dshield.org'
  tags: scanner

feeds:
  scanners:
    remote: http://feeds.dshield.org/block.txt
    confidence: 75
    pattern: ^(\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}\b)\t\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}\b\t(\d+)
    values:
      - observable
      - mask
```

Parameter Name	Values	Description
pattern	<string>	a regex string that splits up a line feed
values	- <value>	nested series entry indicator that maps to the regex extracted values

Delimited Text Files

```
parser: delim
defaults:
  confidence: 85
  tlp: amber
  provider: malwaredomains.com

feeds:
  domains:
    remote: http://mirror3.malwaredomains.com/files/domains.zip
    pattern: '[\t|\f]'
    values:
      - null
      - null
      - observable
      - description
      - provider
      - null
    tags:
      - exploit
      - malware
```

Parameter Name	Values	Description
delimiter	<string>	a sudo-regex that splits up the feed
values	- <value>	nested series entry indicator that maps to the parsed columns

XML Files

```
parser: xml
defaults:
  confidence: 50
  tlp: amber
  provider: gist.githubusercontent.com-giovino

feeds:
  domains:
    remote:
https://gist.githubusercontent.com/giovino/3584e069cfe0c04cb5ab/raw/481bf543dfb
d6cc523778312a03b6f5d3f99ba21/gistfile1.xml
    node: root
    map:
      - assessment
      - address
    values:
      - tags
      - observable
```

Parameter Name	Values	Description
map	- <value>	nested series entry indicator of xml elements
values	- <value>	nested series entry indicator of xml element contents

JSON Files

```
parser: json
defaults:
  provider: phishtank.com
  tlp: amber
  application:
    - http
    - https
  confidence: 85
  tags: phishing
  protocol: tcp
  remote: http://data.phishtank.com/data/online-valid.json.gz
  altid_tlp: green

feeds:
  urls:
    otype: url
    map:
      - submission_time
      - url
      - target
      - phish_detail_url
      - details
    values:
      - lasttime
      - observable
      - description
      - altid
      - additional_data
```

Parameter Name	Values	Description
map	- <value>	nested series entry indicator of json keys
values	- <value>	nested series entry indicator of json values

More examples

Additional example feed configuration files can be found here (https://github.com/csirtgadgets/massive-octo-spice/tree/develop/src/rules/example).

Appendix

All Parameters

Parameter Name	Values	Description	Queryable	Required
adata	<string>	Additional data - string, json, csv	no	
altid	<string>	usually a url pointing to the original data point (as a reference id)	no	no
altid_tlp	<string>	white, green, amber, red	no	no
application	<string>	?	yes	no
asn	<string>	Autonomous System Number	yes	no

asn_desc	<string>	Autonomous System Description	no	no
cc	<string>	Two Letter Country Code	yes	no
citycode	<string>	?	no	no
confidence	<int>	See Confidence (https://github.com/csirtgadgets/massive-octo-spice/wiki/Confidence)	yes	?
content	?	?	?	no
description	<string>	Text description	yes	no
disabled	<string>	Values: true, false	no	no
end	<int>	?	no	no
firsttime	?	?	yes	?
group	?	?	yes	?
header	?	?	no	no
ignore	?	?	no	?
lasttime	?	?	yes	?
latitude	double	?	no	?
longitude	double	?	no	?
limit	?	?	no	?
map	?	?	no	?
mask	?	?	no	?
metrocode	?	?	no	?
node	<string>	XML node	no	no
null	?	?	no	?
observable	<string>	IPv4, IPv6, FQDN, URI, Hash, Email address, Binary	yes	Yes
otype	<string>	IPv4, IPv6, FQDN, URI, Hash, Email address, Binary	yes	no
parser	<string>	default (?), csv, html, pipe, rss, delim, json, rss, text	no	?
password	?	?	no	?
pattern	<string>	Perl regex with capturing	no	no
peers	?	?	no	?
portlist	<int>	22 or 80,443 or 6660-7000	yes	no
prefix	?	?	no	?
protocol	<int> <string>	1,6,17 or icmp, tcp, udp	no	no
provider	<string>	Friendly name of entity providing the feed	yes	yes
rank	?	?	no	?
rdata	?	?	yes	?
reference	?	?	no	?
related	?	?	no	?
remote	<string>	http(s) URL of feed	no	yes?
reporttime	?	?	yes	?
rir	?	?	no	?
skip	<string>	Regex patter of line to skip (/^<word>/)	no	no
start	<int>	?	no	no
		0 = no, 1 = yes - used for text parsing do you want		

store_content	<int>	to store the line of text as additional data	no	no
subdivision	?	?	no	?
tags	<string>	See Tags (https://github.com/csirtgadgets/massive-octo-spice/wiki/Tags)	yes	yes
timezone	?	?	no	?
title	?	?	no	?
tlp	<string>	white, green, amber, red	no	no
username	?	?	?	?
values	<string>	Used with pattern, map;	no	no
?	?	?	?	?

cif-smrt usage documentation

```
$ /opt/cif/bin/cif-smrt -h

Usage: /opt/cif/bin/cif-smrt [OPTIONS] [-D status|start|stop|restart|reload]

 Options:
    -C,  --config=FILE       specify cofiguration file, default: /etc/cif/cif-
smrt.yml
    -d,  --debug             turn on debugging (max verbosity)
    -v+, --verbosity         turn up verbosity
    -h,  --help              this message

    -r, --rule=STRING       specify a rule or a rules directory, default:
/etc/cif/rules/default
    -f, --feed=STRING       specify a feed (within a rule)
    -R, --remote=STRING     specify a remote to connect to, default
http://localhost:5000
    -T, --token=STRING      specify a default token/apikey to use
    --not-before=STRING     specify a time to begin processing the data "
[today|yesterday|X days ago]"

    --limit=INT             limit parsing to a subset of records (useful for
debugging)

    --proxy                 specify a proxy address for cif-smrt to use in
fetching feeds
    --https-proxy           specify a proxy for cif-smrt to use for feeds
hosted on https

 Daemon Options:
    -D, --daemon            run as daemon
    -u, --user              run daemon as user, default: cif
    -g, --group             run daemon as group, default: cif
    -p, --pid               pidfile location, default: /var/run/smrt.pid

    --randomstart           random start delay, default: 30 min
    --interval              runtime interval, default: 60 min
    --testmode              run now, overrides randomstart

    --logfile:              logfile location, default: /var/log/cif-smrt.log
    --logging:              turn on logging [to file]

 Notification Options:
    --notify:               turn on notification, default: off.
    --notify-to:            default: root@localhost
    --notify-from:          default: cif
    --notify-subj:          default: [cif-smrt] ERROR
    --notify-level:         default: error

 Advanced Options:
    -M, --meta              apply metadata processors, default: 0
    -c, --clean             clear cache
    -P, --cache             cache location, default /var/smrt/cache

 Examples:
    /opt/cif/bin/cif-smrt -C /etc/cif/cif-smrt.yml
    /opt/cif/bin/cif-smrt -C /etc/cif/cif-smrt.yml -p /var/run/smrt.pid -D
start
    /opt/cif/bin/cif-smrt -r /etc/cif/rules/default -D start
```

Parsing-Feeds-Tutorial

Introduction

This is a walk-through of how to create a feed configuration file to add new threat intelligence feeds to your CIF installation. It explains the commonly used configuration values and how they affect feed generation. If you are already familiar with feed configuration and just need details about all of the configuration parameters, see the parsing feeds (https://github.com/csirtgadgets/massive-octo-spice/wiki/ParsingFeeds) page.

In this example, we'll walk through creating a single feed configuration file to pull two feeds from the Dragon Research Group (http://dragonresearchgroup.org/insight/):

- VNC Probe Report (http://dragonresearchgroup.org/insight/vncprobe.txt)
- SSH Password Authentication Report (http://dragonresearchgroup.org/insight/sshpwauth.txt)

Details

Config Files

File Syntax

YAML (http://en.wikipedia.org/wiki/YAML) is the syntax (http://www.yaml.org/refcard.html) used to generate CIF feed configuration files.

File Format

All parameters can be a Global parameter or a Feed parameter. If the parameter is specified twice, the Feed parameter will supersede the Global parameter.

```
parser: <value>
defaults:
  <parameter>: <value>
  <parameter>: <value>
  <parameter>:
    - <value>
    - <value>
feeds:
  <parameter>: <value>
    <parameter>: <value>
  <parameter>: <value>
    <parameter>: <value>
```

File location

CIF feed configuration files can be found in */etc/cif/rules*:

- default - feeds shipped with a standard CIF installation

- disabled - feeds that have been found to have issues
- example - feed configurations files to be used as example configurations

Note: To browse /etc/cif/rules you'll need to be the 'cif' user (e.g. sudo su - cif).

CIF will load all feed configuration files found in */etc/cif/rules/default* with the file extension *.yml*. Any files without the extension of *.yml* are ignored.

Configuration files can contain multiple feeds which provides a way to group related feeds and make use of global values. When adding a feed source not shipped by default with CIF, it is recommended to create a new config file to avoid the process of merging configs when existing feed configuration files are updated.

Global Variables

Both of Dragon Research Group feeds are sourced from the same provider so they inevitably share many similar configuration values, these are placed at the top of the file.

```
parser: pipe
defaults:
  tags: scanner
  protocol: tcp
  provider: dragonresearchgroup.org
  altid_tlp: green
  tlp: amber
  confidence: 85
  values:
    - asn
    - asn_desc
    - observable
    - lasttime
    - null
```

- **parser: pipe** - these are pipe delimited feeds
- **defaults:** - this is a list of feed configuration values that can be shared across both feeds
- **tags: scanner** - scanner is the Tag (https://github.com/csirtgadgets/massive-octo-spice/wiki/Tags) associated with this type of feed data
- **protocol: tcp** - this is network traffic using the TCP protocol
- **provider: dragonresearchgroup.org** - domain of where the feeds can be found
- **altid_tlp: green** - the URL to the feed data is publicly available
- **tlp: amber** - the feed data is free for non-commercial use only
- **confidence: 85** - 85 percent confident the data is as it is described
- **values:** - this is a list of the pipe delimited columns
- **asn** - asn number provided by the feed provider
- **asn_desc** - asn description provided by the feed provider
- **observable** - the indicator being shared, usually a IP address, FQDN or URL
- **lasttime** - timestamp of last time seen
- **null** - null is used to discard data, it's similar to sending data to /dev/null

Feed Variables

```
feeds:
  ssh:
    remote: http://dragonresearchgroup.org/insight/sshpwauth.txt
    application: ssh
    portlist: 22
  vnc:
    remote: http://dragonresearchgroup.org/insight/vncprobe.txt
    application: vnc
    portlist: 5900-5904
```

- **feeds** - this is a list of feed specific configuration values
- **ssh** - name of feed section, used with -f in cif-smrt
- **vnc** - name of feed section, used with -f in cif-smrt
- **remote** - URL to the providers feed
- **application** - application associated with the listening port
- **portlist** - port(s) associated with application

Final Configuration

Testing

Parsing the feed from the provider

Test the ssh configuration is this config

```
sudo su - cif -c "/opt/cif/bin/cif-smrt --testmode -c -d -r
/etc/cif/rules/default/drg.yml -f ssh"
```

Test both the feed configurations in this config

```
sudo su - cif -c "/opt/cif/bin/cif-smrt --testmode -c -d -r
/etc/cif/rules/default/drg.yml"
```

Alternate-methods-for-fetching-and-parsing-data-sets

CIF ships with a utility named cif-smrt (https://github.com/csirtgadgets/massive-octo-spice/wiki/ParsingFeeds#cif-smrt-usage-documentation) to fetch and parse (https://github.com/csirtgadgets/massive-octo-spice/wiki/ParsingFeeds) threat intelligence feeds. There will be times when cif-smrt does not have the features or capabilities that are needed to fetch or normalize the data appropriately. Here are a few alternate solutions for those scenarios.

Script + cif-smrt

You can create a BASH, Python or Perl script to fetch, normalize (if needed) and write the file to the local file system. You can place this script in the CIF users crontab and write the file locally (e.g. /home/cif/data/data.csv). You would then create a cif-smrt configuration file and place the local file in the remote parameter of the feed configuration file (e.g. remote: /home/cif/data/data.csv).

Script + API

cif-smrt is a tool that fetches, parses and injects data to CIF using the CIF API (https://github.com/csirtgadgets/massive-octo-spice/wiki/API). You can just as easily leverage the API to ingest data into CIF. Here are two example projects that demonstrate how to do this: py-cifapwg (https://github.com/csirtgadgets/py-cifapwg) and py-cifcleanmx (https://github.com/csirtgadgets/py-cifcleanmx)

FAQ:-Parsing-Feeds

Why are some records in a feed not picked up by cif-smrt?

cif-smrt is configured by default to only ingest records with a timestamp that matches the same day of the feed being parsed. Example:

Feed:

```
1.1.1.1, 2015-12-15
2.2.2.2, 2015-12-16
```

Config:

```
parser: csv
defaults:
  provider: example.com
  tlp: green
  altid_tlp: white
  confidence: 85
  alt_tlp: green
  tags:
    - botnet

feeds:
  scanners:
    remote: <url>
    values:
      - observable
      - lasttime
```

If you were to parse that feed on 2015-12-16, the records with a timestamp (lasttime) of 2015-12-15 would be skipped. A reason for this is, some people create feeds that never expire records. Once you parse that feed, you do not need to ingest records that have already been ingested in previous days.

What are some ways around this?

1. You can not parse out the timestamp and cif-smrt will stamp with records with the current day.

```
...

feeds:
  scanners:
    remote: <url>
    values:
      - observable
      - null
```

2. You can instruct (https://github.com/csirtgadgets/massive-octo-spice/blob/develop/src/bin/cif-smrt#L143) cif-smrt to via /etc/cif/cif-smrt.yml to ingest

records with a timestamp X days ago.

Example:

```
$ sudo cat /etc/cif/cif-smrt.yml
---
client:
  remote: http://localhost:5000
  token: <token>
  notbefore: '7 days ago'
```

I have a feed with multiple observables in the same record, how do I correctly parse that record?

You need to parse the feed multiple times parsing out the different observable each time.
Example:

Feed:

```
# IP, FQDN, Timestamp
1.1.1.1, one.example.com, 2015-12-16
```

Config:

```
parser: csv
defaults:
  provider: example.com
  tlp: green
  altid_tlp: white
  confidence: 85
  alt_tlp: green
  tags:
    - botnet

feeds:
  botnet-ip:
    remote: hxxp://example.com/botnet.csv
    values:
      - observable
      - null
      - lasttime

feeds:
  botnet-fqdn:
    remote: hxxp://example.com/botnet.csv
    values:
      - null
      - observable
      - lasttime
```

CIF-Feeds

Introduction

CIF has the ability to generate Threat Intelligence "feeds" from its database of ingested and normalized threats. Minimum characteristics of a CIF feed are:

1. Filtered by observable type (ipv4, fqdn, url, ipv6, email)
2. De-duplicated or aggregated by observable
3. Whitelisting data-sets applied

With those minimum characteristics we would expect that people would apply additional filters, examples of these additional filters would be:

1. confidence (-c)
2. type (--tags botnet)
3. time period (--today, --last-day, --firsttime YYYY-MM-DDT00:00:00Z)
4. format (-f csv, -f bind, -f snort)

Examples

FQDN

- Observable type: fqdn, Confidence: 95, Type (tags): phishing, Period: today, Output format: csv

  ```
  cif --feed --otype fqdn -c 95 --tags phishing --today -f csv
  ```

- Observable type: fqdn, Confidence: 85, Type (tags): botnet, Period: today, Output format: bind

  ```
  cif --feed --otype fqdn -c 85 --tags botnet --today -f bind
  ```

IPv4

- Observable type: ipv4, Confidence: 85, Output format: csv

  ```
  cif --feed --otype ipv4 -c 85 --last-day -f csv
  ```

- Observable type: ipv4, Confidence: 85, Type (tags): exploit, Output format: csv

  ```
  cif --feed --otype ipv4 -c 95 --tags exploit --last-day -f csv
  ```

URL

- Observable type: url, Confidence: 85, Type (tags): phishing, Period: last-day, Output format: json

  ```
  cif --feed --otype url -c 85 --tags phishing --last-day -f json
  ```

- Observable type: url, Confidence: 75, Type (tags): malware, Period: today, Output format: csv

```
cif --feed --otype url -c 75 --tags malware --today -f csv
```

Email

- Observable type: email, Confidence: 75, Type (tags): phishing, Period: last-day, Output format: csv

```
cif --feed --otype email -c 75 --tags phishing --last-day -f csv
```

IPv6

- Observable type: ipv6, Confidence: 75, Type (tags): scanner, Period: today, Output format: csv

```
cif --feed --otype ipv6 -c 75 --tags scanner --today -f csv
```

API

SDK Examples

For more examples, be sure to check out the SDK (https://github.com/csirtgadgets/massive-octo-spice/wiki/SDK) implementations.

Overview

This describes the resources that make up the official CIF API v2. If you have any problems or requests please log an issue (https://github.com/csirtgadgets/massive-octo-spice/issues/new)

- Current Version
- Authorization
- Schema
- Root Endpoint
- Parameters

Current Version

By default, all requests receive the **v2** of the API. We encourage you to explicitly request this version via the `Accept` header.

```
Accept: application/vnd.cif.v2+json
```

Authorization

```
$ curl -H "Accept: application/vnd.cif.v2+json" -H "Authorization: Token
token=8b66f1594f40fc81d907860f2e89b76aeaab6f78941f7a2001f092135421366a"
https://localhost
```

Schema

All data is sent and received as JSON.

Blank fields are can be included as 'null' or omitted.

Basic

```
$ curl -H ... -i https://localhost/observables -H "Authorization: Token
token=8b66f1594f40fc81d907860f2e89b76aeaab6f78941f7a2001f092135421366a"

HTTP/1.0 200 OK
Date: Mon, 01 Dec 2014 13:09:43 GMT
Server: HTTP::Server::PSGI
Content-Length: 2096429
Date: Mon, 01 Dec 2014 13:09:43 GMT
Content-Type: application/json
X-CIF-Media-Type: cif.v2

[]
```

Query

```
$ curl -i -k -H "Accept: application/vnd.cif.v2+json" -H "Authorization: Token
token=0b0bc0da9d596462ab4fbeaf1243318d164cd4371d59e96688570b0f65f45162"
'https://localhost/observables?cc=cn&otype=ipv4&limit=1'

HTTP/1.1 200 OK
Date: Thu, 04 Dec 2014 17:46:54 GMT
Server: Apache/2.4.7 (Ubuntu)
X-CIF-Media-Type: cif.v2
Content-Length: 1734
Content-Type: application/json

[{"prefix":"122.224.0.0\/12","lasttime":"2014-12-
04T09:39:57Z","timezone":"Asia\/Shanghai","asn":"4134","provider":"dragonresear
chgroup.org","otype":"ipv4","citycode":"Hangzhou","asn_desc":"CHINANET-BACKBONE
No.31,Jin-rong Street,CN","tags":["scanner"],"firsttime":"2014-12-
04T09:39:57Z","portlist":"22","cc":"CN","lang":"EN","reporttime":"2014-12-
04T13:16:47Z","latitude":30.2936,"tlp":"amber","observable":"122.225.109.221","
peers":[{"asn_description":"COGENT-174 Cogent
Communications,US","asn":"174","rir":"apnic","date":"2006-11-
16","prefix":"122.224.0.0\/12","cc":"CN"}],"group":
["everyone"],"subdivision":"33","altid_tlp":"green","altid":"http:\/\/dragonres
earchgroup.org\/insight\/sshpwauth.txt","longitude":120.1614,"id":"216cba10185b
97dfb148f98c3dcc1f40023ec5055592561f896df87dbdef72ee","rir":"apnic","confidence"
:85,"application":"ssh","protocol":6}]
```

an expanded version of this can be found here (APIQueryExpanded)

Root Endpoint

The root endpoint for the API is `/observables`.

Parameters

Many API methods take optional parameters. For GET requests, any parameters not specified as a segment in the path can be passed as an HTTP query string parameter:

```
$ curl -H ... -i "https://localhost/observables?cc=us"
```

In this example, the ‘observables’ is provided for the :observables parameters in the path while :cc is passed in the query string.

For PUT requests, parameters not included in the URL should be encoded as JSON with a Content-Type of ‘application/x-www-form-urlencoded’.

Current supported parameters include:

Name	Type	Description
q	string	The observable to query for
otype	string	(ipv4, ipv6, fqdn, url, email)
nolog	int	Do NOT log the query
observable	string	The observable to query for
portlist	string	list of ports (ex: 1,2,445-557)
protocol	string	layer 4 protocol (icmp, tcp, udp)
cc	string	The country code to filter on
asn	int	The ASN to filter on
confidence	int	The confidence (or greater) to filter on
group	string	The group(s) to filter on (CSV accepted as OR)
tags	string	The tag(s) to filter on (CSV accepted as OR)
provider	string	The provider(s) to filter on (CSV accepted as AND)
application	string	The application(s) to filter on (CSV accepted as AND)
description	string	Text description of the observable
rdata	string	Related data: used mainly by cif-worker when re-injecting intelligence
reporttime	string	Reported timestamp, (YYYY-MM-DDTHH:MM:SSZ) - Greater than or equal to
reporttimeend	string	A filter to limit results, (YYYY-MM-DDTHH:MM:SSZ) - Less than or equal to
firsttime	string	First seen machine generated timestamp, (YYYY-MM-DDTHH:MM:SSZ) - Greater than or equal to
lasttime	string	Last seen machine generated timestamp, (YYYY-MM-DDTHH:MM:SSZ) - Less than or equal to
limit	int	limit the results returned
adata	string	Additional data: could be a text string or json blob

Examples include:

```
$ curl -H ... -i "https://localhost/observables?cc=us"
$ curl -H ... -i "https://localhost/observables?q=example.com"
$ curl -H ... -i "https://localhost/observables?
observable=1.2.3.4&provider=dragonresearchgroup.com"
$ curl -H ... -i "https://localhost/observables?
tags=botnet,zeus&confidence=65&cc=us"
```

CIF-SDK

CIF Software Development Kit's (SDK)

CIF has multiple SDKs which make it much easier to interact with CIF using the API or the CLI.

- CIF Software Development Kit for Perl (https://github.com/csirtgadgets/p5-cif-sdk)
- CIF Software Development Kit for Python (https://github.com/csirtgadgets/py-cifsdk)

Bulk-Submissions

If you are using the API to submit over 50 records you will want to ensure you are not making 50 individual HTTP POSTs to the CIF server.

Use a list / array

Make sure you are submitting a single list/array of 50 records. Note that the server will not return the submission IDs until all the post processing is done, if you are submitting 10000 records it could take a few minutes to get a return from the server.

Python SDK Example:

```
from cifsdk.client import Client
import json

cli = Client(token='<token>',
             remote='https://localhost',
             no_verify_ssl=1)

# sample dict
d =
{"observable":"1.1.1.1","tlp":"amber","confidence":"85","tags":"malware","provi
der":"me.com","group":"everyone"}

# build list of dicts
build = 0
data = []
while build < 10000:
    data.append(d)
    build += 1

# json'ify the list
json_data = json.dumps(data)

ret = cli.submit(json_data)
print(ret)
```

Use the nowait argument

Use the client argument nowait to tell the server to batch submissions on the server side.

Python SDK Example:

```
cli = Client(token='<token>',
             remote='https://localhost',
             no_verify_ssl=1
             nowait=True)
```

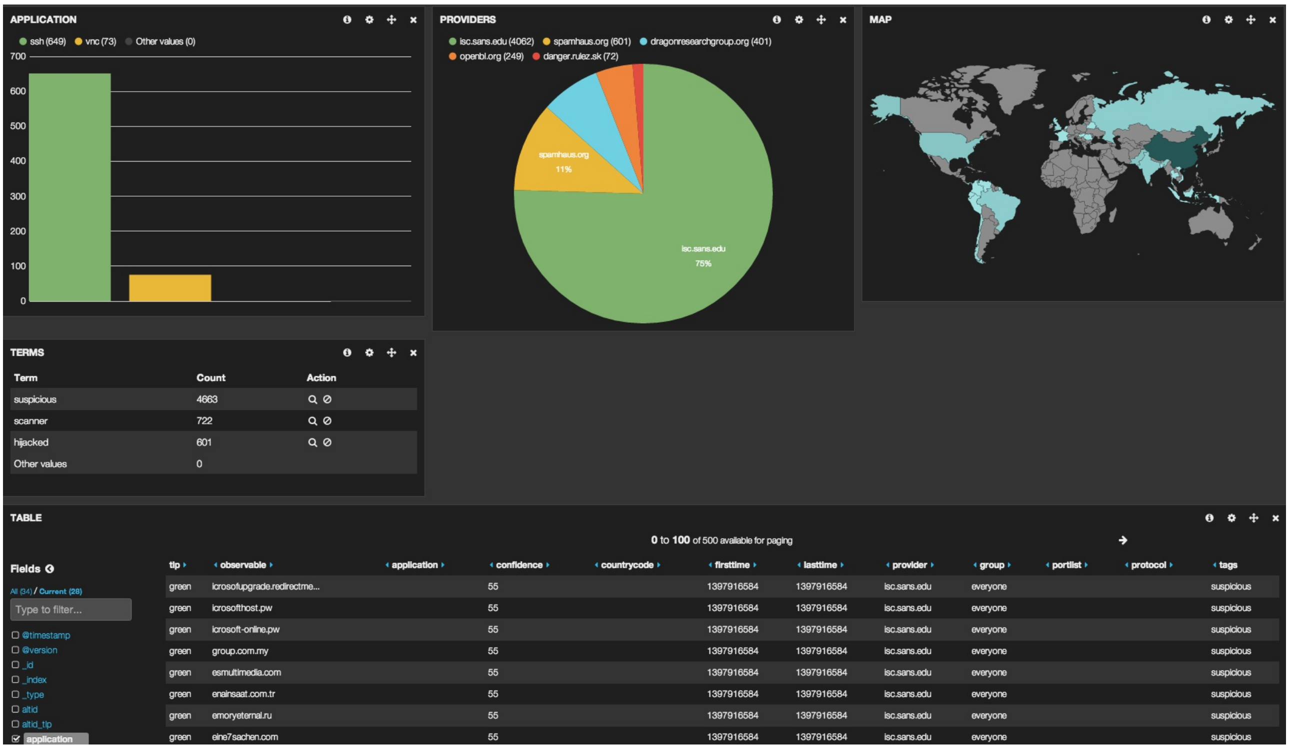
Reference:

Bottleneck sending events through HTTP API (https://groups.google.com/forum/#!topic/ci-framework/8RPtr4jvZkI)

KibanaGuide

Kibana

Since CIF is built upon [ElasticSearch (http://elasticsearch.org)](http://elasticsearch.org), Kibana can be installed to talk directly to the ElasticSearch instance and create some eye-candy:


APPLICATION
● ssh (649) ● vnc (73) ● Other values (0)
700
600
500
400
300
200
100
0
PROVIDERS
● isc.sans.edu (4062) ● spamhaus.org (601) ● dragonresearchgroup.org (401)
● openbl.org (249) ● danger.rulez.sk (72)
spamhaus.org
11%
isc.sans.edu
75%
MAP

TERMS

Term	Count	Action
suspicious	4663	🔍 ⊘
scanner	722	🔍 ⊘
hijacked	601	🔍 ⊘
Other values	0	

TABLE

0 to **100** of 500 available for paging

Fields

All (34) / Current (28)

Type to filter...

□ @timestamp
□ @version
□ _id
□ _index
□ _type
□ altid
□ altid_tlp
☑ application

tlp	observable	application	confidence	countrycode	firsttime	lasttime	provider	group	portlist	protocol	tags
green	icrosofupgrade.redirectme...		55		1397916584	1397916584	isc.sans.edu	everyone			suspicious
green	icrosofthost.pw		55		1397916584	1397916584	isc.sans.edu	everyone			suspicious
green	icrosoft-online.pw		55		1397916584	1397916584	isc.sans.edu	everyone			suspicious
green	group.com.my		55		1397916584	1397916584	isc.sans.edu	everyone			suspicious
green	esmultimedia.com		55		1397916584	1397916584	isc.sans.edu	everyone			suspicious
green	enainsaat.com.tr		55		1397916584	1397916584	isc.sans.edu	everyone			suspicious
green	emoryeternal.ru		55		1397916584	1397916584	isc.sans.edu	everyone			suspicious
green	eine7sachen.com		55		1397916584	1397916584	isc.sans.edu	everyone			suspicious

Keep in mind, if you're going to run Kibana on the same host as CIF, you might need to alter the default proxy paths in `/etc/apache2/cif.conf` as by default, the CIF API runs under `/`. It might be best to run kibana under a different host or the API as a separate virtual host

```
ProxyPass /api http://localhost:5000/ keepalive=Off
ProxyPassReverse /api http://localhost:5000/
```

Installation

1. install [Kibana3 (https://github.com/elasticsearch/kibana/tree/3.0#kibana)](https://github.com/elasticsearch/kibana/tree/3.0#kibana) into `/var/www/kibana`
2. integrate with [a webserver (https://github.com/elasticsearch/kibana/blob/3.0/sample)](https://github.com/elasticsearch/kibana/blob/3.0/sample)

```
<Location /kibana>
  Allow from all
  Options -Multiviews
</Location>
```

1. read [the guide (http://www.elasticsearch.org/guide/en/kibana/current/)](http://www.elasticsearch.org/guide/en/kibana/current/)
2. copy the demo CIF dashboard

```
$ sudo cp contrib/kibana-dashboard.json /var/www/kibana/app/dashboards/cif.json
$ sudo chown -R www-data:www-data /var/www/kibana/app/dashboards/cif.json
```

1. log into the demo dash: `https://localhost/kibana/index.html#/dashboard/file/cif.json`

Notes

- Make sure the permissions are correct for `/var/www/kibana`
- If you're using apache, make sure mod-proxy is enabled (ubuntu: `libapache2-mod-proxy-html`)
- with apache, this can be included in `sites-enabled/default-ssl` as an `Include` directive to `/etc/apache2/kibana.conf` where the sample apache config (https://github.com/elasticsearch/kibana/blob/3.0/sample/apache_ldap.conf) can be placed.

```
<VirtualHost *:80>
   ServerAdmin webmaster@localhost
   Include /etc/apache2/cif.conf
   Include /etc/apache2/kibana.conf

   DocumentRoot /var/www
   ...
```

References

- http://www.ragingcomputer.com/2014/02/securing-elasticsearch-kibana-with-nginx

Bind

Configure Bind

1. Edit named.conf

```
$ sudo vim /etc/bind/named.conf
```

1. Add the following

```
$ include "/var/lib/bind/sink_local.conf";
```

1. Create a sink_local.conf file

```
$ sudo touch /var/lib/bind/sink_local.conf
```

1. Change permissions on sink_local.conf file to root:bind

```
$ sudo chown root:bind /var/lib/bind/sink_local.conf
```

1. Run the command "named-checkconf" to make sure you have no errors in your named.conf file.

```
$ sudo /usr/sbin/named-checkconf
```

1. Create a zone file

```
$ sudo vim /etc/bind/cif_domain_malware.zone
```

1. Copy the following

```
$TTL 600

@       IN      SOA     localhost     root (
                      1               ; serial number
                      3H              ; Refresh
                      15M             ; Retry
                      1W              ; Expire
                      1D )            ; Min TTL

      24H IN NS              @
      24H IN A               127.0.0.1
*       24H IN A               127.0.0.1
```

1. For '''testing / demonstration''' purposes only, allow any user to write to the

```
$ sudo chmod 666 /var/lib/bind/sink_local.conf
```

1. Configure the client to export a sinkhole file

```
$ cif --otype fqdn --tags malware,botnet -c 85 --feed --format bind >
/var/lib/bind/sink_local.conf
```

1. Reload configuration file and new zones only

```
$ sudo /usr/sbin/rndc reconfig
```

1. Run the command "named-checkconf" to make sure you have no errors

```
$ sudo /usr/sbin/named-checkconf
```

Test

1. Find a domain in sink_local.conf

```
$ cat /var/lib/bind/sink_local.conf
```

1. Test the domain against the local server using dig

```
$ dig @localhost hjmnuuyej1152klu.com

; <<>> DiG 9.7.0-P1 <<>> @localhost hjmnuuyej1152klu.com
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17755
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;hjmnuuyej1152klu.com.          IN      A

;; ANSWER SECTION:
hjmnuuyej1152klu.com.   86400   IN      A       127.0.0.1

;; AUTHORITY SECTION:
hjmnuuyej1152klu.com.   86400   IN      NS      hjmnuuyej1152klu.com.

;; Query time: 42 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Jan 19 10:55:03 2012
;; MSG SIZE  rcvd: 68
```

External References

- https://github.com/mrmuth/SafeDNS

TippingPoint

This doc needs some work

```
#!/bin/sh

#
# cif_domains_to_sms.sh
#
# Loads CIF v0.01 Malicious Domain Feed Into Tippingpoint SMS Reputation Database
# v0.04 - 20120314
# Jeff Kell - Original Idea, Debug, Input
# Anthony Maszeroski - Polish and Pancakes
#

#
# SMS Instructions:
#
# a.) Log in to your SMS
# b.) On the "Profiles" tab, Select "Reputation Database" in the left-hand nav bar
# c.) Select the "Tag Categories" tab in the right-hand pane
# d.) Add the following User Defined Tag Categories
#       i.) confidence (Numeric Range)
#      ii.) description (Text)
#     iii.) impact (Text)
#      iv.) restriction (Text)
#       v.) severity (Text)
# e.) Select the SMS profile that is applied to your outbound Internet traffic
# f.) Select "Reputation" under "Infrastructure Protection"
# g.) Add an appropriate policy, e.g.:
#       i.) Filter Info : Name=CIF; Action : State=Enabled, Action Set=Block+Notify
#      ii.) Entry Criteria : DNS Domains; Tag Criteria : Include Tagged Entries, confidence="greater than or equal to 65"
# h.) Distribute the profile
#
# You'll know that it's working when you see a slew of blocked DNS query traffic involving domains in the feed
#

#
# Temp File / Directory Info
#

OUTFILE='bt-cif-domains.csv'
TMPFILE='bt-cif-domains.txt'
WORKDIR='/tmp/cif'
```

```
#
# Location Of System Binaries
# (These Are FreeBSD Defaults)
#

AWK='/usr/bin/awk'
CAT='/bin/cat'
CIF='/usr/local/bin/cif'
CURL='/usr/local/bin/curl'
MKDIR='/bin/mkdir'
GREP='/usr/bin/grep'
GZIP='/usr/bin/gzip'
RM='/bin/rm'
SED='/usr/bin/sed'
SLEEP='/bin/sleep'
SORT='/usr/bin/sort'
WGET='/usr/local/bin/wget'

#
# Tippingpoint SMS Configuration
#

SMSSERVER='HOST.DOMAIN'

##
## SMS Throws Errors If Successive API Calls Are Made Too Quickly
##

SMSSLEEPSECS='10'

SMSID=''
SMSPW=''

if [ "${SMSID}" = "" ]; then
    SMSID=`cat ~root/.smsid`
fi

if [ "${SMSPW}" = "" ]; then
    SMSPW=`cat ~root/.smspw`
fi

#
# Create Scratch Space
#

if [ ! -d "${WORKDIR}" ]; then
   ${MKDIR} -m 0700 "${WORKDIR}" > /dev/null 2>&1
fi

#
# Fetch CIF Domain Feeds
```

```
#

${CIF} --config ~root/.cif --tags malware --otype fqdn --format csv --
confidence 65 | ${GREP} -v ^# | ${GREP} -v ^$ | awk -F, '{$4=sprintf("%d",$4 +
0.5)} {print $1",confidence,"$4",description,"$5","$8",restriction,"$14"}' >
"${WORKDIR}/${TMPFILE}"
${CIF} --config ~root/.cif --tags botnet --otype fqdn --confidence 65 --format
csv | ${GREP} -v ^# | ${GREP} -v ^$ | awk -F, '{$4=sprintf("%d",$4 + 0.5)}
{print $1",confidence,"$4",description,"$5","$8",restriction,"$14"}' >>
"${WORKDIR}/${TMPFILE}"

#
# (Optional) - Delete All Existing User Reputation Entries
#

if [ -s "${WORKDIR}/${TMPFILE}" ]; then
    ${WGET} -q --no-check-certificate "https://${SMSSERVER}/repEntries/delete?
smsuser=${SMSID}&smspass=${SMSPW}&criteria=user" -O - > /dev/null 2>&1
fi

# Sort The Feed, Deduplicate

${CAT} "${WORKDIR}/${TMPFILE}" | ${SORT} -t, -u -k1,1 | ${SORT} >
"${WORKDIR}/${OUTFILE}"

#
# Load Combined Domain Lists Into The SMS
#

if [ -s "${WORKDIR}/${OUTFILE}" ]; then
    ${SLEEP} ${SMSSLEEPSECS}
    ${CURL} -s -f -k -F "file=@${WORKDIR}/${OUTFILE}"
"https://${SMSSERVER}/repEntries/import?
smsuser=${SMSID}&smspass=${SMSPW}&type=dns"
fi

#
# Cleanup Bits Of Pancakes And Syrup
#

if [ -d "${WORKDIR}" ]; then
   ${RM} -rf "${WORKDIR}" > /dev/null 2>&1
fi
```

PassiveDNS

Simple passive dns integration

Ubuntu

1. install the gamelinux passive dns sensor

```
$ sudo apt-get git build-essential libldns-dev libpcap-dev
$ git clone https://github.com/gamelinux/passivedns
$ cd passivedns/src && make
$ sudo make install
$ sudo passivedns -i eth0
```

1. test with the following CIF config

```
confidence = 95
tlp = green
tags = 'passive'

# https://github.com/gamelinux/passivedns
[gamelinux]
provider = localhost
remote = /var/log/passive.log
parser = delim
pattern = '^(\d+\.\d+)\|\|[\w\.]+\|\|[\w\.]+\|\|[\w\.]+\|\|([\w\.]+)\.\|\|[A-Z]\|\|([\w\.]+)\|\|'
values = 'firsttime,rdata,observable'
lasttime = <firsttime>
```

TODO

1. https://github.com/JustinAzoff/passive-dns

FireEye

This page is Under Construction

Ingress

Explore taking in data from a FireEye appliance. On page 9 of the guide[1] it looks like you can export FireEye notifications to:

1. Email
2. HTTP
3. rsyslog
4. SNMP

In the following formats:

1. Text (Normal, Concise, Extended)
2. JSON (Normal, Concise, Extended)
3. XML (Normal, Concise, Extended)

ToDo:

Export JSON -> Store JSON -> Parse JSON -> Push threat intelligence into CIF

[1] FireEye + Splunk: Intermediate Guide

https://www.fireeye.com/resources/pdfs/FireEye-Splunk-Intermediate-Guide.pdf

Sharing-Threat-Intelligence

Sharing Threat Intelligence

At some point you may want to share your threat intelligence with others. This may be public like Zeustracker (https://zeustracker.abuse.ch/) or with trusted private partners or private communities. This is a introductory guide to sharing threat intelligence.

Baseline

Method of Sharing

The most common way to share threat intelligence in 2016 is to place the threat intelligence in a CSV file and make it available via http or https with or without basic auth. A harder way to share threat intelligence in 2016 is to use SMTP and possibly GPG, which requires your partners to parse SMTP messages and possibly unencrypt (https://github.com/giovino/perl-mail-gpg-example) if encrypted.

One of the goals of the CIF project is to make it easier to digest and share threat intelligence, once familiar with CIF (which is no small feat), CIF can give you a lot of advanced capabilities essentially for free.

Most Specific Indicator

Whenever possible share the most specific indicator you have. If you have:

- URL - share the malicious URL
- IP address - share the ip address, port and protocol
- FQDN - share the FQDN

All too often someone will start with a malicious URL then resolve the A record or strip out the domain and share the IP address or domain as the malicious indicator. Due to shared hosting, compromised servers or compromised web applications, often the most specific indicator is the best indicator (most confident) of potential compromise.

Minimum Sharing

There is a bare minimum that one should strive for when sharing threat intelligence. You can share less than what is described below but the entity on the other side will have to make a lot of assumptions and these assumptions will likely lead to a decreased level of confidence in the shared threat intelligence.

Common Parameters

Parameter Name	Values	Description
observable	<string>	IP address, URI, domain
description	<string>	describe the observation

lasttime	<string>	ISO 8601 (2013-06-18T10:10:00Z)
portlist	<int>	22,25,80
protocol	<int> <string>	6 or tcp, 17 or udp

Infrastructure

```
#address,portlist,protocol,lasttime,description
"192.168.1.1","22","tcp","2013-06-18T10:10:00Z","scanner"
"192.168.10/24","80,443","tcp","2013-06-17T08:01:56Z","botnet"
```

Domain

```
#address,lasttime,description
"example.com","2013-06-16T12:00:00Z","botnet"
"car.example.com","2013-06-16T12:00:00Z","malware"
"google.com","2013-06-01T12:00:00Z","whitelist"
```

URI

```
#address,lasttime,description
"http://www.example.com/bad.php","2013-06-16T12:00:00Z","malware"
"https://controller.example.com/bad.php","2013-06-16T12:00:00Z","botnet"
```

Advanced Sharing

As you mature in your threat intelligence sharing capabilities, you may find that your partners need more than the bare minimum as described above. Below are some common parameters and an example description found in mature threat intelligence feeds.

Common Parameters

Parameter Name	Values	Description
alternativeid	<string>	usually a url pointing to the original data point (as a reference id)
alternativeid_restriction	<string>	rfc5070 (http://www.ietf.org/rfc/rfc5070.txt) (public, need-to-know, private) or TLP (http://www.us-cert.gov/tlp)
confidence	<int>	see Confidence (https://github.com/csirtgadgets/massive-octo-spice/wiki/Confidence)
description	<string>	short (1-2 space delimited word) description of the activity (eg: tdss spyeye)
restriction	<string>	rfc5070 (http://www.ietf.org/rfc/rfc5070.txt) (public, need-to-know, private) or TLP (http://www.us-cert.gov/tlp)
source	<string>	source of the feed, usually the domain where the feed is from (eg: example.com)

Description

Mature threat intelligence feeds will give a description about the data that can be found in the feed. Sometimes that description will be found in a separate document or webpage and in other cases it will be found as a header in the feed itself. Here's an example high quality description from the

Dragon Research Group (DRG) that can be found as a header in their CSV sshpwauth report.

```
# Dragon Research Group (DRG)
# sshpwauth report
# 2016-04-19 16:55:02 - 2016-04-26 16:55:02
#
# To read more about SSH password authentication issues and how to
# mitigate SSH password authentication brute force attacks based on
# report data such as this, see:
#
#  <http://www.dragonresearchgroup.org/insight/sshpwauth-tac.html>
#
# README: The sshpwauth report is for free for non-commercial use
#         ONLY.  If you wish to discuss commercial use of this
#         service, please contact the Dragon Research Group (DRG)
#         for more information.  Redistribution of the sshpwauth
#         report is prohibited without the express permission of
#         the Dragon Research Group (DRG).
#
#         This report is informational.  It is not a blacklist, but some
#         operators may choose to use it to help protect their networks
#         and hosts in the forms of automated reporting and mitigation
#         services.  The data is provided on an as-is basis with no
#         expressed warranty or guarantee of accuracy.  Use of this data
#         is at your own risk.  If you have questions about this report
#         do not hesitate to contact us by any of the means below.
#
#         The Dragon Research Group (DRG) is a volunteer research
#         organization dedicated to further the understanding of
#         online criminality and to provide actionable intelligence
#         for the benefit of the entire Internet community.
#
#               URL: <http://www.dragonresearchgroup.org>
#             email: dragon@dragonresearchgroup.org
#           PGP key: 0x47196BBF
#               IRC: irc://irc.freenode.net/drg
#           Twitter: http://twitter.com/dragonresearch
#
# Entries consist of fields with identifying characteristics of a
# a source IP address that has been seen attempting to remotely
# login to a host using SSH password authentication.  This report
# lists hosts that are highly suspicious and are likely conducting
# malicious SSH password authentication attacks.  Each entry is
# sorted according to a route origination ASN.  An entry for the
# IP address may be listed more than once if there are multiple
# origin AS (MOAS) announcements for the covering prefix.  We use
# the Team Cymru IP address to ASN mapping service to construct a
# origin AS number and name.  For details about this Team Cymru
# service, see <http://www.team-cymru.org/Services/ip-to-asn.html>.
#
# Formatting is as follows:
#
```

```
# ASN  |  ASname  |  saddr  |  utc  |  category
#
# Each field is described below.  Please note any special formatting
# rules to aid in processing this file with automated tools and scripts.
# Blank lines may be present to improve the visual display of this file.
# Lines beginning with a hash ('#') character are comment lines.  All
# other lines are report entries.  Each field is separated by a pipe
# symbol ('|') and at least two whitespace characters on either side.
#
# ASN       Autonomous system number originating a route for the entry
#           IP address. Note, 4-byte ASNs are supported and will be
#           displayed as a 32-bit integer.
#
# ASname    A descriptive network name for the associated ASN.  The
#           name is truncated to 30 characters.
#
# saddr     The source IPv4 or IPv6 address that is being reported.
#
# utc       A last seen timestamp formatted as YYYY-MM-DD HH:MM:SS
#           and in UTC time.
#
# category  Descriptive tag name for this entry.  For this report,
#           the text sshpwauth will appear.
#
```

Sharing with CIF

As mentioned above, one of CIF's goals is to make it easier to share threat intelligence. If you deploy a CIF instance and feed your threat intelligence to CIF, what capabilities does CIF give you in regard to sharing threat intelligence?

- Create users with API keys (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-tokens)
- Create groups (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Groups) to share threat intelligence selectively
- Generate feeds (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Feeds) of threat intelligence
- Support many Output_Plugins output types (https://github.com/csirtgadgets/massive-octo-spice/wiki/Introducing-the-CIF-client#format), not only CSV
- Give your partners an API (https://github.com/csirtgadgets/massive-octo-spice/wiki/API) to program against
- Whitelisting (https://github.com/csirtgadgets/massive-octo-spice/wiki/Whitelist) capabilities

Contributing-Threat-Intelligence

Contributing Threat Intelligence

One of the major reasons projects like CIF are valuable is because so many organizations and people are willing to share their visibility into active threats. Many of the data providers are crowd sourced efforts; if you find OSINT valuable you should consider giving back to the larger OSINT sharing community.

Here's a list of data providers that you can share your data with:

DShield

General Information On Submitting Logs To DShield (https://isc.sans.edu/howto.html)

Troubleshooting-CIF

Troubleshooting CIF

Basic troubleshooting steps

What can I do to if my CIF server isn't working as I expect?

1. Reboot the CIF server
2. Run the cif ping command with debug

```
$ cif -p -d
[2016-01-13T03:32:38,391Z][INFO][main:261]: starting up client...
[2016-01-13T03:32:38,392Z][INFO][main:272]: pinging: https://localhost...
[2016-01-13T03:32:38,392Z][DEBUG][CIF::SDK::Client:203]: generating ping...
[2016-01-13T03:32:38,392Z][DEBUG][CIF::SDK::Client:165]: uri created:
https://localhost/ping?
[2016-01-13T03:32:38,392Z][DEBUG][CIF::SDK::Client:166]: making request...
[2016-01-13T03:32:38,877Z][INFO][CIF::SDK::Client:170]: status: 200
[2016-01-13T03:32:38,877Z][DEBUG][CIF::SDK::Client:173]: decoding content..
roundtrip: 0.485375 ms
...
[2016-01-13T03:32:44,223Z][INFO][main:393]: done...
```

1. Make a cif query with debug

```
$ cif -q example.com -d
[2016-01-13T02:58:21,076Z][INFO][main:261]: starting up client...
[2016-01-13T02:58:21,076Z][INFO][main:296]: running search...
[2016-01-13T02:58:21,076Z][DEBUG][CIF::SDK::Client:165]: uri created:
https://localhost/observables?observable=example.com
[2016-01-13T02:58:21,076Z][DEBUG][CIF::SDK::Client:166]: making request...
[2016-01-13T02:58:21,745Z][INFO][CIF::SDK::Client:170]: status: 200
[2016-01-13T02:58:21,745Z][DEBUG][CIF::SDK::Client:173]: decoding content..
[2016-01-13T02:58:21,745Z][INFO][main:356]: search returned, formatting..
tlp  |group   |reporttime          |observable |cc|asn|confidence|tags
|description|rdata|provider      |altid_tlp|altid
amber|everyone|2015-12-21T20:01:16Z|example.com|  |   |25        |search|
|     |root@localhost|         |
amber|everyone|2015-12-21T20:01:18Z|example.com|  |   |25        |search|
|     |root@localhost|         |
...

[2016-01-13T02:58:21,757Z][INFO][main:393]: done...
```

1. Read through all the CIF logs:

```
$ tail /var/log/cif-router.log
[2016-01-13T03:00:48,136Z][12139][INFO]: staring up..
[2016-01-13T03:00:48,258Z][12141][INFO]: started, waiting for messages..

$ tail /var/log/cif-smrt.log
[2016-01-13T03:00:52,979Z][12325][INFO]: staring up...
[2016-01-13T03:00:52,996Z][12329][INFO]: delaying start for: 4min then running
every 60min there after...
[2016-01-13T03:00:52,997Z][12329][INFO]: to run immediately, set: --randomstart
0 or --testmode
[2016-01-13T03:00:52,997Z][12329][INFO]: to see the list of options, use -h

$ tail /var/log/cif-starman.log
[2016-01-13T03:00:52,233Z][12295][INFO]: starting CIF::REST
[2016-01-13T03:00:52,238Z][12297][INFO]: starting CIF::REST
[2016-01-13T03:00:52,255Z][12299][INFO]: starting CIF::REST

$ tail /var/log/cif-worker.log
[2016-01-13T03:00:50,256Z][12188][INFO]: sending ping...
[2016-01-13T03:00:50,313Z][12195][INFO]: staring worker..
[2016-01-13T03:00:50,315Z][12196][INFO]: staring worker..
...
[2016-01-13T03:00:50,337Z][12192][INFO]: starting...
```

1. Verify apache is working

```
$ curl -ik https://localhost/
HTTP/1.1 200 OK
Date: Wed, 13 Jan 2016 13:05:53 GMT
Server: Apache
Vary: Accept-Encoding
Content-Length: 671
Content-Type: text/html;charset=UTF-8

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
...
```

1. Read through the Apache logs

```
$ sudo tail /var/log/apache2/error.log
$ sudo tail /var/log/apache2/ssl_access.log
```

1. Verify ElasticSearch is working

```
$ curl -i 'http://localhost:9200/_cluster/health?pretty'
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 283

{
"cluster_name" : "elasticsearch",
"status" : "yellow",
"timed_out" : false,
"number_of_nodes" : 1,
"number_of_data_nodes" : 1,
"active_primary_shards" : 155,
"active_shards" : 155,
"relocating_shards" : 0,
"initializing_shards" : 0,
"unassigned_shards" : 155
}
```

1. Read through the ElasticSearch logs

```
$ tail /var/log/elasticsearch/elasticsearch.log
```

1. restart cif to look for errors

```
$ sudo service cif-services restart
* Stopping cif-router
[ OK ]
* Starting cif-router
[ OK ]
* Stopping cif-worker
[ OK ]
* Starting cif-worker
[ OK ]
* Stopping cif-starman
[ OK ]
* Starting cif-starman
[ OK ]
* Stopping cif-smrt
[ OK ]
* Starting cif-smrt
[ OK ]
```

1. Verify the host has enough free disk space

```
$ df -h
Filesystem                         Size  Used Avail Use% Mounted on
udev                               7.9G  4.0K  7.9G   1% /dev
tmpfs                              1.6G  444K  1.6G   1% /run
/dev/mapper/cifv2--rc6--vg-root  230G   14G  205G   7% /
none                               4.0K     0  4.0K   0% /sys/fs/cgroup
none                               5.0M     0  5.0M   0% /run/lock
none                               7.9G     0  7.9G   0% /run/shm
none                               100M     0  100M   0% /run/user
/dev/sda1                          236M   69M  155M  31% /boot
```

1. Verify the host has enough free memory

```
$ free -m
             total       used       free     shared    buffers     cached
Mem:         16047      12496       3551          0        158       2947
-/+ buffers/cache:       9389       6657
Swap:        16383         62      16321
```

Enable debug logging across all CIF services

1. Add '-d' to `CIF_DEBUGGING` in `/etc/default/cif`

```
$ cat /etc/default/cif
# Directory where the binary distribution resides
CIF_HOME=/opt/cif

PATH=$CIF_HOME/bin:$PATH

if [ -d /opt/cif/lib/perl5 ]; then
  export PERL5LIB=/opt/cif/lib/perl5
fi

# Run as this user ID and group ID
CIF_USER=cif
CIF_GROUP=cif

# data directory
DATA_DIR=/var
LOG_DIR=/var/log

# configuration directory
CONF_DIR=/etc/cif

# add -d to turn on debugging
CIF_DEBUGGING="-d"
```

1. Restart all CIF services

```
$ sudo service cif-services restart
* Stopping cif-router
[ OK ]
* Starting cif-router
[ OK ]
* Stopping cif-worker
[ OK ]
* Starting cif-worker
[ OK ]
* Stopping cif-starman
[ OK ]
* Starting cif-starman
[ OK ]
* Stopping cif-smrt
[ OK ]
* Starting cif-smrt
[ OK ]
```

1. Make a cif query with debug

```
$ cif -q example.com -d
[2016-01-13T02:58:21,076Z][INFO][main:261]: starting up client...
[2016-01-13T02:58:21,076Z][INFO][main:296]: running search...
[2016-01-13T02:58:21,076Z][DEBUG][CIF::SDK::Client:165]: uri created:
https://localhost/observables?observable=example.com
[2016-01-13T02:58:21,076Z][DEBUG][CIF::SDK::Client:166]: making request...
[2016-01-13T02:58:21,745Z][INFO][CIF::SDK::Client:170]: status: 200
[2016-01-13T02:58:21,745Z][DEBUG][CIF::SDK::Client:173]: decoding content..
[2016-01-13T02:58:21,745Z][INFO][main:356]: search returned, formatting..
tlp  |group   |reporttime          |observable |cc|asn|confidence|tags
|description|rdata|provider      |altid_tlp|altid
amber|everyone|2015-12-21T20:01:16Z|example.com|  |   |25        |search|
|     |root@localhost|         |
amber|everyone|2015-12-21T20:01:18Z|example.com|  |   |25        |search|
|     |root@localhost|         |
```

1. Read through the apache logs

```
$ sudo tail /var/log/apache2/ssl_access.log
::1 - - [13/Jan/2016:03:38:23 -1000] "GET /observables?observable=example.com
HTTP/1.1" 200 5685 "-" "cif-sdk-perl/2.00_30"

$ sudo tail /var/log/apache2/ssl_access.log
```

1. Read through the CIF logs

```
$ tail /var/log/cif-router.log
$ tail /var/log/cif-smrt.log
$ tail /var/log/cif-starman.log
$ tail /var/log/cif-worker.log
```

1. Once done troubleshooting, be sure to turn off CIF debugging and restart all the CIF services; the logging is verbose and will use up a lot of disk space.

Migration

CIFv1 Migration

Overview

Upgrades are never an exact science, and generally are not recommended unless you have the time/skills to perform them if/when things fail. We do our best to provide the utilities and doc necessarily to cover edge cases, if something goes horribly bad, it might make more sense to start fresh with a new installation and new data. This is a learning process for the project and we don't currently have the resources to debug problems that might occur during a database upgrade between versions.

That being said, as long as you follow the directions carefully, backup your archive table (external usb-drives can be handy), you should be able to recover from a failed upgrade with out losing everything.

If you do lose everything, ... it's not the end of the world, within a few weeks, your shinny new v2 instance will have lots of data in it to make lemonade with. :)

There are no "in-place" upgrades from v1 to v2, a new host must be created.

Components

API Keys

CIFv2 changes the way we've generated API tokens from a "UUID" to a random SHA1 hash. That said, it's *HIGHLY RECOMMENDED* that you skip exporting and importing your old v1 tokens and generate new ones. If you choose to import your old tokens, we've developed a helper export utility (https://github.com/collectiveintel/cif-v1/blob/master/libcif-dbi/bin/cif_apikeys_export) for v1 that exports the old tokens as cleanly as possible into JSON compatible with the v2 `cif-tokens --import` flag (via STDIN).

1. on your v1 instance

```
$ cif_apikeys_export > tokens.json
```

2. on your v2 instance:

```
$ cif-tokens --import tokens.json
```

Data Migration

The data migration is a little more complicated. Effectively we're taking data from a postgres database and re-inserting it to an elastic-search database. Along with that, we're re-mapping some of the old style fields to our new v2 schema. We've created some more helper utilities (https://github.com/csirtgadgets/massive-octo-spice/tree/master/v1migration) to aid in the process.

1. setup a new CIFv2 host
2. be sure the new v2 host has psql (tcp/5432) to the v1 database host

3. be sure to use a token generated from cif-tokens that has access to all groups on the new host (if you're using more than the default 'everyone' group, otherwise the root@localhost token will work OK)
4. run the migration tool (this will take a few days depending how much data you have):

```
$ sudo aptitude install -y postgresql-common libpq-dev postgresql-client-9.3 autoconf
$ cd massive-octo-spice/v1migration
$ sudo cpanm DBD::Pg ZMQ::FFI@0.17 Compress::Snappy --force
$ sudo cpanm --installdeps .  # be sure to install cpanm if it's not on this box
$ perl -I../src/lib -Ilib bin/migrate-data.pl -h
$ perl -I../src/lib -Ilib bin/migrate-data.pl --threads 4 --psql-host 192.168.1.1 --es-token XXXXXXX
```

This should migrate the data over in stages, while keeping track of what records it's written in the journal located at `/tmp/cif-migrate.journal`. That way, if the tool fails, or something happens, it will start where it left off.

Backup-and-Restore

To be completed.

References:

- Snapshot And Restore (http://www.elastic.co/guide/en/elasticsearch/reference/master/modules-snapshots.html)
- Knapsack (https://github.com/jprante/elasticsearch-knapsack)
- elasticdump (https://github.com/taskrabbit/elasticsearch-dump)

Monit

Monit (http://mmonit.com/monit/) is installed and configured to monitor the following processes:

- cif-worker
- cif-router
- cif-starman
- cif-smrt
- elasticsearch

You can view the Monit config files here (https://github.com/csirtgadgets/massive-octo-spice/blob/b36701cd36ae89a42bd1999428e677277eb6822a/hacking/platforms/ubuntu/cif.monit) and here (https://github.com/csirtgadgets/massive-octo-spice/blob/b36701cd36ae89a42bd1999428e677277eb6822a/hacking/platforms/ubuntu/elasticsearch

Pruning-the-ElasticSearch-database

You can use ElasticSearch Curator (https://www.elastic.co/guide/en/elasticsearch/client/curator/current/index.html) to prune the ElasticSearch database.

Example usage:

1. Installation

```
apt-get -y install python-pip
pip install elasticsearch-curator
```

1. Delete all indexes, that are older than 3 days and prefix starts with cif.observables

```
/usr/local/bin/curator --host localhost --master-only delete indices --prefix
cif.observables --older-than 3 --time-unit days --timestring '%Y.%m.%d'
```

1. Using `--dry-run` will show you what would be deleted

```
/usr/local/bin/curator --dry-run --host localhost --master-only delete indices
--prefix cif.observables --older-than 3 --time-unit days --timestring
'%Y.%m.%d'
2015-06-24 07:51:46,243 INFO      Job starting: delete indices
2015-06-24 07:51:46,316 INFO      Pruning Kibana-related indices to prevent
accidental deletion.
2015-06-24 07:51:46,317 INFO      DRY RUN MODE.  No changes will be made.
2015-06-24 07:51:46,322 INFO      DRY RUN: delete: cif.observables-2015.05.17
2015-06-24 07:51:46,327 INFO      DRY RUN: delete: cif.observables-2015.05.18
2015-06-24 07:51:46,332 INFO      DRY RUN: delete: cif.observables-2015.05.26
2015-06-24 07:51:46,337 INFO      DRY RUN: delete: cif.observables-2015.05.27
2015-06-24 07:51:46,341 INFO      DRY RUN: delete: cif.observables-2015.05.28
<SNIP>
2015-06-24 07:51:46,397 INFO      DRY RUN: delete: cif.observables-2015.06.17
2015-06-24 07:51:46,402 INFO      DRY RUN: delete: cif.observables-2015.06.18
2015-06-24 07:51:46,406 INFO      DRY RUN: delete: cif.observables-2015.06.19
2015-06-24 07:51:46,411 INFO      DRY RUN: delete: cif.observables-2015.06.20
```

Source: cif-users group few questions (https://goo.gl/fNfNeU)

Newer versions of curator don't take command line arguments. You can suss out the various config file entries that will duplicate what's here, or you can do:

```
pip install elasticsearch-curator==3.5.1
```

for the most recent version of curator that doesn't require the config file.

Development_GitHub

First Steps

1. Create a GitHub account (https://help.github.com/articles/signing-up-for-a-new-github-account/)
2. Fork the CIFv2 repository (https://help.github.com/articles/fork-a-repo/#fork-an-example-repository)
3. Sync your fork with the CIFv2 repository (https://help.github.com/articles/fork-a-repo/#keep-your-fork-synced)
4. Create a branch for your feature (https://github.com/Kunena/Kunena-Forum/wiki/Create-a-new-branch-with-git-and-manage-branches)
5. Create a pull request (https://help.github.com/articles/creating-a-pull-request/)

Ongoing

1. Sync your fork with the upstream (https://help.github.com/articles/syncing-a-fork/)

Tutorials:

- How to GitHub: Fork, Branch, Track, Squash and Pull Request (https://gun.io/blog/how-to-github-fork-branch-and-pull-request/)

CIF-Architecture-Overview

Introduction

This page presents an overview of the CIF architecture and explains how data moves through the system.

- [How CIF fetches, parses and normalizes data (CIF-Architecture-Overview#how-cif-fetches-parses-and-normalizes-data)]
- [How CIF post-processes data (CIF-Architecture-Overview#how-cif-post-processes-data)]
- [How CIF stores data (CIF-Architecture-Overview#how-cif-stores-data)]
- [How the CIF API allows data to be queried and submitted (CIF-Architecture-Overview#how-the-cif-api-allows-data-to-be-queried-and-submitted)]
- [How CIF permissions data (CIF-Architecture-Overview#how-cif-permissions-data)]
- [How CIF produces feeds of data (CIF-Architecture-Overview#how-cif-produces-feeds-of-data)]

```
                                                  cif-worker
                                                     ^  +
                                                     |  |
                                                    ZMQ-PUB
                                                     |  |
                                                     +  v
cif-smrt +---> apache2  <---> cif-starman  <--->  cif-router
                ^  +                                 +  ^
                |  |                                 |  |
                HTTP                                 HTTP
                |  |                                 |  |
                +  v                                 v  +
               client                            elasticsearch
```

How CIF fetches, parses and normalizes data

cif-smrt is a service that runs every hour with a random start time within a thirty minute window. cif-smrt uses configuration files found in `/etc/cif/rules/default` as the instructions to specify on what to download, how to parse and how to normalize.

- cif-smrt uses LWP::UserAgent to fetch the data
- cif-smrt uses RegEx, HTML::TableExtract, JSON::XS, XML:RSS, String::Tokenizer, and XML::LibXML to parse the data
- cif-smrt normalizes the data to a JSON data structure
- cif-smrt submits the JSON data structure to the CIF RESTful API interface

```
/etc/cif/rules/default/*.cfg

             +
             |
             |
             |
             v

          cif-smrt  +--->  apache2

                              +
                              |
                              |
                              |
                              v

                          cif-router
```

How CIF post-processes data

cif-worker is responsible for the post-processing of data; CIF ships with four post-processers:

- UrlResolver - extract the FQDN from a URL
- Resolver - resolve DNS records from a FQDN
- Spamhaus - query Spamhaus
- BGPWhitelist - create whitelisted CIDR ranges from IP addresses resolved from FQDNs tagged at "whitelist"

```
https://example.com/evil.htm    +--->     cif-worker

                                               +
                                               |
                                               |
                                               v

               cif-router  <----------+   example.com [lower confidence]
```

How CIF stores data

CIF uses ElasticSearch for it's data warehouse. ElasticSearch is a json document store where every field is indexed and searchable.

How the CIF API allows data to be queried and submitted

CIF uses Mojo::Base and Apache as the core for it's RESTful API (PSGI). The CIF API sits on top of the ElasticSearch API enforcing things like:

- User Permissions
- Data Limits

```
network +--> client +--> apache2 <--> cif-starman <--> cif-router
```

How CIF permissions data

CIF stamps each record with a group id. CIF tokens (API keys) are associated with Groups and have read, write attributes. The CIF API ensures that users (API keys) are limited to only returning data it has been given read access to and limiting users from writing to the CIF data store.

How CIF produces feeds of data

The CIF SDK (client) is responsible for generating CIF feeds. The primary attributes of a feed are:

- Filtered by observable type (ipv4, fqdn, url, ipv6, email)
- De-duplicated or aggregated by observable
- Whitelisting data-sets applied

The CIF client makes a query to they CIF server to retrieve a overly broad data set and then reduces said data set by the attributes above before returning the data to the user.

Note: In an all-in-one CIF server where the CIF client is on the CIF server, all the processing is completed on a single host. In a distributed environment, the CIF client is able to reduce load on the CIF server by processing data on a separate client host.

CIF-Manpage

Name

cif

Synopsis

```
$ cif [--config] [--remote] [--token] [-q] [--limit] [--feed] [--format]
example.org
$ cif --otype ipv4 --format csv --feed
$ cif --otype ipv4 --format bro --feed
```

Description

cif is a command line tool to query the collective intelligence framework for observables, to generate data feeds and to submit data.

Options

```
Options:

    -q, --query=STRING          specify a search
    --id STRING                 specify an id to retrieve
    -f, --format=FORMAT         specify the output format (Table, CSV, Json,
Snort, Bro, default: table)
    -l, --limit=INT             specify a return limit (default set at router)
    -s, --submit                submit data via STDIN (json keypairs)

    -h, --help                  this message

Filters:

    -c, --confidence=INT        by confidence (greater or equal to)
    -n, --nolog                 do not log the query
    --tags=STRING,STRING        by tags (scanner,hijacked,botnet, ...)
    --description=STRING        by description
    --cc=STRING,STRING          by country codes (RU,US, ...)
    --asn=INT,INT               by asns (1234,2445, ...)
    --otype=STRING,STRING       by observable type (ipv4,fqdn,url, ...)
    --provider=STRING,STRING    by provider
(spamhaus.org,dragonresearchgroup.org, ...)
    --application=STRING        filter based on the application field
    --rdata=STRING              by rdata
    --group=STRING              by groups (everyone,group1,group2, ...)
    --lasttime STRING           specify filter based on lasttime timestamp
(less than, format: YYYY-MM-DDTHH:MM:SSZ)
    --firsttime STRING          specify filter based on firsttime timestmap
```

```
(greater than, format: YYYY-MM-DDTHH:MM:SSZ)
    --reporttime STRING         specify filter based on reporttime timestmap
(greater than, format: YYYY-MM-DDTHH:MM:SSZ)
    --reporttime-end STRING     specify filter based on reporttime timestmap
(less than, format: YYYY-MM-DDTHH:MM:SSZ)

    --today                     auto-sets reporttime to today, 00:00:00Z (UTC)

    --last-hour                 auto-sets reporttime to the beginning of the
previous full hour
                                and reporttime-end to end of previous full hour

    --last-day                  auto-sets reporttime to 23 hours and 59 seconds
ago (current time UTC)
                                and reporttime-end to "now"

    --days                      number of days to go back
    --feed                      generate a feed of data, meaning deduplicated
and whitelisted
    --whitelist-confidence=INT  by confidence (greater or equal to) (default
25)
    --whitelist-limit=INT       specify a return limit of generated whitelist
(default 50000)


Advanced Options:

    -C, --config=STRING         specify a config file
    -d, --debug                 print debug output to stdout
    -p, --ping                  ping the router for testing connectivity
    -T, --token=STRING          specify an access token
    -R, --remote=STRING         specify the remote, default: https://localhost
    -v, --verbosity             -v (level 1) through -vvvvvv (level 6)
    --no-verify-ssl             turn off SSL/TLS verification

Formatting Options:

    --sortby                    sort output, default: lasttime
    --sortby-direction          sortby direction, default: asc
    --aggregate                 aggregate output based on field (ie:
observable)
    --fields                    specify output fields [default:
tlp,group,reporttime,observable,cc,asn,confidence,tags,description,rdata,provid
er,altid_tlp,altid]

Ping Options:
    --ttl=INT                   specify number of pings to send, default: 4
                                (0 infinite, halt with SIGINT or CTRL+C)
```

Files

~/.cif.yml

Advanced Examples

```
$ cif -q 130.201.0.2
$ cif -q 130.201.0.0/16
$ cif -q 2001:4860:4860::8888
$ cif -q example.com
$ cif -q 'http://www.example.com'
$ cif -q 'john@example.com'
$ cif -q bf9d457bcd702fe836201df1b48c0bec

$ cif --tags botnet,zeus -c 85
$ cif --application vnc,ssh --asns 1234 --cc RU,US
$ cif -q example.com --tags botnet,zeus -c 85 --limit 50

$ cif --otype ipv4 --aggregate observable --today

$ cif --feed --otype ipv4 -c 85 -f csv
$ cif --feed --otype fqdn -c 95 --tags botnet -f csv
$ cif --feed --otype url -c 75 --today -f csv
```

CIF-Router-Manpage

Name

cif-router

Synopsis

`cif-router [options] [status|start|stop|restart|reload]`

```
$ cif-router -C /etc/cif/cif-router.conf
$ cif-router -D start -C /etc/cif/cif-router.conf -p /var/run/cif-router.pid
```

Description

cif-router provides the broker mechanism between the client/web framework (`cif`, apache2, `cif-starman`) and the elastic search backend. It also is responsible for seed raw messages into the `cif-worker` pipeline and returning results to the client.

Options

```
Options:
   -F,  --frontend=STRING   specify the frontend binding, default: tcp://*:4961
   -B,  --backend=STRING    specify the backend binding, default: tcp://*:4962
   -C,  --config=FILE       specify cofiguration file, default: /etc/cif/cif-
router.conf
   -d,  --debug             turn on debugging (max verbosity)
   -v+, --verbosity         turn up verbosity
   -h,  --help              this message

Daemon Options:
   -D, --daemon            run as daemon
   -u, --user              run daemon as user, default: cif
   -g, --group             run daemon as group, default: cif
   -p, --pid               pidfile location, default: /var/run/cif-router.pid
   --logging               turn on logging [to file]
   --logfile               logfile location, default: /var/log/cif-router.log

Notification Options:
   --notify:               turn on notification, default: off.
   --notify-to:            default: root@localhost
   --notify-from:          default: cif
   --notify-subj:          default: [cif-router] ERROR
   --notify-level:         default: error

Advanced Options:
   -A, --auth              specify authorization plugin, default: dummy

Storage:
   -s, --storage           default: elasticsearch
   --storage-host          default: localhost:9200
```

Files

/etc/cif/cif-router.conf

cif-smrt-FAQ

Q1: How often does cif-smrt run?

A1: Hourly

Q2: Does cif-smrt download the same observation 24 times a day?

A2: No cif-smrt aggregates observations that are identical per day starting at 00:00. This is done using a journal of file hashes (/var/smrt/cache/*.log) where cif-smrt effectively hashes (sha1) a feed line as it’s pulled in to verify if cif-smrt has seen that line of text before. If cif-smrt has seen it, it ignores the entry. (Reference (https://groups.google.com/forum/#!topic/ci-framework/o-Wv5Z6cRhI))

CIF-Smrt-Manpage

Name

cif-smrt

Synopsis

`cif-smrt [options]`

```
$ cif-smrt -C /etc/cif/cif-smrt.yml
$ cif-smrt -C /etc/cif/cif-smrt.yml -p /var/run/smrt.pid -D start
$ cif-smrt -r /etc/cif/rules/default -D start
```

Description

cif-smrt is an application that uses the feed configuration files to download, parse and ingest data into CIF. It is typically run hourly in daemon mode hourly to consistently seek updated data sources.

Options

```
 Options:
    -C,  --config=FILE       specify cofiguration file, default: /etc/cif/cif-smrt.yml
    -d,  --debug             turn on debugging (max verbosity)
    -v+, --verbosity         turn up verbosity
    -h,  --help              this message

    -r, --rule=STRING       specify a rule or a rules directory, default: /etc/cif/rules/default
    -f, --feed=STRING       specify a feed (within a rule)
    -R, --remote=STRING     specify a remote to connect to, default http://localhost:5000
    -T, --token=STRING      specify a default token/apikey to use
    --not-before=STRING     specify a time to begin processing the data "[today|yesterday|X days ago]"

    --limit=INT             limit parsing to a subset of records (useful for debugging)

    --proxy                 specify a proxy address for cif-smrt to use in fetching feeds
    --https-proxy           specify a proxy for cif-smrt to use for feeds hosted on https

 Daemon Options:
    -D, --daemon            run as daemon
    -u, --user              run daemon as user, default: cif
    -g, --group             run daemon as group, default: cif
    -p, --pid               pidfile location, default: /var/run/smrt.pid

    --randomstart           random start delay, default: 30 min
    --interval              runtime interval, default: 60 min

    --testmode              run now, overrides randomstart

    --logfile:              logfile location, default: /var/log/cif-smrt.log
    --logging:              turn on logging [to file]

 Notification Options:
    --notify:               turn on notification, default: off.
    --notify-to:            default: root@localhost
    --notify-from:          default: cif
    --notify-subj:          default: [cif-smrt] ERROR
    --notify-level:         default: error

 Advanced Options:
    -M, --meta              apply metadata processors, default: 0
    -c, --clean             clear cache
    -P, --cache             cache location, default /var/smrt/cache
```

Files

```
/etc/cif/cif-smrt.yml
/etc/cif/rules/default
```

CIF-Tokens-Manpage

Name

cif-tokens

Synopsis

`cif-tokens [options]`

```
$ cif-tokens --new --username me@example.com --expires 2016-07-01 --admin
$ cif-tokens --new --username root --groups everyone,groupA,groupB
$ cif-tokens --username me@example.com --write-enable
```

Description

cif-tokens is an application to manage API keys associated with a CIF instance.

Options

```
Options:

    -h, --help      this message

    --username      specify a username
    --admin         set the admin flag for the user
    --read          set read permissions for a token
    --write         set write permissions for a token
    --expires       set an expiration date for the token
    --groups        specify the groups for the user (default: everyone)

Actions:

    --new           generate a new token
    --delete        remove token
    --revoke        revoke a user / token
    --import        import tokens list from v1 instance
(bin/cif_apikeys_export) using STDIN
    --import-path   specify a path to read for importing tokens (aka: apikeys
in v1)

    --write-enable  enable write access for a specified user / token
    --write-disable disable write access for a specified user / token

Advanced:

    --generate-config-path      generate a new config with token
    --generate-config-remote    default: https://localhost
    --generate-config-tls       default: true

Storage:

    --storage       default: elasticsearch
    --storage-host  default: localhost:9200
```

CIF-Worker-Manpage

Name

cif-worker

Synopsis

`cif-worker [options]`

```
$ cif-worker -C /etc/cif/cif-worker.yml
$ cif-worker -D start -C /etc/cif/cif-worker.yml -p /var/run/cif-worker.pid
```

Description

`cif-worker` is the "analytics" pipeline subscribed from `cif-router`. It's responsible for processing messages and generating new intelligence from those messages.

- Resolves FQDN's from URLs
- Resolves IP addresses from FQDNs

Options

```
 Options:
    --remote=STRING         specify a remote to connect to, default
tcp://localhost:4961
    --token=STRING          specify a default token/apikey to use
    --publisher=STRING      specify a remote publisher to connect to and
receive data, default tcp://localhost:4963

    -C,  --config=FILE       specify cofiguration file, default: /etc/cif/cif-
worker.yml
    -d,  --debug             turn on debugging (max verbosity)
    -v+, --verbosity         turn up verbosity
    -h,  --help              this message

 Daemon Options:
    -D, --daemon            run as daemon
    -u, --user              run daemon as user, default: cif
    -g, --group             run daemon as group, default: cif
    -p, --pid               pidfile location, default: /var/run/cif-worker.pid
    --logging               turn on logging [to file]
    --logfile               logfile location, default: /var/log/cif-worker.log

 Notification Options:
    --notify:               turn on notification, default: off.
    --notify-to:            default: root@localhost
    --notify-from:          default: cif
    --notify-subj:          default: [cif-worker] ERROR
    --notify-level:         default: error
```

Files

/etc/cif/cif-worker.yml

Debian7

Overview

This installation generally takes 15-30min on hardware with more than 4 cores. This is due to the CPAN dependencies that are being compiled and tested. Someday maybe someone will contrib .deb ... :+1:

Setting up the Environment

```
$ sudo apt-get install -y curl cpanminus build-essential
$ sudo cpanm --self-upgrade Regexp::Common http://search.cpan.org/CPAN/authors/id/S/SH/SHERZODR/Config-Simple-4.59.tar.gz
$ ./configure --enable-geoip --sysconfdir=/etc/cif --localstatedir=/var --prefix=/opt/cif
$ sudo make debian7
$ make && sudo make deps
$ make test
$ sudo make install
$ sudo make fixperms-rules
$ make elasticsearch
```

Bind Interface

Bind Forwarding

1. modify `/etc/bind/named.conf.options` to point at public-dns

```
options {
    ...
    forward only;
    forwarders {
        8.8.8.8;
        8.8.4.4;
    };
    ...
};
```

Forwarder Whitelisting

1. verify `/etc/bind/named.conf.local`

```
// bypass any forwarders

zone "cymru.com" {
    forward only;
    type forward;
    forwarders { };
};

zone "zen.spamhaus.org" {
    forward only;
    type forward;
    forwarders { };
};

zone "dbl.spamhaus.org" {
    forward only;
    type forward;
    forwarders { };
};
```

Bind Testing

1. reload bind
2. verify bind is working properly

   ```
   $ dig ns1.google.com
   ```

Apache PSGI Interface

Apache is the default configured gateway to `cif-router`, other solutions such as Nginx (PSGINginx) can also be used with a little different configuration.

1. configure the default-ssl site

   ```
   <IfModule mod_ssl.c>
   <VirtualHost _default_:443>
       ServerAdmin webmaster@localhost
   +   Include /etc/apache2/cif.conf

       DocumentRoot /var/www
       ...
   ```

2. configure the api configuration

   ```
   <Location "/api">
       SetHandler perl-script
       PerlResponseHandler Plack::Handler::Apache2
       PerlSetVar psgi_app /opt/cif/bin/cif.psgi
   </Location>
   ```

3. restart apache

```
service apache2 restart
```

Testing

Router

1. start cif-router

```
$ sudo service cif-router start
```

1. test connectivity to the router

```
$ curl -w "\n" -X GET 'http://localhost/api/_ping?token=1234'
{"timestamp":[1400585768,265584]}
```

Smrt

1. perform an initial `cif-smrt` test run

```
$ sudo -u cif cif-smrt --randomstart 0 --consolemode -d -r
/opt/cif/etc/rules/default
[2014-04-19T16:00:51,868Z][INFO]: cleaning up tmp...
[2014-04-19T16:00:52,012Z][INFO]: generating ping request...
[2014-04-19T16:00:52,077Z][INFO]: sending ping...
[2014-04-19T16:00:52,089Z][INFO]: ping returned
[2014-04-19T16:00:52,106Z][INFO]: processing: bin/cif-smrt -d -r
/opt/cif/etc/rules/default/bruteforceblocker.cfg -f ssh
[2014-04-19T16:00:52,427Z][INFO]: starting at: 2014-04-19T00:00:00Z
[2014-04-19T16:00:52,431Z][INFO]: processing...
[2014-04-19T16:00:54,532Z][INFO]: building events: 1273
[2014-04-19T16:00:55,335Z][INFO]: sending: 78
[2014-04-19T16:00:55,955Z][INFO]: took: ~0.921849
[2014-04-19T16:00:55,956Z][INFO]: rate: ~84.6125558524227 o/s
[2014-04-19T16:00:55,956Z][INFO]: processing: bin/cif-smrt -d -r
/opt/cif/etc/rules/default/drg.cfg -f ssh
...
```

1. start cif-smrt

   ```
   $ sudo service cif-smrt start
   ```

References

- http://www.spamhaus.org/zen
- http://www.spamhaus.org/dbl
- http://www.spamhaus.org/faq/answers.lasso?section=DNSBL%20Usage
- http://www.team-cymru.org/Services/ip-to-asn.html
- http://www.bind9.net/BIND-FAQ

Distributed-Environment

(to be done... by Audrius if we can trick him into it) ;)

Feeds-Example

```
#!/bin/bash

LIMIT=5000
CONFIDENCE="75,85,95"
BIN=/usr/local/bin/cif-feed -d

set -e

/usr/local/bin/cif-feed --otype ipv4 --confidence $CONFIDENCE --tags scanner --limit $LIMIT
/usr/local/bin/cif-feed --otype ipv4 --confidence $CONFIDENCE --tags botnet --limit $LIMIT
/usr/local/bin/cif-feed --otype ipv4 --confidence $CONFIDENCE --tags malware,phishing --limit $LIMIT

$BIN --otype ipv4 --confidence 95 --tags hijacked --limit $LIMIT

/usr/local/bin/cif-feed --otype fqdn --confidence $CONFIDENCE -d --tags malware,phishing --limit $LIMIT
/usr/local/bin/cif-feed --otype fqdn --confidence $CONFIDENCE -d --tags botnet --limit $LIMIT
```

FireFox

https://code.google.com/p/collective-intelligence-framework/w/edit/ClientInstall_Browser_v1

paloalto

Overview

References

https://isc.sans.edu/forums/diary/Subscribing+to+the+DShield+Top+20+on+a+Palo+Alto+Networl

PerlBrew

Using the latest version of perl can drastically improve performance. This is not required, but highly recommended. Perlbrew (http://perlbrew.pl/Perlbrew-In-Shell-Scripts.html) will compile the latest version of perl on your system, the process takes anywhere from 15-45min depending on system resources.

via CommandLine

```
$ curl -L http://cpanmin.us | perl - --sudo App::cpanminus
$ export PERLBREW_ROOT=/opt/perl5/perlbrew
$ cpanm -n -f -q App::perlbrew
$ perlbrew init
$ source $PERLBREW_ROOT/etc/bashrc
$ perlbrew install -v 5.18.2 -n -Dusethreads
$ perlbrew install-cpanm
$ echo "export PERLBREW_ROOT=/opt/perl5" >> ~/.profile
$ echo 'source ${PERLBREW_ROOT}/etc/bashrc' >> ~/.profile
```

via Bash

```
#!/bin/bash

set -e

PERL_VERSION=perl-5.18.2
PERLBREW_ROOT=/opt/perl5/perlbrew

curl -L http://cpanmin.us | perl - --sudo App::cpanminus
cpanm -n -f -q App::perlbrew

PERLBREW_ROOT=${PERLBREW_ROOT} perlbrew init

. ${PERLBREW_ROOT}/etc/bashrc
PERLBREW_ROOT=${PERLBREW_ROOT} perlbrew install -v ${PERL_VERSION} -n -
Dusethreads
PERLBREW_ROOT=${PERLBREW_ROOT} perlbrew install-cpanm

echo "source ${PERLBREW_ROOT}/etc/bashrc" >> ${HOME}/.profile
```

PlatformDebian7x

see the Ubuntu Guide (PlatformUbuntu12).

PlatformGuides

Platform Guides

Ubuntu LTS is the operating system in which CIF is developed against and is the most commonly used. RHEL and CentOS are a derivative is the second most common platform used by the community, but lags in community support.

Contributions welcome! (https://github.com/csirtgadgets/massive-octo-spice/issues/new).

Routers

- (stable) Ubuntu 12 (PlatformUbuntu12)
- (testing) Ubuntu 14 (PlatformUbuntu14)

Client SDK's

- Ubuntu12 (SDKUbuntu12)
- Ubuntu14 (SDKUbuntu14)

Fine Print

bleeding-edge style distro's (eg: release cycles less than 18-24months, Fedora, normal ubuntu, etc...) are highly discouraged and are generally not supported

PlatformUbuntu

Overview

This installation generally takes 5-10min on hardware with more than 8 cores. Generally you'd want something with at-least 16GB ram and 8cores.

Setting up the Environment

Because @giovino is so awesome, the helper script will configure apache2, bind and install CIF to `/opt/cif` for you, as well as install any required dependencies too!

1. Bash the EasyButton!(tm)

```
$ curl -Ls https://raw.githubusercontent.com/csirtgadgets/massive-octo-spice/master/hacking/platforms/easybutton_curl.sh | sudo bash -
$ sudo chown `whoami`:`whoami` ~/.cif.yml
```

Testing

1. test connectivity to the router

```
$ cif -p
roundtrip: 0.518286 ms
roundtrip: 0.487317 ms
roundtrip: 0.47499 ms
roundtrip: 0.518493 ms
```

1. perform an initial `cif-smrt` test run

```
$ sudo service monit stop
$ sudo service cif-smrt stop
$ sudo -u cif /opt/cif/bin/cif-smrt --testmode
[2014-10-21T15:17:10,668Z][INFO][main:322]: cleaning up tmp: /var/smrt/cache
[2014-10-21T15:17:10,691Z][DEBUG][main:294]: id4.us - ssh
[2014-10-21T15:17:10,691Z][INFO][main:295]: processing: /opt/cif/bin/cif-smrt -d -r /etc/cif/rules/default/1d4_us.yml -f ssh
[2014-10-21T15:17:10,692Z][INFO][CIF::Smrt:92]: starting at: 2014-10-21T00:00:00Z
[2014-10-21T15:17:10,692Z][DEBUG][CIF::Smrt:97]: fetching...
...
```

1. re-start cif-smrt

   ```
   $ sudo service cif-smrt start
   $ sudo service monit start
   ```

2. test out a query:

```
$ cif --cc US
$ cif --cc CN
$ cif --tags scanner --cc us
$ cif --otype ipv4 --cc cn
```

1. checkout the SDK Guides (SDK) to setup a client locally.

PlatformUbuntu12

Overview

This installation generally takes 15-30min on hardware with more than 4 cores. This is due to the CPAN dependencies that are being compiled and tested. Someday maybe someone will contrib a PPA... :+1:

Setting up the Environment

Because @giovino is so awesome, the helper script will configure apache2, bind and install CIF to /opt/cif for you, as well as install any required CPAN dependencies too!

If you find @giovino in the wild, buy him a beer.

```
$ tar -zxvf cif-2.xx.xx.tar.gz
$ cd cif-2.xx.xx
$ sudo bash ./hacking/platforms/easybutton.sh
```

Notes

- When it asks you what type of mail server you want setup, you can choose 'Internet', just make sure it's fire-walled appropriately and you re-configure it post install to match your local host policy.

Testing

Router

1. start cif-router

```
$ sudo service cif-router start
```

1. test connectivity to the router

```
$ curl -k -w "\n" -X GET 'https://localhost:443/api/_ping?token=1234'
{"timestamp":[1400585768,265584]}
```

Smrt

1. perform an initial `cif-smrt` test run

```
$ sudo /opt/cif/bin/cif-smrt --testmode -d
[2014-04-19T16:00:51,868Z][INFO]: cleaning up tmp...
[2014-04-19T16:00:52,012Z][INFO]: generating ping request...
[2014-04-19T16:00:52,077Z][INFO]: sending ping...
[2014-04-19T16:00:52,089Z][INFO]: ping returned
[2014-04-19T16:00:52,106Z][INFO]: processing: bin/cif-smrt -d -r
/opt/cif/etc/rules/default/bruteforceblocker.cfg -f ssh
[2014-04-19T16:00:52,427Z][INFO]: starting at: 2014-04-19T00:00:00Z
[2014-04-19T16:00:52,431Z][INFO]: processing...
[2014-04-19T16:00:54,532Z][INFO]: building events: 1273
[2014-04-19T16:00:55,335Z][INFO]: sending: 78
[2014-04-19T16:00:55,955Z][INFO]: took: ~0.921849
[2014-04-19T16:00:55,956Z][INFO]: rate: ~84.6125558524227 o/s
[2014-04-19T16:00:55,956Z][INFO]: processing: bin/cif-smrt -d -r
/opt/cif/etc/rules/default/drg.cfg -f ssh
...
```

1. start cif-smrt

   ```
   $ sudo service cif-smrt start
   ```

PlatformUbuntu14

Overview

This installation generally takes 15-30min on hardware with more than 4 cores. This is due to the CPAN dependencies that are being compiled and tested. Someday maybe someone will contrib a PPA... :+1:

Setting up the Environment

Because @giovino is so awesome, the helper script will configure apache2, bind and install CIF to /opt/cif for you, as well as install any required CPAN dependencies too!

If you find @giovino in the wild, buy him a beer.

```
$ sudo apt-get install -y htop build-essential automake autoconf git
$ git clone -b master https://github.com/csirtgadgets/massive-octo-spice
$ cd massive-octo-spice
$ bash autogen.sh
$ sudo bash ./hacking/platforms/easybutton.sh
```

Notes

- When it asks you what type of mail server you want setup, you can choose 'Internet', just make sure it's fire-walled appropriately and you re-configure it post install to match your local host policy.

Testing

Router

1. start cif-router

```
$ sudo service cif-router start
```

1. test connectivity to the router

```
$ curl -k -w "\n" -X GET 'https://localhost:443/api/_ping?token=1234'
{"timestamp":[1400585768,265584]}
```

Smrt

1. perform an initial `cif-smrt` test run

```
$ sudo /opt/cif/bin/cif-smrt --testmode -d -M
[2014-04-19T16:00:51,868Z][INFO]: cleaning up tmp...
[2014-04-19T16:00:52,012Z][INFO]: generating ping request...
[2014-04-19T16:00:52,077Z][INFO]: sending ping...
[2014-04-19T16:00:52,089Z][INFO]: ping returned
[2014-04-19T16:00:52,106Z][INFO]: processing: bin/cif-smrt -d -r
/opt/cif/etc/rules/default/bruteforceblocker.cfg -f ssh
[2014-04-19T16:00:52,427Z][INFO]: starting at: 2014-04-19T00:00:00Z
[2014-04-19T16:00:52,431Z][INFO]: processing...
[2014-04-19T16:00:54,532Z][INFO]: building events: 1273
[2014-04-19T16:00:55,335Z][INFO]: sending: 78
[2014-04-19T16:00:55,955Z][INFO]: took: ~0.921849
[2014-04-19T16:00:55,956Z][INFO]: rate: ~84.6125558524227 o/s
[2014-04-19T16:00:55,956Z][INFO]: processing: bin/cif-smrt -d -r
/opt/cif/etc/rules/default/drg.cfg -f ssh
...
```

1. start cif-smrt

   ```
   $ sudo service cif-smrt start
   ```

PlatformUbuntuManual

Overview

This installation generally takes 5-10min on hardware with more than 8 cores. Generally you'd want something with at-least 16GB ram and 8cores.

Setting up the Environment

1. Setup the Dependencies

```
$ sudo apt-get update && sudo apt-get upgrade -y && sudo apt-get install -y
htop build-essential automake autoconf git
```

1. Download the latest CIF release (https://github.com/csirtgadgets/massive-octo-spice/releases)
2. Un-tar the release and smash the EasyButton(tm)

```
$ wget https://github.com/csirtgadgets/massive-octo-spice/archive/2.00.00-
rc.4.tar.gz -O massive-octo-spice-2.00.00-rc.4.tar.gz
$ tar -zxvf massive-octo-spice-2.00.00-rc.4.tar.gz
$ cd massive-octo-spice--2.00.00-rc.4
$ VERSION=2.00.00-rc.4 bash autogen.sh
$ sudo bash ./hacking/platforms/easybutton.sh
$ sudo chown `whoami`:`whoami` ~/.cif.yml
```

Testing

1. test connectivity to the router

```
$ cif -p
roundtrip: 0.518286 ms
roundtrip: 0.487317 ms
roundtrip: 0.47499 ms
roundtrip: 0.518493 ms
```

1. perform an initial `cif-smrt` test run

```
$ sudo service monit stop
$ sudo service cif-smrt stop
$ sudo -u cif /opt/cif/bin/cif-smrt --testmode
[2014-10-21T15:17:10,668Z][INFO][main:322]: cleaning up tmp: /var/smrt/cache
[2014-10-21T15:17:10,691Z][DEBUG][main:294]: id4.us - ssh
[2014-10-21T15:17:10,691Z][INFO][main:295]: processing: /opt/cif/bin/cif-smrt -
d -r /etc/cif/rules/default/1d4_us.yml -f ssh
[2014-10-21T15:17:10,692Z][INFO][CIF::Smrt:92]: starting at: 2014-10-
21T00:00:00Z
[2014-10-21T15:17:10,692Z][DEBUG][CIF::Smrt:97]: fetching...
...
```

1. re-start cif-smrt

   ```
   $ sudo service cif-smrt start
   $ sudo service monit start
   ```

2. test out a query:

```
$ cif --cc US
$ cif --cc CN
$ cif --tags scanner --cc us
$ cif --otype ipv4 --cc cn
```

1. checkout the SDK Guides (SDK) to setup a client locally.

Probability

Probability of Risk

Probability of Risk describes how likely an "observable" is specifically meant to cause harm. Unlikely describes services such as Facebook, Google and Microsoft. Certain describes services such as a specific phishing or botnet url or a known to be harmful binary hash.

(97 - 100) Certain

- The very specific observable is ONLY used to produce harm.
- botnet urls, domains, binary hashes

(61 - 96) Likely

- domains resolved from urls
- ip's resolved from domains

(26 - 60) Possible

- Virtual hosting providers
- searches

(0 - 25) Unlikely or Rare

- Facilitating Infrastructure (application providers such as DNS, SMTP)
- Godaddy, secureserver.net
- Google, Facebook, Netflix

Notes

https://en.wikipedia.org/wiki/Risk_Matrix
https://en.wikipedia.org/wiki/Standard_deviation

PSGIApache2

Overview

There are many different ways to provide a simple PSGI (http://plackperl.org/) interface to the cif-router interface which uses ZeroMQ (http://zeromq.org). This document describes how to do this via apache2. Other ways include Nginx (PSGINginx).

Configuration

- install mod-perl
- configure the default-ssl site

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerAdmin webmaster@localhost
+   Include /etc/apache2/cif.conf

    DocumentRoot /var/www
    ...
```

`/etc/apache2/sites-available/default-ssl` under ubuntu

PSGINginx

Nginx Overview

Nginx acts as a proxy to a native running PSGI application. It requires that either plackup, starman (https://github.com/miyagawa/Starman) or some other PSGI handler (http://plackperl.org/) is running behind the scenes.

Plack

1. start plackup

```
$ sudo plackup /opt/cif/bin/cif.psgi --path /api
HTTP::Server::PSGI: Accepting connections at http://0:5000/
```

Starman

1. start starman

Configuration

1. configure `/etc/nginx/sites-enabled/cif.conf`

```
server {
    server_name   myapp.example.com;
    listen 80;
    location /api {
      proxy_set_header Host $http_host;
      proxy_set_header X-Forwarded-Host $http_host;
      proxy_set_header X-Real-IP $remote_addr;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
      proxy_pass http://localhost:5000;
    }
}
```

1. restart nginx
2. ping the interface

```
$ curl -w "\n" -X GET 'http://localhost/api/_ping?token=1234'
{"timestamp":[1400585768,265584]}
```

References

- http://www.matt-peters.com/blog/?p=35

QuickStart

Quick Start

Ubuntu LTS is the operating system in which CIF is developed against and is the most commonly used. RHEL and CentOS are a derivative is the second most common platform used by the community, but lags in community support.

Contributions welcome! (https://github.com/csirtgadgets/massive-octo-spice/issues/new).

Routers

- (stable) Ubuntu 12 (PlatformUbuntu12)
- (testing) Ubuntu 14 (PlatformUbuntu14)

Client SDK's

- Ubuntu12 (SDKUbuntu12)
- Ubuntu14 (SDKUbuntu14)

Fine Print

bleeding-edge style distro's (eg: release cycles less than 18-24months, Fedora, normal ubuntu, etc...) are highly discouraged and are generally not supported

SDK

SDKs

The SDKs (Software Development Kits) are meant to be a thin programming layer between the CIF API and your application. The SDK goal is to function using the minimum dependencies required to make a connection and provide data to/from your application and a cif-router.

If you would like to contribute to, or create a SDK in another language, see the References section below for more ideas.

- Perl (https://github.com/csirtgadgets/cif-sdk-perl)
- Python (https://github.com/csirtgadgets/cif-sdk-python)
- Javascript (https://github.com/csirtgadgets/js-cif-sdk)

Examples

Ping

```
$ cif -p --remote 'https://localhost' --token 1234
$ cif -p --no-verify-ssl --remote 'https://localhost' --token 1234
```

Search

```
$ cif -q example.com
$ cif -q 1.2.3.4
```

Submit

```
echo
'{"observable":"example.com","tlp":"amber","confidence":"85","tags":"malware","
provider":"me.com","group":"everyone"}' | /usr/local/bin/cif --no-verify-ssl --
remote 'https://localhost' -s --token 1234...
```

Sample Config [YAML]

```
# ~/.cif.yml
client:
  remote: https://localhost
  token: 1234
  no_verify_ssl: true
```

References

These are just references, none of these projects are in any way affiliated with the CSIRT Gadgets Foundation or the CIF project.

- Splunk SDKs (https://github.com/splunk/?query=sdk)
- PayPal SDKs (https://github.com/paypal?query=sdk)
- ElasticSearch SDKs (https://github.com/elasticsearch/?query=lang)

SDKUbuntu12

Overview

Perl

Python

Sharing

https://code.google.com/p/collective-intelligence-framework/wiki/Sharing_Threat_Intelligence_v1

https://code.google.com/p/collective-intelligence-framework/wiki/Recipe_FederatedSharing_v1

Table-of-Contents

Introduction

- What is the Collective Intelligence Framework? (https://github.com/csirtgadgets/massive-octo-spice/wiki/What-is-the-Collective-Intelligence-Framework%3F)
- Getting Started (https://github.com/csirtgadgets/massive-octo-spice/wiki)
- QuickStart (https://github.com/csirtgadgets/massive-octo-spice/wiki/QuickStart)

Installation

- PlatformGuides (https://github.com/csirtgadgets/massive-octo-spice/wiki/PlatformGuides)
- (stable) Ubuntu 14.04 LTS (https://github.com/csirtgadgets/massive-octo-spice/wiki/PlatformUbuntu)
- (stable) AWS Guide (https://github.com/csirtgadgets/massive-octo-spice/wiki/AWS%20Guide)
- Debian7 (https://github.com/csirtgadgets/massive-octo-spice/wiki/Debian7)
- PlatformDebian7x (https://github.com/csirtgadgets/massive-octo-spice/wiki/PlatformDebian7x)
- PlatformUbuntu12 (https://github.com/csirtgadgets/massive-octo-spice/wiki/PlatformUbuntu12)
- PlatformUbuntu14 (https://github.com/csirtgadgets/massive-octo-spice/wiki/PlatformUbuntu14)

Clients

- Google Chrome (https://github.com/csirtgadgets/massive-octo-spice/wiki/Chrome)
- Firefox (https://github.com/csirtgadgets/massive-octo-spice/wiki/FireFox)

API

- API (https://github.com/csirtgadgets/massive-octo-spice/wiki/API)
- API Observables (https://github.com/csirtgadgets/massive-octo-spice/wiki/APIObservables)
- API Query Expanded (https://github.com/csirtgadgets/massive-octo-spice/wiki/APIQueryExpanded)
- SDK (https://github.com/csirtgadgets/massive-octo-spice/wiki/SDK)
- SDK Ubuntu 12.04 (https://github.com/csirtgadgets/massive-octo-spice/wiki/SDKUbuntu12)

Feeds

- Feeds Example (https://github.com/csirtgadgets/massive-octo-spice/wiki/Feeds-Example)
- Parsing Feeds (https://github.com/csirtgadgets/massive-octo-spice/wiki/ParsingFeeds)
- Parsing Feeds Tutorial (https://github.com/csirtgadgets/massive-octo-spice/wiki/Parsing-Feeds-Tutorial)

Integration

- Integration Guides (https://github.com/csirtgadgets/massive-octo-spice/wiki/IntegrationGuides)
- Kibana (https://github.com/csirtgadgets/massive-octo-spice/wiki/KibanaGuide)
- Bro (https://github.com/csirtgadgets/massive-octo-spice/wiki/Bro)

- Snort (https://github.com/csirtgadgets/massive-octo-spice/wiki/Snort)
- Bind (https://github.com/csirtgadgets/massive-octo-spice/wiki/Bind)
- PassiveDNS (https://github.com/csirtgadgets/massive-octo-spice/wiki/PassiveDNS)
- TippingPoint (https://github.com/csirtgadgets/massive-octo-spice/wiki/TippingPoint)

Cookbook

- Book (https://github.com/csirtgadgets/massive-octo-spice/wiki/Book)
- Sharing (https://github.com/csirtgadgets/massive-octo-spice/wiki/Sharing)

Help

- FAQ (https://github.com/csirtgadgets/massive-octo-spice/wiki/FAQ)
- Troubleshooting (https://github.com/csirtgadgets/massive-octo-spice/wiki/Troubleshooting)

Advanced

- Exploring the file system (https://github.com/csirtgadgets/massive-octo-spice/wiki/Exploring-the-file-system)
- Exploring the network services (https://github.com/csirtgadgets/massive-octo-spice/wiki/Exploring-the-listening-network-services)
- Exploring the software installed (https://github.com/csirtgadgets/massive-octo-spice/wiki/Exploring-the-software-packages-installed)

Development

- Getting Involved (https://github.com/csirtgadgets/massive-octo-spice#getting-involved)
- Tutorial: CIF development using vagrant (https://github.com/csirtgadgets/massive-octo-spice/wiki/Tutorial:-CIF-development-using-vagrant)
- Vagrant prerequisites (https://github.com/csirtgadgets/massive-octo-spice/wiki/Vagrant-prerequisites)

Orphaned

- Monit (https://github.com/csirtgadgets/massive-octo-spice/wiki/Monit)
- PerlBrew (https://github.com/csirtgadgets/massive-octo-spice/wiki/PerlBrew)
- PSGIApache2 (https://github.com/csirtgadgets/massive-octo-spice/wiki/PSGIApache2)
- PSGINginx (https://github.com/csirtgadgets/massive-octo-spice/wiki/PSGINginx)
- TestRouter (https://github.com/csirtgadgets/massive-octo-spice/wiki/TestRouter)
- TestSmrt (https://github.com/csirtgadgets/massive-octo-spice/wiki/TestSmrt)

Tag-Definitions

Botnet

The botnet assessment depicts:

- typically a host used to control another host or malicious process
- matching traffic would usually indicate infection
- typically used to identify compromised hosts

Exploit / Malware

The malware assessment depicts:

- typically a host used to exploit and/or drop malware to a host for the first time
- typically NOT a botnet controller (although they could overlap)
- communications with these indicators may lead to a compromise and then to a possible botnet controller communication (if the infection was successful).
- typically used in preemptive blocking, alerts may not indicate infection was successful

Typical examples might include items from:

- http://www.malwaredomains.com

Phishing

The phishing assessment depicts:

- a luring attempt at a victim to exfiltrate some sort of credential
- a targeted attempt at getting someone to unintentionally cause infection (spear phishing)

Typical examples might include items from:

- http://www.phishtank.com

Fastflux

The fastflux assessment depicts:

- typically describing a botnet profile where fastflux activity is taking place

Scanner

The scanner assessment depicts:

- typically infrastructure being used to scan or brute-force (ssh, rdp, telnet, etc...)

Typical examples might include observations from:

- http://sshbl.org
- http://dragonresearchgroup.org/insight/sshpwauth.txt

Spam

The spam assessment depicts:

- typically infrastructure being used to facilitate the sending of spam

Searches

The search assessment depicts:

- identify's that someone searched for something of possible significance

Suspicious

The suspicious assessment depicts:

- Unknown assessment
- used as the "last default" assessment, combined with "description" for more accurate assessment (eg: assessment- suspicious, description- 'hijacked prefix', or assessment-suspicious, description- 'nameserver').

Whitelist

The Whitelist assessment depicts:

- denotes that specific entity (usually an address) should be considered harmless in nature
- denotes that blocking an entity would result in mass collateral damage (eg: yahoo virtually hosted servies)
- confidence should be applied to each entry to help calculate risk associated with whitelist

TestRouter

Router Testing

1. start the router in debug mode:

   ```
   $ sudo -u cif /opt/cif/bin/cif-router -d
   [2014-04-19T15:41:04,481Z][INFO]: frontend started on: tcp://*:4961
   [2014-04-19T15:41:04,486Z][INFO]: publisher started on: tcp://*:4963
   [2014-04-19T15:41:04,487Z][INFO]: router started...
   ^C
   ```

2. in a separate terminal, test connectivity to the router using the client `ping` flag:

   ```
   $ cif -p
   pinging: tcp://localhost:4961...
   roundtrip: 0.332042 ms
   roundtrip: 0.345236 ms
   roundtrip: 0.391154 ms
   roundtrip: 0.371904 ms
   done...
   ```

TestSmrt

Smrt Testing

1. perform a cif-smrt initial test run:

```
$ sudo -u cif cif-smrt --randomstart 0 --consolemode -d -r
/opt/cif/etc/rules/default
[2014-04-19T16:00:51,868Z][INFO]: cleaning up tmp...
[2014-04-19T16:00:52,012Z][INFO]: generating ping request...
[2014-04-19T16:00:52,077Z][INFO]: sending ping...
[2014-04-19T16:00:52,089Z][INFO]: ping returned
[2014-04-19T16:00:52,106Z][INFO]: processing: bin/cif-smrt -d -r
/opt/cif/etc/rules/default/bruteforceblocker.cfg -f ssh
[2014-04-19T16:00:52,427Z][INFO]: starting at: 2014-04-19T00:00:00Z
[2014-04-19T16:00:52,431Z][INFO]: processing...
[2014-04-19T16:00:54,532Z][INFO]: building events: 1273
[2014-04-19T16:00:55,335Z][INFO]: sending: 78
[2014-04-19T16:00:55,955Z][INFO]: took: ~0.921849
[2014-04-19T16:00:55,956Z][INFO]: rate: ~84.6125558524227 o/s
[2014-04-19T16:00:55,956Z][INFO]: processing: bin/cif-smrt -d -r
/opt/cif/etc/rules/default/drg.cfg -f ssh
...
```

The-CIF-Book

Chapter 1 - Introduction

1. What is the Collective Intelligence Framework (CIF)? (https://github.com/csirtgadgets/massive-octo-spice/wiki/Introduction)
2. The CIF community (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Community)
3. Who supports CIF? (https://github.com/csirtgadgets/massive-octo-spice/wiki/Who-supports-CIF%3F)
4. Frequently asked questions (https://github.com/csirtgadgets/massive-octo-spice/wiki/FAQ)

Chapter 2 - Getting Started

1. Building a CIF Server (https://github.com/csirtgadgets/massive-octo-spice/wiki/Building-a-CIF-Server)
2. Installing CIF (https://github.com/csirtgadgets/massive-octo-spice/wiki)
3. Exploring the file system (https://github.com/csirtgadgets/massive-octo-spice/wiki/Exploring-the-file-system)
4. Exploring the CIF binaries (https://github.com/csirtgadgets/massive-octo-spice/wiki/Exploring-the-CIF-binaries)
5. Exploring the network services (https://github.com/csirtgadgets/massive-octo-spice/wiki/Exploring-the-listening-network-services)
6. Exploring the software packages installed (https://github.com/csirtgadgets/massive-octo-spice/wiki/Exploring-the-software-packages-installed)
7. Exploring Tags (https://github.com/csirtgadgets/massive-octo-spice/wiki/Tags)
8. Exploring Confidence (https://github.com/csirtgadgets/massive-octo-spice/wiki/Confidence)
9. Exploring Timestamps (https://github.com/csirtgadgets/massive-octo-spice/wiki/Timestamp)
10. Exploring Whitelisting (https://github.com/csirtgadgets/massive-octo-spice/wiki/Whitelist)
11. Exploring Tokens (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-tokens)
12. Exploring Groups (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Groups)
13. Exploring the OSINT pre-configured (https://github.com/csirtgadgets/massive-octo-spice/wiki/Exploring-the-OSINT-pre-configured)

Chapter 3 - CIF clients

1. Introducing the CIF client (https://github.com/csirtgadgets/massive-octo-spice/wiki/Introducing-the-CIF-client)
2. Web Browser Plugins

Chapter 4 - Consuming Threat Intelligence

1. Creating a feed parsing configuration file (https://github.com/csirtgadgets/massive-octo-spice/wiki/ParsingFeeds)
2. Tutorial: Parsing a new threat intelligence feed (https://github.com/csirtgadgets/massive-octo-spice/wiki/Parsing-Feeds-Tutorial)
3. Advanced: Alternate methods for parsing data sets (https://github.com/csirtgadgets/massive-octo-spice/wiki/Alternate-methods-for-fetching-

and-parsing-data-sets)
4. FAQ: Parsing Feeds (https://github.com/csirtgadgets/massive-octo-spice/wiki/FAQ:-Parsing-Feeds)

Chapter 5 - Generating Threat Intelligence feeds

1. Introducing CIF feeds (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-Feeds)

Chapter 6 - CIF API

1. Exploring the API (https://github.com/csirtgadgets/massive-octo-spice/wiki/API)
2. CIF SDK (https://github.com/csirtgadgets/massive-octo-spice/wiki/CIF-SDK)
3. Bulk Submissions (https://github.com/csirtgadgets/massive-octo-spice/wiki/Bulk-Submissions)

Chapter 7 - CIF integration

1. Kibana (https://github.com/csirtgadgets/massive-octo-spice/wiki/KibanaGuide)
2. Snort (https://github.com/csirtgadgets/massive-octo-spice/wiki/Snort)
3. Bro (https://github.com/csirtgadgets/massive-octo-spice/wiki/Bro)
4. Bind (https://github.com/csirtgadgets/massive-octo-spice/wiki/Bind)
5. Tipping Point (https://github.com/csirtgadgets/massive-octo-spice/wiki/TippingPoint)
6. PassiveDNS (https://github.com/csirtgadgets/massive-octo-spice/wiki/PassiveDNS)
7. FireEye (https://github.com/csirtgadgets/massive-octo-spice/wiki/FireEye)

Chapter 8 - CIF Cookbooks

1. Sharing Threat Intelligence (https://github.com/csirtgadgets/massive-octo-spice/wiki/Sharing-Threat-Intelligence)
2. Contributing Threat Intelligence (https://github.com/csirtgadgets/massive-octo-spice/wiki/Contributing-Threat-Intelligence)

Chapter 9 - Administration

1. Troubleshooting (https://github.com/csirtgadgets/massive-octo-spice/wiki/Troubleshooting-cif)
2. CIFv1 to CIFv2 Migration (Migration)
3. Backup and Restore (https://github.com/csirtgadgets/massive-octo-spice/wiki/Backup-and-Restore)
4. Distributed Environment
5. Monit (https://github.com/csirtgadgets/massive-octo-spice/wiki/Monit)
6. Pruning the ElasticSearch database (https://github.com/csirtgadgets/massive-octo-spice/wiki/Pruning-the-ElasticSearch-database)

Chapter 10 - Development

1. Introduction
2. Using GitHub for CIF development (Development_GitHub)
3. CIF Architecture Overview (CIF-Architecture-Overview)

Troubleshooting-CIF

Troubleshooting CIF

Basic troubleshooting steps

What can I do to if my CIF server isn't working as I expect?

1. Reboot the CIF server
2. Run the cif ping command with debug

```
$ cif -p -d
[2016-01-13T03:32:38,391Z][INFO][main:261]: starting up client...
[2016-01-13T03:32:38,392Z][INFO][main:272]: pinging: https://localhost...
[2016-01-13T03:32:38,392Z][DEBUG][CIF::SDK::Client:203]: generating ping...
[2016-01-13T03:32:38,392Z][DEBUG][CIF::SDK::Client:165]: uri created:
https://localhost/ping?
[2016-01-13T03:32:38,392Z][DEBUG][CIF::SDK::Client:166]: making request...
[2016-01-13T03:32:38,877Z][INFO][CIF::SDK::Client:170]: status: 200
[2016-01-13T03:32:38,877Z][DEBUG][CIF::SDK::Client:173]: decoding content..
roundtrip: 0.485375 ms
...
[2016-01-13T03:32:44,223Z][INFO][main:393]: done...
```

1. Make a cif query with debug

```
$ cif -q example.com -d
[2016-01-13T02:58:21,076Z][INFO][main:261]: starting up client...
[2016-01-13T02:58:21,076Z][INFO][main:296]: running search...
[2016-01-13T02:58:21,076Z][DEBUG][CIF::SDK::Client:165]: uri created:
https://localhost/observables?observable=example.com
[2016-01-13T02:58:21,076Z][DEBUG][CIF::SDK::Client:166]: making request...
[2016-01-13T02:58:21,745Z][INFO][CIF::SDK::Client:170]: status: 200
[2016-01-13T02:58:21,745Z][DEBUG][CIF::SDK::Client:173]: decoding content..
[2016-01-13T02:58:21,745Z][INFO][main:356]: search returned, formatting..
tlp  |group   |reporttime          |observable |cc|asn|confidence|tags
|description|rdata|provider      |altid_tlp|altid
amber|everyone|2015-12-21T20:01:16Z|example.com|  |   |25        |search|
|     |root@localhost|         |
amber|everyone|2015-12-21T20:01:18Z|example.com|  |   |25        |search|
|     |root@localhost|         |
...

[2016-01-13T02:58:21,757Z][INFO][main:393]: done...
```

1. Read through all the CIF logs:

```
$ tail /var/log/cif-router.log
[2016-01-13T03:00:48,136Z][12139][INFO]: staring up..
[2016-01-13T03:00:48,258Z][12141][INFO]: started, waiting for messages..

$ tail /var/log/cif-smrt.log
[2016-01-13T03:00:52,979Z][12325][INFO]: staring up...
[2016-01-13T03:00:52,996Z][12329][INFO]: delaying start for: 4min then running
every 60min there after...
[2016-01-13T03:00:52,997Z][12329][INFO]: to run immediately, set: --randomstart
0 or --testmode
[2016-01-13T03:00:52,997Z][12329][INFO]: to see the list of options, use -h

$ tail /var/log/cif-starman.log
[2016-01-13T03:00:52,233Z][12295][INFO]: starting CIF::REST
[2016-01-13T03:00:52,238Z][12297][INFO]: starting CIF::REST
[2016-01-13T03:00:52,255Z][12299][INFO]: starting CIF::REST

$ tail /var/log/cif-worker.log
[2016-01-13T03:00:50,256Z][12188][INFO]: sending ping...
[2016-01-13T03:00:50,313Z][12195][INFO]: staring worker..
[2016-01-13T03:00:50,315Z][12196][INFO]: staring worker..
...
[2016-01-13T03:00:50,337Z][12192][INFO]: starting...
```

1. Verify apache is working

```
$ curl -ik https://localhost/
HTTP/1.1 200 OK
Date: Wed, 13 Jan 2016 13:05:53 GMT
Server: Apache
Vary: Accept-Encoding
Content-Length: 671
Content-Type: text/html;charset=UTF-8

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
...
```

1. Read through the Apache logs

```
$ sudo tail /var/log/apache2/error.log
$ sudo tail /var/log/apache2/ssl_access.log
```

1. Verify ElasticSearch is working

```
$ curl -i 'http://localhost:9200/_cluster/health?pretty'
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 283

{
"cluster_name" : "elasticsearch",
"status" : "yellow",
"timed_out" : false,
"number_of_nodes" : 1,
"number_of_data_nodes" : 1,
"active_primary_shards" : 155,
"active_shards" : 155,
"relocating_shards" : 0,
"initializing_shards" : 0,
"unassigned_shards" : 155
}
```

1. Read through the ElasticSearch logs

```
$ tail /var/log/elasticsearch/elasticsearch.log
```

1. restart cif to look for errors

```
$ sudo service cif-services restart
* Stopping cif-router
[ OK ]
* Starting cif-router
[ OK ]
* Stopping cif-worker
[ OK ]
* Starting cif-worker
[ OK ]
* Stopping cif-starman
[ OK ]
* Starting cif-starman
[ OK ]
* Stopping cif-smrt
[ OK ]
* Starting cif-smrt
[ OK ]
```

1. Verify the host has enough free disk space

```
$ df -h
Filesystem                        Size  Used Avail Use% Mounted on
udev                              7.9G  4.0K  7.9G   1% /dev
tmpfs                             1.6G  444K  1.6G   1% /run
/dev/mapper/cifv2--rc6--vg-root  230G   14G  205G   7% /
none                              4.0K     0  4.0K   0% /sys/fs/cgroup
none                              5.0M     0  5.0M   0% /run/lock
none                              7.9G     0  7.9G   0% /run/shm
none                              100M     0  100M   0% /run/user
/dev/sda1                         236M   69M  155M  31% /boot
```

1. Verify the host has enough free memory

```
$ free -m
             total       used       free     shared    buffers     cached
Mem:         16047      12496       3551          0        158       2947
-/+ buffers/cache:       9389       6657
Swap:        16383         62      16321
```

Enable debug logging across all CIF services

1. Add '-d' to CIF_DEBUGGING in /etc/default/cif

```
$ cat /etc/default/cif
# Directory where the binary distribution resides
CIF_HOME=/opt/cif

PATH=$CIF_HOME/bin:$PATH

if [ -d /opt/cif/lib/perl5 ]; then
  export PERL5LIB=/opt/cif/lib/perl5
fi

# Run as this user ID and group ID
CIF_USER=cif
CIF_GROUP=cif

# data directory
DATA_DIR=/var
LOG_DIR=/var/log

# configuration directory
CONF_DIR=/etc/cif

# add -d to turn on debugging
CIF_DEBUGGING="-d"
```

1. Restart all CIF services

```
$ sudo service cif-services restart
* Stopping cif-router
[ OK ]
* Starting cif-router
[ OK ]
* Stopping cif-worker
[ OK ]
* Starting cif-worker
[ OK ]
* Stopping cif-starman
[ OK ]
* Starting cif-starman
[ OK ]
* Stopping cif-smrt
[ OK ]
* Starting cif-smrt
[ OK ]
```

1. Make a cif query with debug

```
$ cif -q example.com -d
[2016-01-13T02:58:21,076Z][INFO][main:261]: starting up client...
[2016-01-13T02:58:21,076Z][INFO][main:296]: running search...
[2016-01-13T02:58:21,076Z][DEBUG][CIF::SDK::Client:165]: uri created:
https://localhost/observables?observable=example.com
[2016-01-13T02:58:21,076Z][DEBUG][CIF::SDK::Client:166]: making request...
[2016-01-13T02:58:21,745Z][INFO][CIF::SDK::Client:170]: status: 200
[2016-01-13T02:58:21,745Z][DEBUG][CIF::SDK::Client:173]: decoding content..
[2016-01-13T02:58:21,745Z][INFO][main:356]: search returned, formatting..
tlp  |group   |reporttime          |observable |cc|asn|confidence|tags
|description|rdata|provider      |altid_tlp|altid
amber|everyone|2015-12-21T20:01:16Z|example.com|  |   |25        |search|
|     |root@localhost|         |
amber|everyone|2015-12-21T20:01:18Z|example.com|  |   |25        |search|
|     |root@localhost|         |
```

1. Read through the apache logs

```
$ sudo tail /var/log/apache2/ssl_access.log
::1 - - [13/Jan/2016:03:38:23 -1000] "GET /observables?observable=example.com
HTTP/1.1" 200 5685 "-" "cif-sdk-perl/2.00_30"

$ sudo tail /var/log/apache2/ssl_access.log
```

1. Read through the CIF logs

```
$ tail /var/log/cif-router.log
$ tail /var/log/cif-smrt.log
$ tail /var/log/cif-starman.log
$ tail /var/log/cif-worker.log
```

1. Once done troubleshooting, be sure to turn off CIF debugging and restart all the CIF services; the logging is verbose and will use up a lot of disk space.

Tutorial:-CIF-development-using-vagrant

This tutorial will show you how to use OS X + Vagrant + Virtualbox + Git to easily spin up and tear down CIFv2 virtual machines for development.

1. Ensure you have all the Vagrant prerequisites installed and configured (https://github.com/csirtgadgets/massive-octo-spice/wiki/Vagrant-prerequisites).

2. Clone the CIFv2 report to your OS X machine

3. Open the Terminal

4. (optional) Create a folder named Development

   ```
   mkdir Development && cd Development
   ```

5. Clone CIFv2 (a fork??) using Git and cd into it's directory

   ```
   git clone https://github.com/csirtgadgets/massive-octo-spice.git && cd
   massive-octo-spice/
   ```

6. Start a Vagrant virtual machine

   ```
   vagrant up
   ```

7. SSH into the Vagrant virtual machine

   ```
   vagrant ssh
   ```

8. Within the virtual machine cd into /vagrant. This is a shared folder to your host machine (e.g. `~/Development/massive-octo-spice`)

   ```
   cd /vagrant
   ```

9. Install CIFv2

   ```
   sudo apt-get update && sudo apt-get upgrade -y && sudo apt-get install -y
   htop build-essential automake autoconf git
   bash autogen.sh
   sudo bash ./hacking/platforms/easybutton.sh
   sudo chown `whoami`:`whoami` ~/.cif.yml
   ```

Once you are finished with your development, you can stop, delete etc using the these common Vagrant commands (https://github.com/csirtgadgets/massive-octo-spice/wiki/Vagrant-prerequisites#common-vagrant-commands).

Vagrant-prerequisites

Why Vagrant?

- Vagrant (https://docs.vagrantup.com/v2/) allows you to programmatically spin up a Virtual Machine from a template VM (think .iso) to a configured VM (think post OS installer). Post install being all the things you would have to configure during a distribution setup (networking, users, etc)
- You can easily share these Vagrant configurations amongst your team
- You can configure Vagrant to bring up more than one machine in a single "vagrant up" command
- You can script post install actions (e.g. apt-get install apache2)

Common Vagrant commands

- vagrant up - starts and provisions the vagrant environment
- vagrant halt - stops the vagrant machine
- vagrant status - outputs status of the vagrant machine
- vagrant global-status - outputs status Vagrant environments for this user
- vagrant destroy - stops and deletes all traces of the vagrant machine
- vagrant resume - resume a suspended vagrant machine
- vagrant suspend - suspends the machine
- vagrant -h - help

Prerequisites for OS X

1. Install Virtualbox (https://www.virtualbox.org/wiki/Downloads)
2. Install Vagrant (https://docs.vagrantup.com/v2/installation/)

If you do not have Apple Xcode installed:

1. Install Git (http://git-scm.com/download/mac)

2. Place Git in the PATH of your shell

3. Create/edit your bash_profile

   ```
   vim ~/.bash_profile
   ```

4. Add the following:

   ```
   PATH=/usr/local/git/bin:$PATH
   export PATH
   ```

5. Reload your bash_profile

   ```
   source ~/.bash_profile
   ```

What-is-the-Collective-Intelligence-Framework?

What is the Collective Intelligence Framework?

CIF is a cyber threat intelligence management system. CIF allows you to combine known malicious threat information from many sources and use that information for identification (incident response), detection (IDS) and mitigation (null route). The most common types of threat intelligence warehoused in CIF are IP addresses, FQDNs and URLs that are observed to be related to malicious activity.

This framework pulls in various data-observations from any source; create a series of messages "over time" (eg: reputation). When you query for the data, you'll get back a series of messages chronologically and make decisions much as you would look at an email thread, a series of observations about a particular bad-actor.

CIF helps you to parse, normalize, store, post process, query, share and produce data sets of threat intelligence.

the Process

Parse

CIF supports ingesting many different sources of data of the same type; for example data sets or “feeds” of malicious domains. Each similar dataset can be marked with different attributes like source and confidence to name a few.

Normalize

Threat intelligence datasets often have subtle differences between them. CIF normalizes these data sets which gives you a predictable experience when leveraging the threat intelligence in other applications or processes.

Post Process

CIF has many post processors that derive additional intelligence from a single piece of threat intelligence. A simple example would be that a domain and an IP address can be derived from a URL ingested into CIF.

Store

CIF uses JSON and ElasticSearch as it's data store to warehouse billions of records of threat intelligence

Query

CIF can be queried via a web browser, native CLI client or directly using the API.

Share

CIF supports users, groups and api keys. Each threat intelligence record can be tagged to be shared with specific group of users. This allows the sharing of threat intelligence among federations.

Produce

CIF supports creating new data sets from the stored threat intelligence. These data sets can be created by type and confidence. CIF also supports whitelisting during the feed generation process.

where-do-i-start

Overview

These integrations assume you have the python SDK (https://github.com/csirtgadgets/cif-sdk-py) or perl SDK (https://github.com/csirtgadgets/p5-cif-sdk) or successfully installed and a valid ~/.cif.yml config. Installing the python client is as easy as:

```
$ sudo pip install 'cifsdk>=2.0,<3.0'
```

While CSIRT Gadgets **DOES NOT ENDORSE ANY of these projects or services**, we do our best to help bootstrap community integration. Please feel free to contribute integrations to the wiki!

Chrome Plugin

TODO

https://github.com/csirtgadgets/cif-chrome

Basic Output Formats

Table

```
$ cif --otype ipv4 --limit 5 --format table
+-------+----------+----------------------+----------------------+-------------
--+-------+----+-------+----------------------------------+-----------+-------
------+----------------+-----------------------------+---------------+
|  tlp  |  group   |       lasttime       |      reporttime      |   observable
| otype | cc |  asn  |             asn_desc             | confidence |
description |      tags      |            rdata            | provider      |
+-------+----------+----------------------+----------------------+-------------
--+-------+----+-------+----------------------------------+-----------+-------
------+----------------+-----------------------------+---------------+
| amber | everyone | 2016-02-23T14:58:21Z | 2016-02-23T14:58:21Z |
107.180.51.16 |  ipv4 | US | 26496 | AS-26496-GO-DADDY-COM-LLC GoDa.. |
13.996   |             | phishing,rdata |       lasttimeserc.com      |
openphish.com |
| amber | everyone | 2016-02-23T14:58:21Z | 2016-02-23T14:58:21Z |
216.69.185.19 |  ipv4 | US | 26496 | AS-26496-GO-DADDY-COM-LLC GoDa.. |
13.996   |             | phishing,rdata |    ns37.domaincontrol.com   |
openphish.com |
| amber | everyone | 2016-02-23T14:58:22Z | 2016-02-23T14:58:22Z |
107.180.51.16 |  ipv4 | US | 26496 | AS-26496-GO-DADDY-COM-LLC GoDa.. |
13.996   |             | phishing,rdata |       lasttimeserc.com      |
openphish.com |
| amber | everyone | 2016-02-23T14:58:22Z | 2016-02-23T14:58:22Z |
188.121.58.1 |  ipv4 | NL | 26496 | AS-26496-GO-DADDY-COM-LLC GoDa.. |   13.996
|             | phishing,rdata | inetsoftwaresolutions.co.uk | openphish.com |
| amber | everyone | 2016-02-23T14:58:22Z | 2016-02-23T14:58:22Z |
216.69.185.19 |  ipv4 | US | 26496 | AS-26496-GO-DADDY-COM-LLC GoDa.. |
20.023   |             | phishing,rdata |    ns37.domaincontrol.com   |
spamhaus.org  |
+-------+----------+----------------------+----------------------+-------------
--+-------+----+-------+----------------------------------+-----------+-------
------+----------------+-----------------------------+---------------+
```

CSV

Most Fields

```
$ cif --otype ipv4 --limit 5 --format csv
amber,everyone,2016-02-23T14:58:21Z,2016-02-
23T14:58:21Z,107.180.51.16,ipv4,US,26496,AS-26496-GO-DADDY-COM-LLC
GoDa..,13.996,,"phishing,rdata",lasttimeserc.com,openphish.com
amber,everyone,2016-02-23T14:58:22Z,2016-02-
23T14:58:22Z,107.180.51.16,ipv4,US,26496,AS-26496-GO-DADDY-COM-LLC
GoDa..,13.996,,"phishing,rdata",lasttimeserc.com,openphish.com
```

Custom Fields

```
$ cif --otype ipv4 --limit 5 --format csv --fields
tlp,group,reporttime,observable
amber,everyone,2016-02-23T14:58:21Z,107.180.51.16
amber,everyone,2016-02-23T14:58:22Z,107.180.51.16
```

JSON

```
$ cif --otype ipv4 --limit 5 --format json
[{"geolocation": "33.6119,-111.8906", "protocol": 6, "cc": "US", "rir": "arin",
"related": "e7ab7044e21120408423e3aef2e7c09842e53d004e48e053c0bc16fe5383b429",
"prefix": "107.180.51.0/24", "timezone": "America/Phoenix", ... }]
```

STIX

```
$ cif --otype ipv4 --limit 5 --format stix
<stix:STIX_Package
    xmlns:AddressObj="http://cybox.mitre.org/objects#AddressObject-2"
    xmlns:cybox="http://cybox.mitre.org/cybox-2"
    xmlns:cyboxCommon="http://cybox.mitre.org/common-2"
    xmlns:cyboxVocabs="http://cybox.mitre.org/default_vocabularies-2"
    xmlns:example="http://example.com"
    xmlns:indicator="http://stix.mitre.org/Indicator-2"
    xmlns:stix="http://stix.mitre.org/stix-1"
    xmlns:stixCommon="http://stix.mitre.org/common-1"
    xmlns:stixVocabs="http://stix.mitre.org/default_vocabularies-1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" id="example:Package-
38984c41-fa98-457a-befe-e97e65c94795" version="1.2">
    <stix:STIX_Header/>
    <stix:Indicators>
        <stix:Indicator id="example:indicator-6bed9b83-0879-4d48-8dd9-
95f93fd2acbe" timestamp="2016-02-23T14:58:21+00:00"
xsi:type='indicator:IndicatorType'>
            <indicator:Description>phishing,rdata</indicator:Description>
            <indicator:Observable id="example:Observable-780dacce-5338-4cee-
b7e1-af2bda9d5502">
                <cybox:Object id="example:Address-a95f9a3a-de3c-49aa-b30c-
331137031105">
                    <cybox:Properties xsi:type="AddressObj:AddressObjectType"
category="ipv4-addr">

<AddressObj:Address_Value>107.180.51.16</AddressObj:Address_Value>
...
```

Open Source Integrations

Bro

While focusing on network security monitoring, Bro provides a comprehensive platform for more general network traffic analysis as well. Well grounded in more than 15 years of research, Bro has successfully bridged the traditional gap between academia and operations since its inception. Today, it is relied upon operationally in particular by many scientific environments for securing their cyberinfrastructure. Bro's user community includes major universities, research labs, supercomputing centers, and open-science communities.

see more at bro.org (http://bro.org)

```
$ cif --otype ipv4 --feed --confidence 85 --format bro --limit 5
#fields indicator   indicator_type  meta.desc   meta.cif_confidence meta.source
92.50.31.66 Intel::ADDR exploit 95  spamhaus.org
210.4.72.138    Intel::ADDR exploit 95  spamhaus.org
61.150.89.67    Intel::ADDR spam    95  spamhaus.org
68.180.32.194   Intel::ADDR exploit 95  spamhaus.org
221.206.72.203  Intel::ADDR spam    95  spamhaus.org
```

Snort

Snort is an open source network intrusion prevention system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching, and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more.

see more at snort.org (http://snort.org)

```
$ cif --otype ipv4 --feed --confidence 85 --format snort --limit 5
alert TCP any any -> 74.28.188.130 any (reference:
http://www.spamhaus.org/query/bl?ip=74.28.188.130; priority: 1; threshold: type
limit,track by_src,count 1,seconds 3600; sid: 5000000000; msg: CIF - GREEN -
exploit;)
alert IP any any -> 74.208.184.119 any (reference:
http://www.spamhaus.org/query/bl?ip=74.208.184.119; priority: 1; threshold:
type limit,track by_src,count 1,seconds 3600; sid: 5000000001; msg: CIF - GREEN
- spam;)
alert TCP any any -> 173.237.190.72 any (reference:
http://www.spamhaus.org/query/bl?ip=173.237.190.72; priority: 1; threshold:
type limit,track by_src,count 1,seconds 3600; sid: 5000000002; msg: CIF - GREEN
- spam;)
```

Bind (Bind)

BIND is open source software that implements the Domain Name System (DNS) protocols for the Internet. It is a reference implementation of those protocols, but it is also production-grade software, suitable for use in high-volume and high-reliability applications. The name BIND stands for “Berkeley Internet Name Domain”, because the software originated in the early 1980s at the University of California at Berkeley.

BIND is by far the most widely used DNS software on the Internet, providing a robust and stable platform on top of which organizations can build distributed computing systems with the knowledge that those systems are fully compliant with published DNS standards.

see more at isc.org (http://www.isc.org/downloads/bind)

```
$ cif --otype fqdn --feed --confidence 85 --format bind --limit 5
// generated by: CIF at 2016-35-23T10:02:55 EST
zone "mail.ghiend.com" {type master; file "/etc/namedb";};
zone "ghiend.com" {type master; file "/etc/namedb";};
zone "ns1.bwreg.com" {type master; file "/etc/namedb";};
```

JusinAzoff - Ninfo

NEEDS TO BE UPDATED FOR V2

QUERY ALL-THE-THINGS!!!!
nInfo is a library, CLI tool, and web interface (and lots of plugins) for gathering information on any of the following:

- IP Address (v4 or v6)
- CIDR Block (v4 or v6)
- MAC Address
- Hostname
- Username
- Hashes (as in md5/sha1 etc)

It consists of multiple plugin classes that implement a get_info function. The classes contain metadata for the type of arguments they accept, and if they are relevant for internal and or external hosts.

see more at github.com/JustinAzoff/ninfo (https://github.com/JustinAzoff/ninfo)

for the CIF plugin, see: https://github.com/JustinAzoff/ninfo-plugin-cif

Kibana (KibanaGuide)

Kibana is an open source (Apache Licensed), browser based analytics and search interface to Logstash and other timestamped data sets stored in ElasticSearch. With those in place Kibana is a snap to setup and start using (seriously). Kibana strives to be easy to get started with, while also being flexible and powerful

Commercial Integrations

PaloAlto

Building on the DShield model (https://isc.sans.edu/forums/diary/Subscribing+to+the+DShield+Top+20+on+a+Palo+Alto+Networ
leverage CIF to generate a text file that can be imported into the dynamic block list of your device:

```
$ cif --otype ipv4 --feed --confidence 85 --format csv --fields observable --limit 5
92.50.31.66
210.4.72.138
61.150.89.67
68.180.32.194
221.206.72.203
```

see more at Paloalto Networks (https://www.paloaltonetworks.com/products/secure-the-network/next-generation-firewall)

where-do-i-start-feeds

Overview

These integrations assume you have the python SDK (https://github.com/csirtgadgets/cif-sdk-py) or perl SDK (https://github.com/csirtgadgets/p5-cif-sdk) or successfully installed and a valid ~/.cif.yml config. Installing the python client is as easy as:

```
$ sudo pip 'cifsdk>=2.0,<3.0'
```

Starter Feeds

If you're not familiar with the [output] Feeds concept with CIF, checkout the CIF book (CIF-Feeds). The most common feed combinations are:

IPV4

```
$ cif --feed --otype ipv4 --confidence 85 --tags scanner
$ cif --feed --otype ipv4 --confidence 85 --tags hijacked
$ cif --feed --otype ipv4 --confidence 85 --tags botnet
$ cif --feed --otype ipv4 --confidence 85 --tags malware
$ cif --feed --otype ipv4 --confidence 85 --tags spam
```

FQDN

```
$ cif --feed --otype fqdn --confidence 85 --tags botnet
$ cif --feed --otype fqdn --confidence 85 --tags malware
$ cif --feed --otype fqdn --confidence 85 --tags phishing

$ cif --feed --otype fqdn --confidence 65 --tags malware
```

URL

```
$ cif --feed --otype url --confidence 85 --tags phishing
$ cif --feed --otype url --confidence 85 --tags malware
$ cif --feed --otype url --confidence 85 --tags botnet
```