Velociraptor

Hunting Evil with open source!

[<Mike Cohen> mike@velocidex.com](mailto:mike@velocidex.com)

AGPL v3 Free Software

Free as in Freedom

What we will do today

- Install a secure deployment of Velociraptor in 15 minutes.
  - We could have several thousand end points on the same deployment!
- Interactively view files/registry from an endpoint.
  - Use Fuse to run third party tools on remote endpoints.
- Collect artifacts from endpoints
  - Run hunts to collect artifacts from end points in seconds. Execution artifacts like amcache, chrome extensions, installed programs, evidence of sysinternal tool execution.
  - Write custom artifact to collect Image File Execution Options backdoors, acquire process memory dump for processes that match a yara sig.
- Collect events from endpoints in real time:
  - Process execution, service installations, dns lookups
  - Write our own artifact: Watch for usb drive insertion then list all doc files added to it. Search for classification markings.
  - Watch user’s temp dir and when a new doc file is added check it for macros.

What we will do today

- Collect and preserve evidence in DFIR case
  - Browser cache, registry hives, event logs
  - Do this locally or remotely.

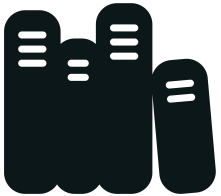
All the above can be done:

- Locally, interactively
- One endpoint remotely at a time
- On 5,000 (or more) endpoints at once!!!

Deploy

Access

Collect



Monitor

Required downloads - preinstall needed software

- Get the latest Velociraptor windows binary from GitHub
- Notepad++ - a better notepad
- Winfsp is a windows version of Fuse.
- Chrome is a better web browser.
- Libreoffice or Excel are excellent spreadsheets
- Sysinternal tools:
  - Psexec
  - Autoruns
  - Or just grab everything
- These slides!

What is Velociraptor?

A FOSS project heavily influenced by

- Google's GRR
- Facebook's OSQuery
- Google's Rekall

Both a triaging tool and an endpoint monitoring and collection tool

Implements a powerful Velociraptor Query Language (VQL) engine.

https://docs.velociraptor.velocidex.com/

https://github.com/Velocidex/velociraptor

AGPL 3 Free Software

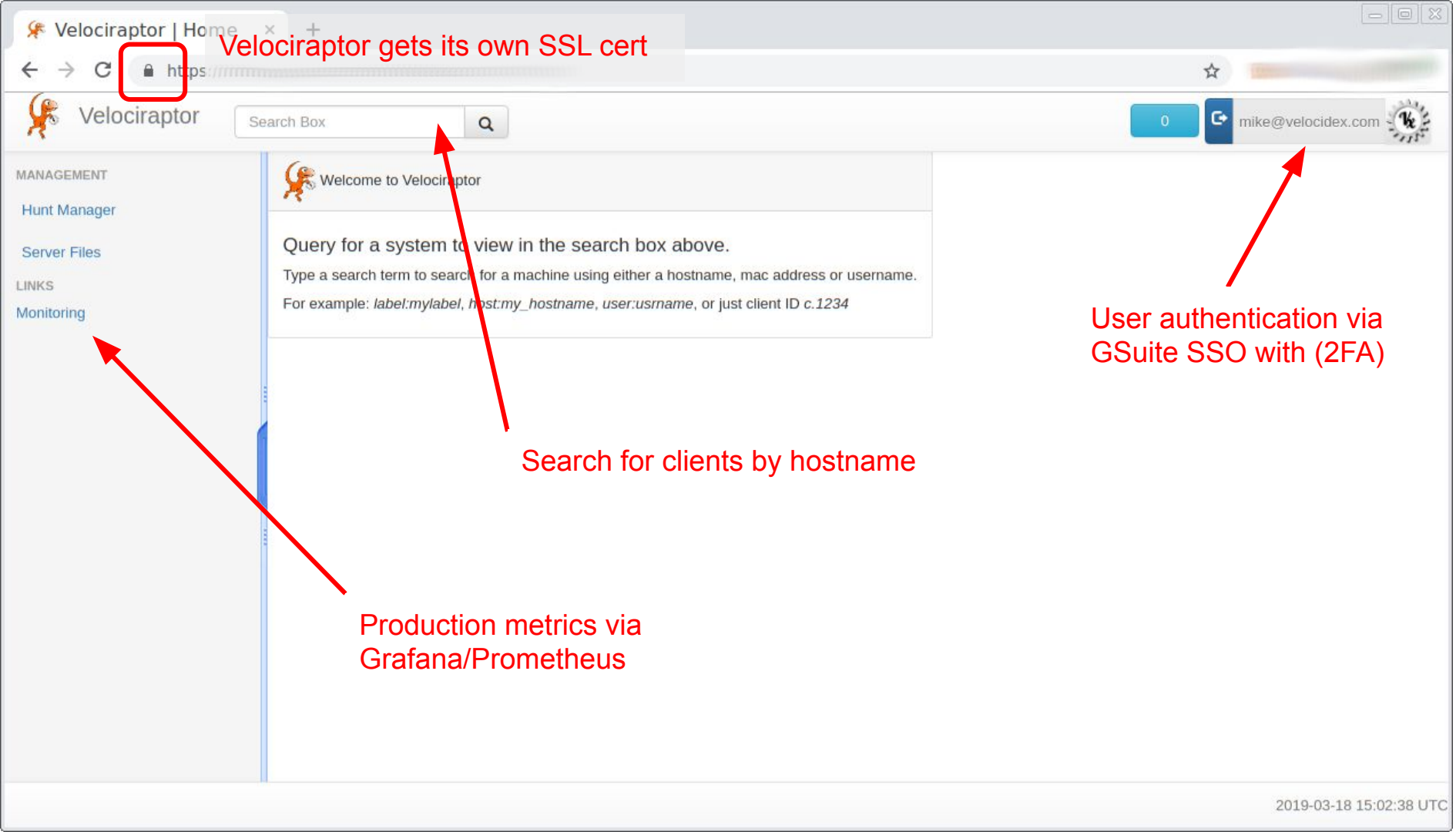
Module 1

Rapid Response

Interactively investigate a single endpoint

Velociraptor Demo

- A quick 15 minute demo of the GUI to show some high level capabilities.
- I will be using a cloud deployment with some test machines
  - There are not many machines on this deployment but hopefully you will get a taste as to what it looks like!
- Don't worry - you will get to install this on your own machine shortly!
  - Try to think of use cases in your own daily work
  - I will present some test cases of how we use it.


Velociraptor | Home
https://
Velociraptor gets its own SSL cert
Velociraptor
Search Box
0
mike@velocidex.com
MANAGEMENT
Hunt Manager
Server Files
LINKS
Monitoring
Welcome to Velociraptor
Query for a system to view in the search box above.
Type a search term to search for a machine using either a hostname, mac address or username.
For example: label:mylabel, host:my_hostname, user:usrname, or just client ID c.1234
User authentication via GSuite SSO with (2FA)
Search for clients by hostname
Production metrics via Grafana/Prometheus
2019-03-18 15:02:38 UTC

Last IP address

Search for hostname: autocomplete,wildcards

Last active < 1 min ago

Labels: grouping hosts

Velociraptor

*computer

TestComputer
Access reason: test
Status: 4 seconds ago
106.71.187.90:51288

Host Information

Start new flows

Collect Artifacts

Browse Virtual Filesystem

Manage launched flows

MANAGEMENT

Hunt Manager

Server Files

LINKS

Monitoring

	Online	ClientID	Host	OS Version	Labels
		C.2ee4642b0f48b849	TestComputer	Microsoft Windows 10 Pro N10.0.17134 Build 17134	AlphaGroup test
		C.5e388e48f20c160a	TestComputer	Microsoft Windows 10 Pro N10.0.17134 Build 17134	
		C.5e4b36965eaab7f8	TestComputer	Microsoft Windows 10 Pro N10.0.17134 Build 17134	
		C.7bcd6d80100bf5f6	TestComputer	Microsoft Windows 10 Pro N10.0.17134 Build 17134	
		C.b71062cbeab5e1a4	TestComputer	Microsoft Windows 10 Pro N10.0.17134 Build 17134	
		C.da531c39568ff1f0	TestComputer	Microsoft Windows 10 Pro N10.0.17134 Build 17134	

Unique Client ID

Client Version

Velociraptor

*computer

0

mike@velocidex.com

TestComputer
Access reason: test
Status: 1 minutes ago
106.71.187.90:51294

Host Information

Start new flows

Collect Artifacts

Browse Virtual Filesystem

Manage launched flows

MANAGEMENT

Hunt Manager

Server Files

LINKS

Monitoring

TestComputer C.2ee4642b0f48b849

Interrogate

Overview | VQL Drilldown

client_id	C.2ee4642b0f48b849	
agent_information	version	2019-02-21T21:34:46+10:00
	name	velociraptor
os_info	system	windows
	release	Microsoft Windows 10 Pro N10.0.17134 Build 17134
	machine	amd64
	fqdn	TestComputer
last_seen_at	2019-03-18 15:23:26 UTC	
last_ip	106.71.187.90:51294	
last_ip_class	EXTERNAL	
labels	AlphaGroup	
	test	

The Virtual File System (VFS)

Velociraptor

Search Box

Client Filesystem

0

mike@velocidex.com

TestComputer
Access reason: test
Status: 46 seconds ago
106.71.187.90:49716

Host Information

Start new flows

Collect Artifacts

Browse Virtual Filesystem

Manage launched flows

MANAGEMENT

Hunt Manager

Server Files

LINKS

Monitoring

- file
  - C:
  - D:
  - Z:
- ntfs
  - \\.\C:
  - \\?\GLOBALROOT\Device\Ha
- registry
- artifacts
- monitoring

> ntfs

ntfs @ 2019-03-18 15:58:09 UTC

Download	Name	Size	Mode	mtime	atime	ctime
	\\.\C:	0	d---------			
	\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1	0	d---------			

> ntfs > \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1

Stats | Download | TextView | HexView | CSVView

Attribute	Value
Mode	d---------
Name	\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
Size	0
_Data	{"DeviceObject":"\\\\?\\GLOBALROOT\\Device\\HarddiskVolumeShadowCopy1","ID":"{C11B8E74-F9FB-42 F1262E1A07CE}","InstallDate":"20190318032327.625965-420","OriginatingMachine":"TestComputer","VolumeName":"\\\\?\\Volume{3dc4b590-0000-0000-0000-501f00000000}\\"}
_FullPath	\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1

Access to VSC

Refresh directory

2019-03-18 23:18:37 UTC

File contents are available

Velociraptor

*computer

0

mike@velocidex.com

TestComputer
Access reason: test
Status: 43 seconds ago
106.71.187.90:51390

Host Information

Start new flows

Collect Artifacts

Browse Virtual Filesystem

Manage launched flows

MANAGEMENT

Hunt Manager

Server Files

LINKS

Monitoring

- file
  - C:
    - $Recycle.Bin
    - Documents a
    - Go
    - MinGW
    - PerfLogs
    - Program File
    - Program File
    - ProgramData
    - Python27
      - DLLs
      - Doc
      - Lib
      - Scripts
      - Tools
      - include
      - libs
      - tcl
    - Recovery
    - System Volu
    - Users

> file > C: > Python27

/file/C:/Python27 @ 2019-03-18 16:26:08 UTC

Download	Name	Size	Mode	mtime	atime	ctime
	DLLs	0	drwxrwxrwx	2017-11-01T11:42:05-07:00	2019-01-14T14:54:55-08:00	2017-11-01T11:39:49-07:00
	Doc	0	drwxrwxrwx	2017-11-01T11:42:08-07:00	2019-01-14T14:54:55-08:00	2017-11-01T11:42:08-07:00
	LICENSE.txt	38580	-rw-rw-rw-	2017-09-16T20:23:38-07:00	2019-01-13T00:16:13-08:00	2017-09-16T20:23:38-07:00
	Lib	0	drwxrwxrwx	2017-11-01T16:38:53-07:00	2019-03-12T03:02:24-07:00	2017-11-01T11:39:44-07:00
	NEWS.txt	486284	-rw-rw-rw-	2017-09-16T19:57:38-07:00	2019-01-12T23:21:29-08:00	2017-09-16T19:57:38-07:00

> file > C: > Python27 > LICENSE.txt

Stats | Download | TextView | HexView | CSVView

```
A. HISTORY OF THE SOFTWARE
==========================

Python was created in the early 1990s by Guido van Rossum at Stichting
Mathematisch Centrum (CWI, see http://www.cwi.nl) in the Netherlands
as a successor of a language called ABC.  Guido remains Python's
principal author, although it includes many contributions from others.

In 1995, Guido continued his work on Python at the Corporation for
National Research Initiatives (CNRI, see http://www.cnri.reston.va.us)
in Reston, Virginia where he released several versions of the
```

2019-03-18 16:27:29 UTC

Server Monitoring

Making sure everything is working well!

Monitoring server health

Using **top** for basic overview. Is the system melting down? (Idle system with ~2k endpoints)

```
top - 15:29:46 up 12 days, 15:30,  7 users,  load average: 0.00, 0.04, 0.05
Tasks: 120 total,   1 running, 119 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.7 us,  0.3 sy,  0.0 ni, 97.8 id,  0.0 wa,  0.0 hi,  0.1 si,  0.2 st
KiB Mem : 16265876 total,   167180 free,   376776 used, 15721920 buff/cache
KiB Swap:        0 total,        0 free,        0 used. 15477612 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 7720 root      20   0  910400 118676  13800 S   9.6  0.7  25:20.82 ./velociraptor --config server.config.yaml frontend -v
10849 centos    20   0 1133316  44268  15384 S   0.3  0.3   5:55.48 ./bin/grafana-server -config velo.ini
31553 centos    20   0  148892  22624   1012 S   0.3  0.1   2:48.03 SCREEN -T screen-256color -S byobu -c /usr/share/byobu/pr
    1 root      20   0   43508   3696   2416 S   0.0  0.0   0:07.70 /usr/lib/systemd/systemd --system --deserialize 22
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.14 [kthreadd]
    3 root      20   0       0      0      0 S   0.0  0.0   0:06.42 [ksoftirqd/0]
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 [kworker/0:0H]
    7 root      rt   0       0      0      0 S   0.0  0.0   0:02.71 [migration/0]
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 [rcu_bh]
    9 root      20   0       0      0      0 S   0.0  0.0   5:04.19 [rcu_sched]
   10 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 [lru-add-drain]
   11 root      rt   0       0      0      0 S   0.0  0.0   0:04.16 [watchdog/0]
```

We use Prometheus and Grafana dashboards

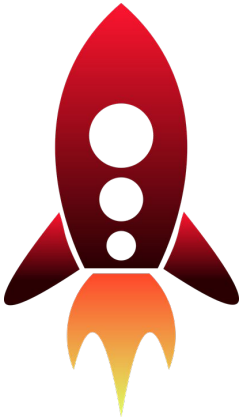
- Download from Prometheus and Grafana
- Unzip into a directory and use the provided configurations.
- Launch commands in separate console shells
- We won't be installing them today but see the appendix for instructions.

prometheus.exe --config.file prometheus.yml

grafana-server.exe

Configure this with the browser
http://localhost:3000/
Default user:password
(admin:admin) change it!

Example: Monitoring Rollouts



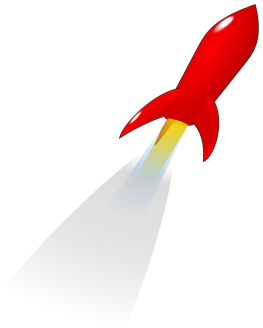
Example: Rollout

Rollout begins with SCCM - server on AWS
~2k clients peaking at 40% cpu load and
230mb resident size

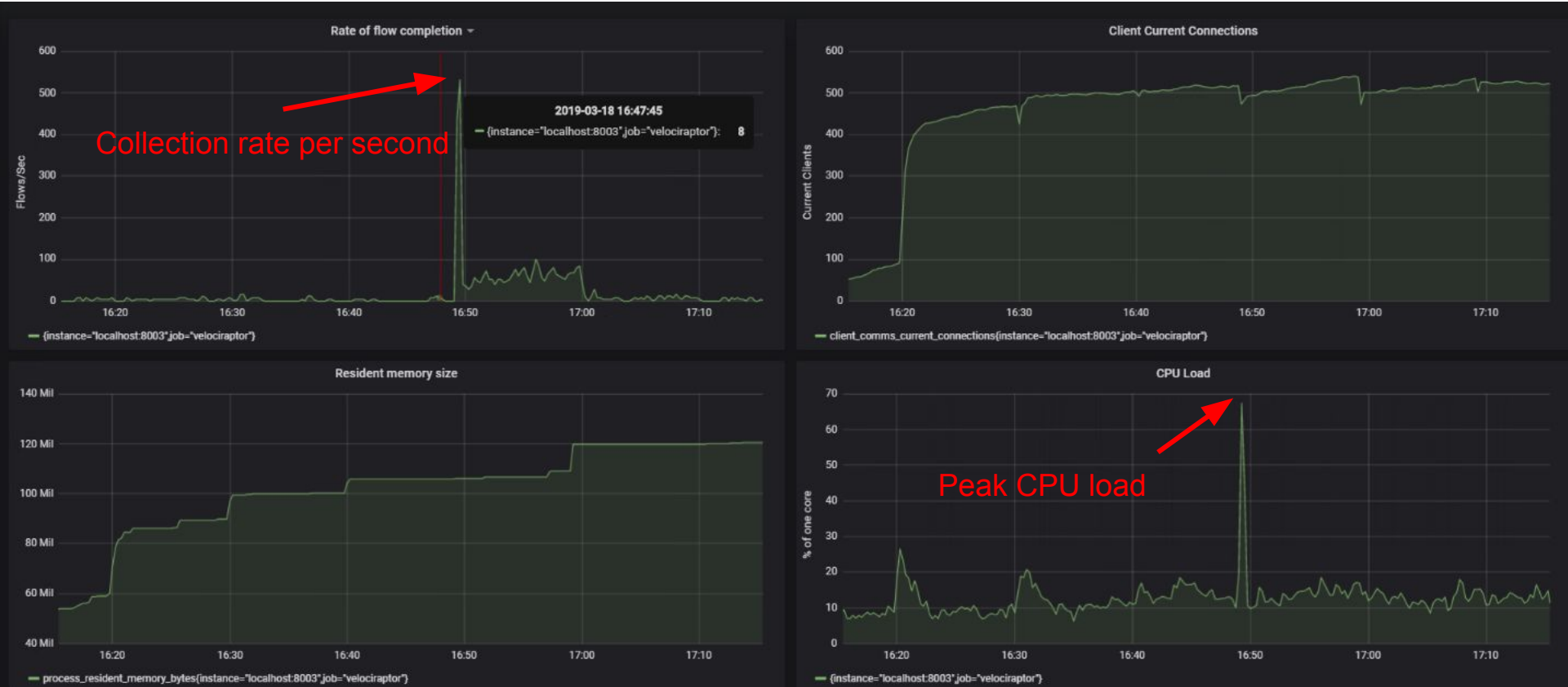
Interrogate flows

Keep an eye on CPU load
and peak memory use

Launch a hunt across the fleet

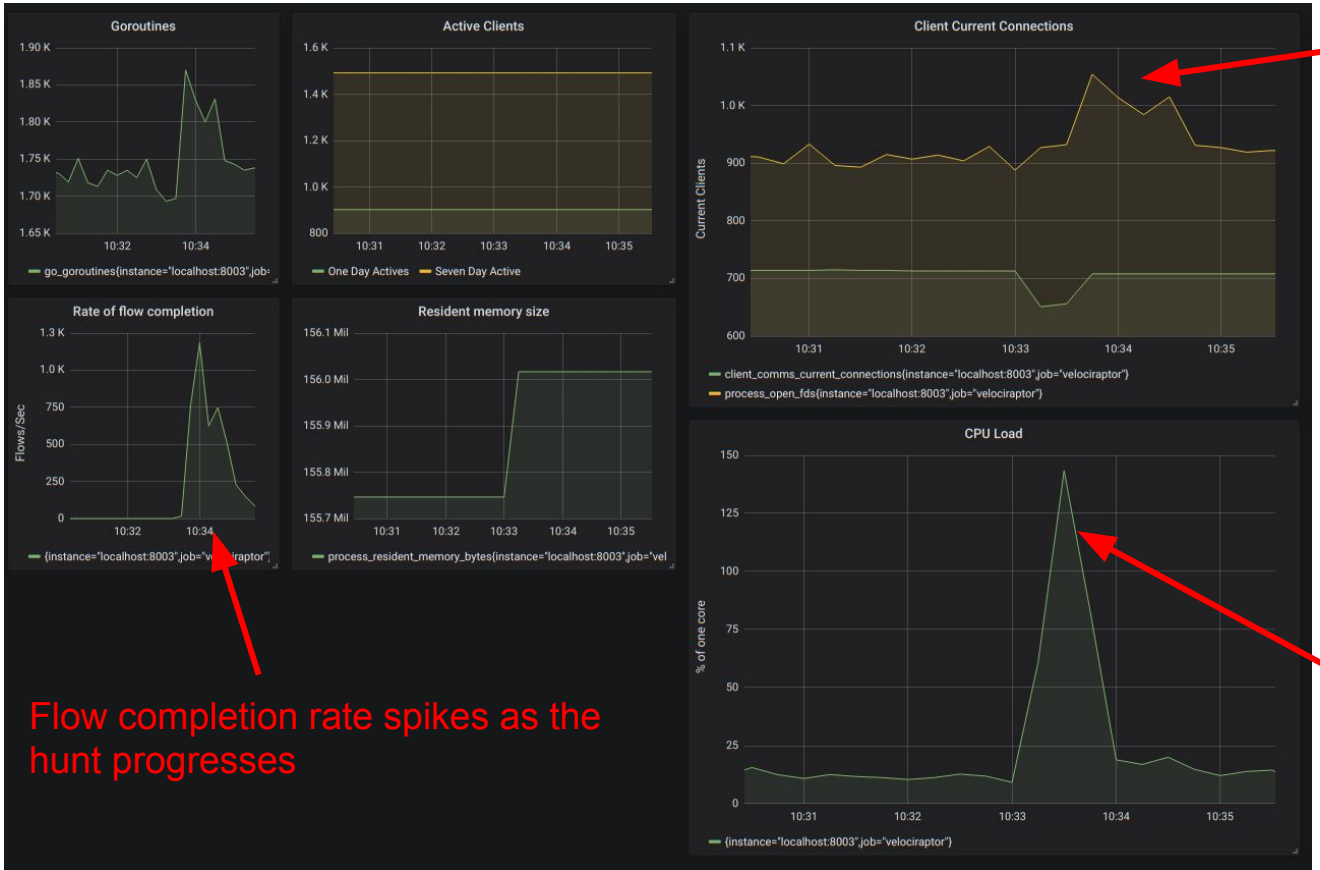


Typical hunt - Collect All Chrome Extensions



Hunt for IOC across the fleet

Open file handles increases temporarily as results are written to disk. Normally open file handles are a bit more than connected clients.



Flow completion rate spikes as the hunt progresses

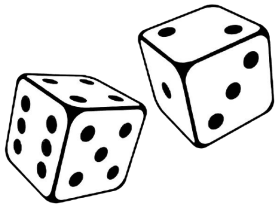
CPU Utilization increases with hunt start then falls off when all the clients are done.

Velociraptor is extremely efficient

1. Most operations occur on the end point via the VQL queries.
2. Server just writes the results to disk
3. Post processing can be done via the API (see later)
4. Server load is very low - typically you can get away with a single frontend even for medium to large deployment size.
5. We typically use larger slower disks for the file store (Cheaper)
   a. The file store accepts uploaded bulk data and VQL CSV files
   b. These are always written and appended, never deleted or modified.
   c. We can implement any desirable archiving/purging policies - everything is just a file.

Now it is your turn!

Deploy Velociraptor on your own machine



Architecture Overview

Client

Send Chan

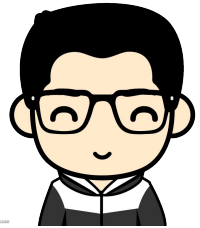
Recv Chan

Server

Front End

GUI

TLS (Self signed or Letsencrypt)



Externally accessible URL
Client.server_urls
In practice this will need a DNS name!

File Store

Path on disk to store files
Datastore.filestore_directory

Data store

Path on disk to store metadata
Datastore.location

Create a deployment configuration

F:\>velociraptor.exe config generate > velo.config.yaml

Generates new keys

F:\>velociraptor.exe --config velo.config.yaml user add mic

Add GUI user
Use --read_only to add read only users

F:\>velociraptor.exe --config velo.config.yaml frontend -v

Start frontend

```
F:\>velociraptor.exe --config velo.config.yaml frontend -v
[INFO] 2019-03-19T05:23:52-07:00 Starting Frontend. {"build_time":"2019-03-19T22:23:42+10:00","commit
[INFO] 2019-03-19T05:23:52-07:00 Loaded 88 built in artifacts
[INFO] 2019-03-19T05:23:52-07:00 Launched Prometheus monitoring server on 127.0.0.1:8003
[INFO] 2019-03-19T05:23:52-07:00 Frontend is ready to handle client TLS requests at 0.0.0.0:8000
[INFO] 2019-03-19T05:23:52-07:00 Starting hunt manager.
[INFO] 2019-03-19T05:23:52-07:00 Launched gRPC API server on 127.0.0.1:8001
[INFO] 2019-03-19T05:23:52-07:00 GUI is ready to handle TLS requests {"listenAddr":"127.0.0.1:8889"}
[INFO] 2019-03-19T05:23:52-07:00 Starting hunt dispatcher.
[INFO] 2019-03-19T05:23:52-07:00 Starting stats collector.
```

Velociraptor | Home

Certificate error https://localhost:8889/app.html

Velociraptor

Search Box

MANAGEMENT

Hunt Manager

Server Files

Welcome to Velociraptor

Query for a system to view in the search box

Type a search term to search for a machine using either a

For example: *label:mylabel*, *host:my_hostname*, *user:usrn*

Self signed SSL

Certificate Information

VelociraptorServer

VelociraptorServer

Certificate error

Issued by

Velociraptor CA

Valid from

Tuesday, March 19, 2019 5:20:08 AM

Valid to

Wednesday, March 18, 2020 5:20:08 AM

Subject organization

Velociraptor

Serial number

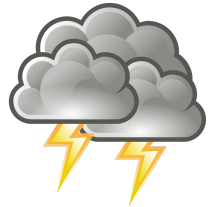
Export to file

Standalone deployment

- In this mode Velociraptor self signs its SSL cert.
- You can limit GUI connectivity by binding it to 127.0.0.1 (default)
- By default uses basic auth with a fixed password provided by the admin.

This mode is useful for standalone isolated deployment (e.g. behind NAT or inside corp network).

Cloud based deployment



- When deploying in the cloud use “autocert” mode.
- Velociraptor will get and manage its own certs from let’s encrypt automatically.
- Optionally we can configure Velociraptor to use Google OAUTH. Then you can specify G-Suite password policy, 2FA etc.

This mode is useful when there is direct internet connectivity to the server.

Caveat - in this mode you must serve the GUI over port 443 and ports 80 and 443 must be externally accessible by any IP. Bonus: you get user’s GSuite avatars!

2. Create a client to deploy

- First make a client configuration from the deployment configuration:

```
F:\>velociraptor.exe --config velo.config.yaml config client > velo_client.yaml
```

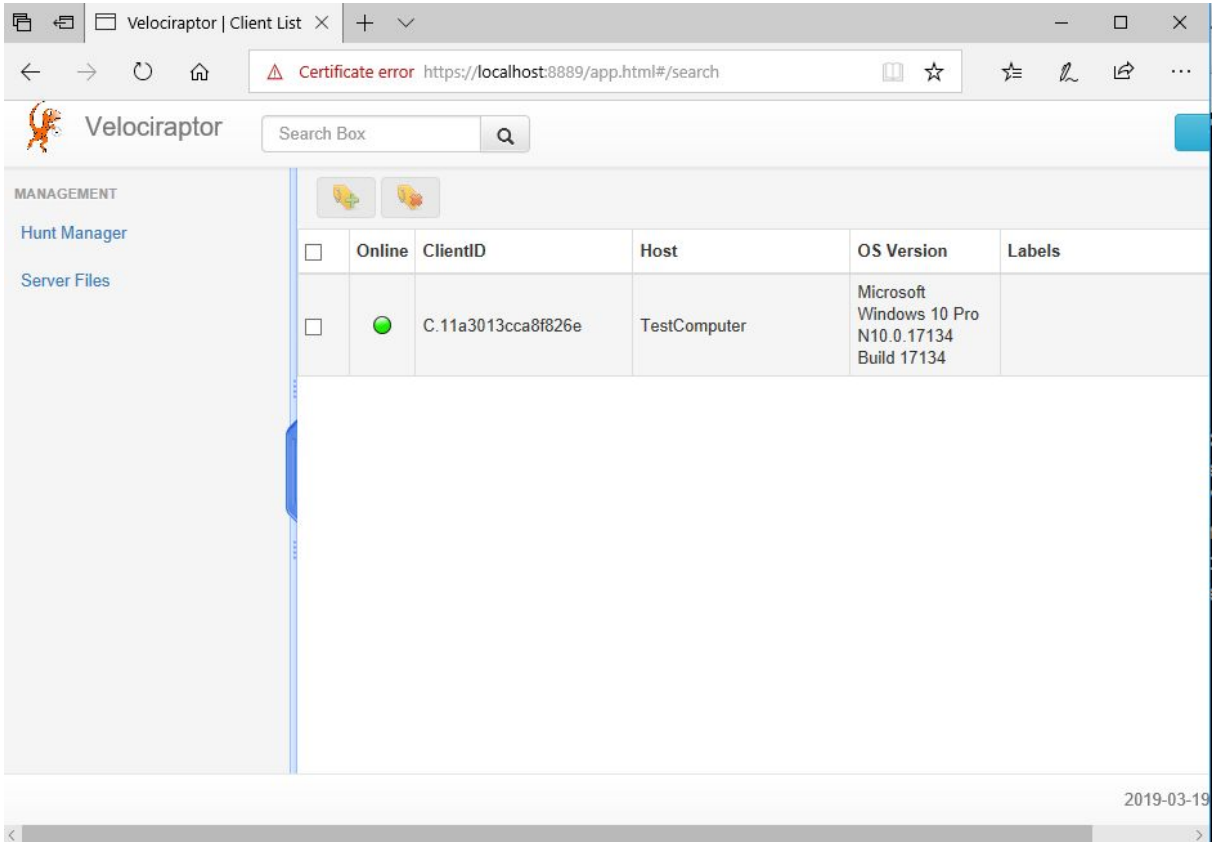
- Client config allows a client to connect to the deployment (crypto keys etc).
- Clients self enroll when they first connect - derive unique client id.
- The Velociraptor client is a single statically linked binary - no need for package management, dependencies etc - run anywhere.

Start the client manually with verbose output

Administrator: Command Prompt - velociraptor.exe --config velo_client.yaml client -v

```
F:\>velociraptor.exe --config velo_client.yaml client -v
[INFO] 2019-03-19T06:05:22-07:00 Starting Crypto for client C.11a3013cca8f826e
[INFO] 2019-03-19T06:05:22-07:00 Expecting self signed certificate for server.
[INFO] 2019-03-19T06:05:22-07:00 Starting HTTPCommunicator: [https://localhost:8000/]
[INFO] 2019-03-19T06:05:22-07:00 Received PEM for VelociraptorServer from https://localhost:8000/
[INFO] 2019-03-19T06:05:22-07:00 Receiver: Connected to https://localhost:8000/reader
[INFO] 2019-03-19T06:05:22-07:00 Receiver: sent 706 bytes, response with status: 200 OK
[INFO] 2019-03-19T06:05:22-07:00 Received request: session_id:"aff4:/clients/C.11a3013cca8f826e/flows/F.Moni
toring" request_id:1 name:"UpdateEventTable" args:"\n\322\003\022\245\002\n\242\002LET Generic_Client_Stats_
0_0 = SELECT * from foreach(\n row={\n   SELECT UnixNano FROM clock(period=atoi(string=Frequency))\n },\n qu
ery={\n   SELECT UnixNano / 1000000000 as Timestamp,\n          Times.user + Times.system as CPU,\n
 MemoryInfo.RSS as RSS\n   FROM pslist(pid=getpid())\n })\n\022\216\001\n&SELECT * FROM Generic_Client_Stats
_0_0\022\035Artifact Generic.Client.Stats\032EAn Event artifact which generates client's CPU and memory stat
istics.\032\017\n\tFrequency\022\002100d\305\001\000\000\310B\020\001" source:"VelociraptorServer" auth_stat
e:AUTHENTICATED args_rdf_name:"VQLEventTable" task_id:1553000722356134 client_type:VELOCIRAPTOR
[INFO] 2019-03-19T06:05:22-07:00 Starting Artifact Generic.Client.Stats
[INFO] 2019-03-19T06:05:23-07:00 Sender: Connected to https://localhost:8000/control
[INFO] 2019-03-19T06:05:23-07:00 Sender: sent 755 bytes, response with status: 200 OK
[INFO] 2019-03-19T06:05:23-07:00 Receiver: Connected to https://localhost:8000/reader
[INFO] 2019-03-19T06:05:23-07:00 Receiver: sent 723 bytes, response with status: 200 OK
[INFO] 2019-03-19T06:07:03-07:00 Sender: Connected to https://localhost:8000/control
[INFO] 2019-03-19T06:07:03-07:00 Sender: sent 1171 bytes, response with status: 200 OK
```

Search for the client in the GUI



How do I deploy Velociraptor to my endpoints?

1. Interactive client - just like we just did
   - Useful for debugging - making sure we have connectivity etc.
2. Agentless configuration
   - Push Velociraptor via group policy - configure to run for specified time and then exit.
3. Self installation:
   - Share velociraptor on a network share (similar to agentless above)
   - With group policy (or interactively) push the command

```
\\share\Velociraptor --config \\share\velo.conf service install
```

4. Build an MSI and push via SCCM
   - Can tweak the name of the service, binaries etc. Use provided wix file.
5. For cloud endpoints can specify in VM metadata startup script
   - Exact mechanism depends on your cloud provider.

Exercise: Interactively investigate the endpoint

- Locate the $MFT - master file table of your NTFS drive.
- Download the $MFT for later processing.
- Locate your user’s NTUSER.dat (c:\users\<username>\ntuser.dat)
- Try to download it the regular way
  - It should be locked it wont work (see the error logs)
  - Grab it using raw NTFS access
- Check for run keys
  - HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
  - HKEY_USERS\<SID>\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

Interactive shell

- Sometimes it is very useful to run shell commands on an endpoint when interactively investigating it.
- Velociraptor can run an interactive shell on the server only. This feature is not available from the GUI and requires server level access.
- Try it:

```
velociraptor.exe --config velo.config.yaml shell C.11a3013cca8f826e
```

Exercise: Get a shell on your endpoint.

Administrator: Command Prompt - velociraptor.exe --config velo.config.yaml shell C.11a3013cca8f826e

```
F:\>velociraptor.exe --config velo.config.yaml shell C.11a3013cca8f826e
C.11a3013cca8f826e (TestComputer) >ping www.google.com
Running ping www.google.com on C.11a3013cca8f826e


Pinging www.google.com [172.217.25.132] with 32 bytes of data:
Reply from 172.217.25.132: bytes=32 time=23ms TTL=53
Reply from 172.217.25.132: bytes=32 time=23ms TTL=53
Reply from 172.217.25.132: bytes=32 time=23ms TTL=53
Reply from 172.217.25.132: bytes=32 time=23ms TTL=53

Ping statistics for 172.217.25.132:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 23ms, Maximum = 23ms, Average = 23ms

C.11a3013cca8f826e (TestComputer) >
```

Shell level auditing - Automated Shell artifact

Search Box

0 mic

- file
- ntfs
- registry
- artifacts
- monitoring
  - Artifact Gene
  - Artifact Shell

monitoring > Artifact Shell

Download	Name	Size	Mode	Timestamp
	2019-03-22	4959	-r--r--r--	2019-03-22T23:15:59.3593188-07:00

monitoring > Artifact Shell > 2019-03-22

Stats | Download | TextView | HexView | CSVView

Show 10 entries

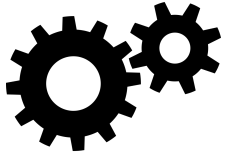
Search: ping

Timestamp	Argv	Stdout	Stderr	ReturnCode
1553321729	{"Argv": ["ping","www.google.com"]}	Pinging www.google.com [172.217.25.132] with 32 bytes of data: Reply from 172.217.25.132: bytes=32 time=23ms TTL=53 Reply from 172.217.25.132: bytes=32 time=23ms TTL=53 Reply from 172.217.25.132: bytes=32 time=23ms TTL=53 Reply from 172.217.25.132: bytes=32 time=23ms TTL=53 Ping statistics for 172.217.25.132: Packets: Sent = 4, Received = 4, Lost = 0 (0% loss), Approximate round trip times in milli-seconds: Minimum = 23ms, Maximum = 23ms, Average = 23ms		0
		Pinging www.google.com [172.217.25.132] with 32 bytes of data: Reply from 172.217.25.132: bytes=32 time=23ms TTL=53 Reply from 172.217.25.132: bytes=32 time=23ms		

Module 3

Velociraptor Artifacts

Automation FTW!

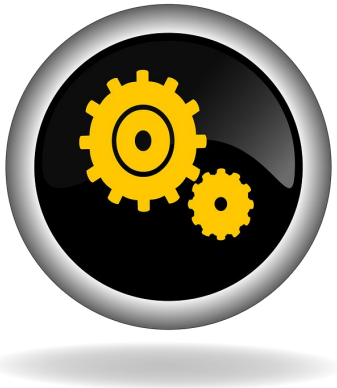


So far we saw how to use Velociraptor to interactively read files on the endpoint. That is pretty boring! The real power rests in Artifacts and VQL.

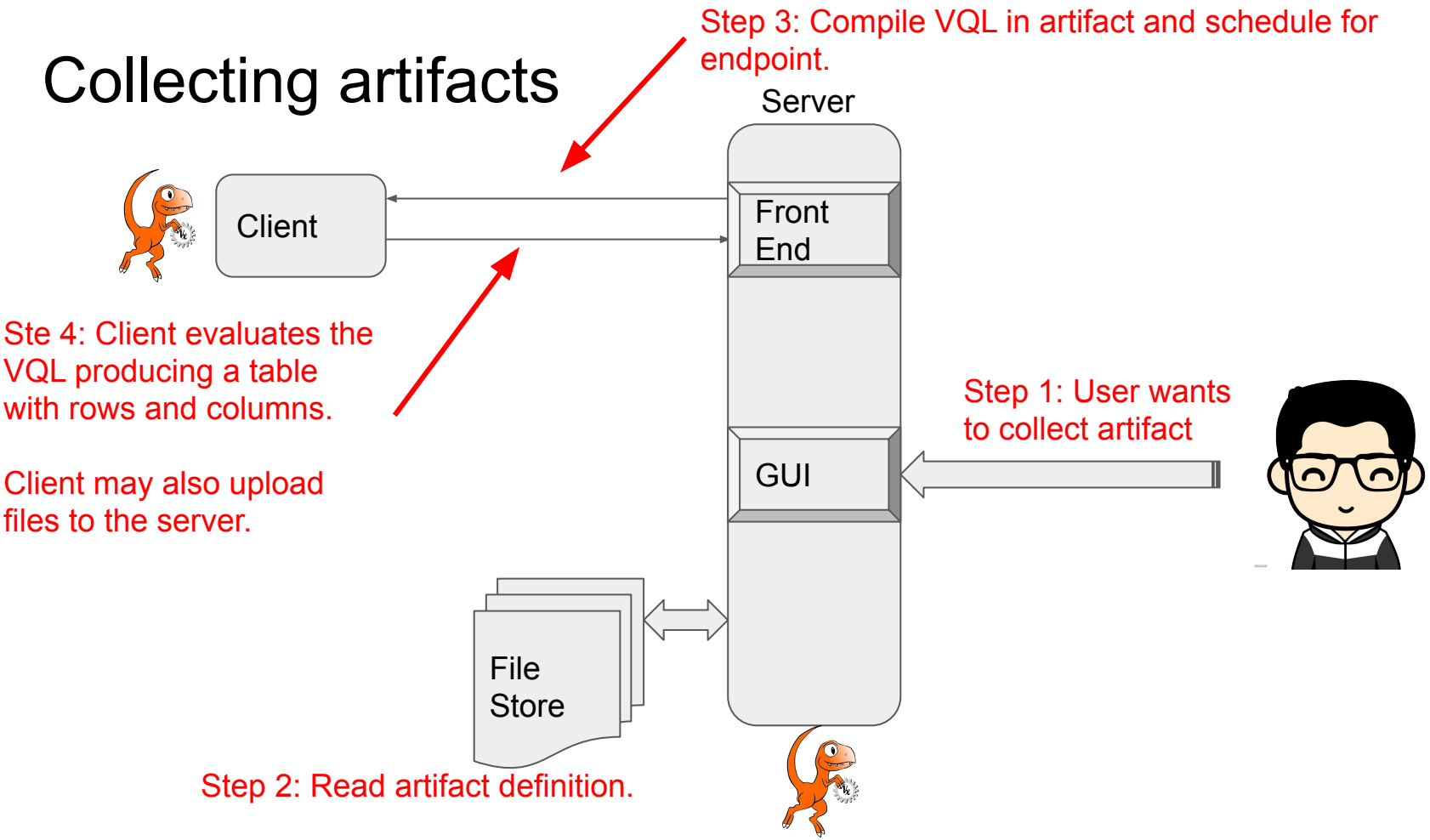
- What if we could tell the endpoint to collect arbitrary information, parse it and filter it on demand:
  - Without needing to push new code to the endpoint?
  - Without having to upgrade clients in the field?
- Then we could flexibly adapt to emerging threats in minutes!
  - Search registry for Yara sig, then parse out the filename, then upload the file to the server.
  - Search files in this directory for a zip signature, then search within the zip file for a keyword.
- What if we could collect all these from thousands of endpoints in seconds?

What are Velociraptor Artifacts?

- Define a specific group of files to fetch as well as a table of data
- The artifact also defines how to fetch this data using a **VQL query.**
- Declare parameters with default values which users can override
  - Allows users to customize the artifact if needed
  - Allows artifacts to be used by other artifacts!
- Once an artifact is defined, users don't need to worry about the VQL - they can collect the artifact at a click of a button!
- This makes artifacts reusable by many other users.



Collecting artifacts



Lets try this!

Collect amcache from your machine

AMCache Artifact

Name and description give human readable context around the artifact.

Parameters allow the artifact to be customized

Preconditions test if the artifact is supported.

A series of VQL queries is run which produce a result set (table).

```yaml
name: Windows.System.Amcache
description: |
  Get information from the system's amcache.

  The Amcache.hve file is a registry file that stores the information
  of executed applications. Amcache.hve records the recent processes
  that were run and lists the path of the files that's executed which
  can then be used to find the executed program.

  This artifact works on Windows 10 1607 version.

  References:
    https://www.andreafortuna.org/cybersecurity/amcache-and-shimcache-in-
    https://www.ssi.gouv.fr/uploads/2019/01/anssi-coriin_2019-analysis_am

parameters:
  - name: amCacheGlob
    default: "%SYSTEMROOT%/appcompat/Programs/Amcache.hve"
  - name: amCacheRegPath
    default: /Root/InventoryApplicationFile/*

sources:
  - precondition:
      SELECT OS From info() where OS = 'windows'
    queries:
      - |
        SELECT FileId,
               Key.FullPath as Key,
               timestamp(epoch=Key.Mtime.Sec) as LastModified,
               LowerCaseLongPath as Binary,
               Name,
               Size,
               ProductName,
               Publisher,
               Version,
               BinFileVersion
        FROM foreach(
          row={
            SELECT FullPath from glob(globs=expand(path=amCacheGlob))
          }, query={
            SELECT * from read_reg_key(
               globs=url(scheme='ntfs', path=FullPath, fragment=amCacheRe
               accessor='raw_reg'
            )
        })
```

Collect the amcache from your machine

This part of the VFS shows all instances of this artifact collected from that endpoint

Velociraptor

Search Box

0 mic

TestComputer
Access reason: test
Status: 49 seconds ago
192.168.0.18:50988

Host Information
Start new flows
Collect Artifacts
Browse Virtual Filesystem
Manage launched flows

MANAGEMENT
Hunt Manager
Server Files

LINKS
Monitoring

- file
- ntfs
- registry
- artifacts
  - Artifact Windows.System.Amcache
  - Artifact Windows.Triage.Collectors.Ch
  - Artifact Windows.Triage.Collectors.Fi
  - Artifact Windows.Triage.Collectors.Re
  - Artifact Windows.Triage.WebBrowser
- monitoring

artifacts > Artifact Windows.System.Amcache

Download	Name	Size	Mode	Timestamp
	F.2d99deb3.csv	430933	-r--r--r--	2019-03-22T00:09:31.063941445+10:00

Each artifact specifies its own set of columns

LastModified	Binary	Name	Size	ProductName	Publisher	Version
2018-12-15T04:19:36-08:00	c:\program files\git\usr\bin\diff.exe	diff.exe	206813			
2019-02-26T21:06:55-08:00	c:\program files (x86)\windows kits\10\debuggers\x86\gflags.exe	gflags.exe	82232	microsoft® windows® operating system	microsoft corporation	10.0.16299.91 (winbuild.160101.08
2019-02-26T21:06:55-08:00	c:\program files (x86)\windows kits\10\debuggers\x86\cdb.exe	cdb.exe	135480	microsoft® windows® operating system	microsoft corporation	10.0.16299.91 (winbuild.160101.08
2018-12-15T04:19:56-08:00	c:\windows\system32\appverif.exe	appverif.exe	207640	microsoft® windows® operating system	microsoft corporation	6.3.9600.16384 (winblue_rtm.130821623)
2018-12-15T04:19:36-08:00	c:\program files\git\usr\bin\cygwin-console-helper.exe	cygwin-console-helper.exe	160592			
2019-02-26T21:05:51-08:00	c:\program files (x86)\windows kits\10\bin\10.0.16299.0\arm\dxcapsviewer.exe	dxcapsviewer.exe	172592	microsoft® windows® operating system	microsoft corporation	10.0.16299.91 (winbuild.160101.08

2019-03-21 14:11:57 UTC

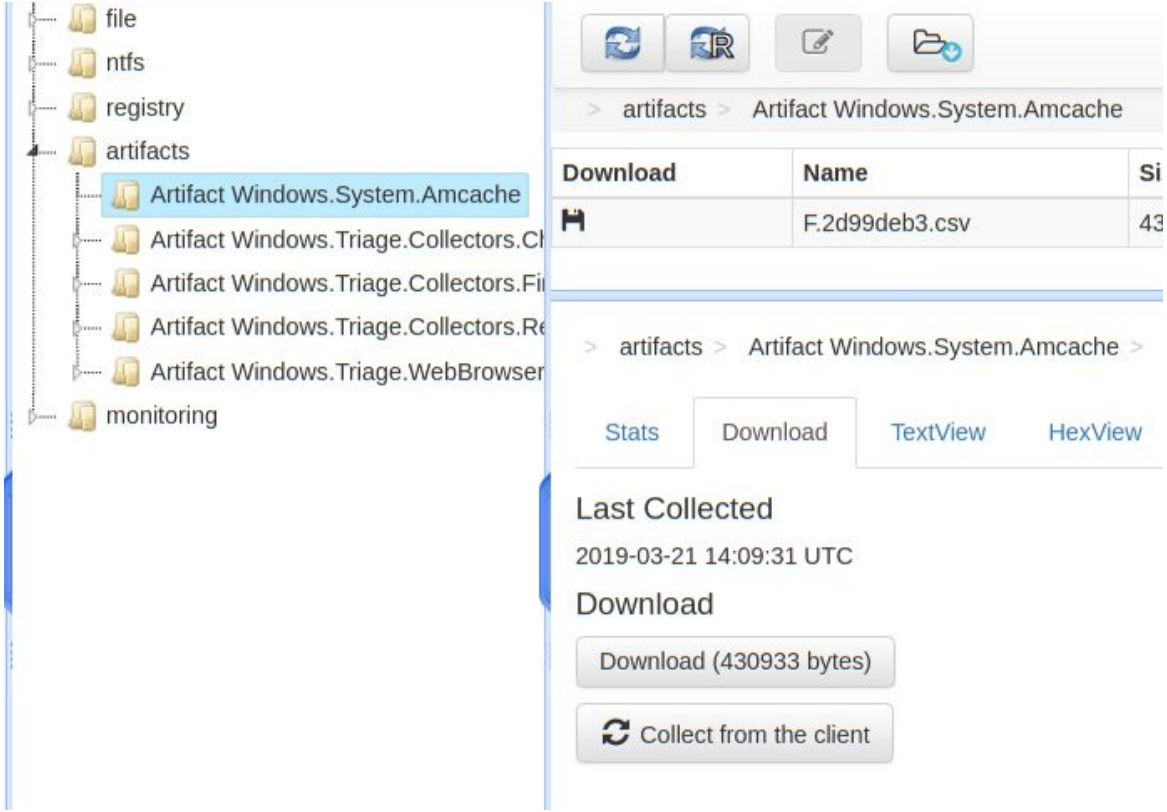
Hunt Results

Artifacts return:

1. A table with columns and rows.
2. Potentially a set of files

You can download a Zip file containing all rows as a CSV file and all downloaded files from the **Managed Launched Flows/Results.**

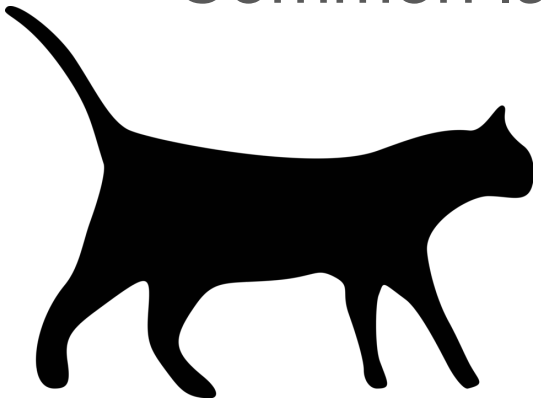
Or just download the CSV file from the VFS view.

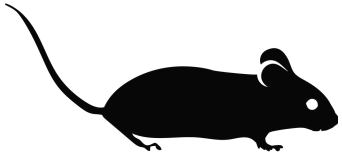


Module 4

Hunting for evil

Common lateral movement techniques





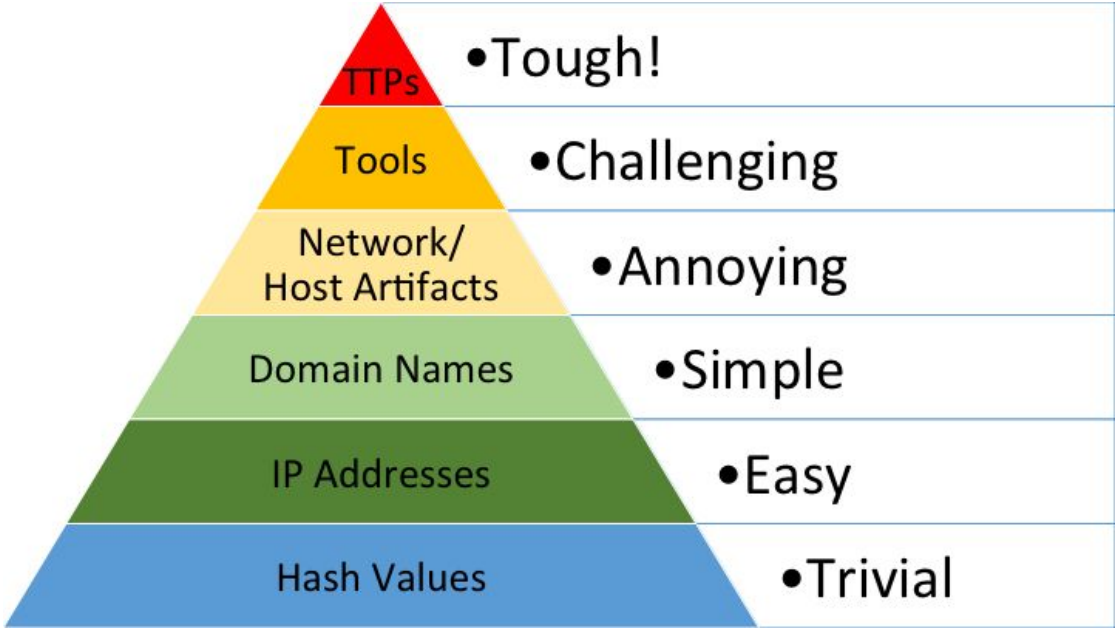
The pyramid of pain!

Indicators of compromise come in many flavors:

Indicators which are easy to detect are also easy for attackers to modify.

Detecting the tools or techniques means it is very hard for attackers to adapt.

We should be aiming for that!



http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html

SANS Hunt Evil poster

PsExec

EVENT LOGS

- security.evtx
  - 4648 – Logon specifying alternate credentials
    - Current logged-on User Name
    - Alternate User Name
    - Destination Host Name/IP
    - Process Name

REGISTRY

- NTUSER.DAT
  - Software\SysInternals\PsExec\EulaAccepted
- ShimCache – SYSTEM
  - psexec.exe
- BAM/DAM – SYSTEM – Last Time Executed
  - psexec.exe
- AmCache.hve – First Time Executed
  - psexec.exe

FILE SYSTEM

- Prefetch – C:\Windows\Prefetch\
  - psexec.exe-{hash}.pf
  - Possible references to other files accessed by psexec.exe, such as executables copied to target system with the "-c" option
- File Creation
  - psexec.exe file downloaded and created on local host as the file is not native to Windows


```
psexec.exe \\host -accepteula -d -c c:\temp\evil.exe
```

EVENT LOGS

- security.evtx
  - 4624 Logon Type 3 (and Type 2 if "-u" Alternate Credentials are used)
    - Source IP/Logon User Name
  - 4672
    - Logon User Name
    - Logon by a user with administrative rights
    - Requirement for access default shares such as C$ and ADMIN$
  - 5140 – Share Access
    - ADMIN$ share used by PsExec
- system.evtx
  - 7045
    - Service Install

REGISTRY

- New service creation configured in SYSTEM\CurrentControlSet\Services\PSEXESVC
  - "-r" option can allow attacker to rename service
- ShimCache – SYSTEM
  - psexesvc.exe
- AmCache.hve First Time Executed
  - psexesvc.exe

FILE SYSTEM

- Prefetch – C:\Windows\Prefetch\
  - psexesvc.exe-{hash}.pf
  - evil.exe-{hash}.pf
- File Creation
  - User profile directory structure created unless "-e" option used
  - psexesvc.exe will be placed in ADMIN$ (\Windows) by default, as well as other executables (evil.exe) pushed by PsExec

Scheduled Tasks

EVENT LOGS

- security.evtx
  - 4648 – Logon specifying alternate credentials
    - Current logged-on User Name
    - Alternate User Name
    - Destination Host Name/IP
    - Process Name

REGISTRY

- ShimCache – SYSTEM
  - at.exe
  - schtasks.exe
- BAM/DAM – SYSTEM – Last Time Executed
  - at.exe
  - schtasks.exe
- AmCache.hve - First Time Executed
  - at.exe
  - schtasks.exe

FILE SYSTEM

- Prefetch – C:\Windows\Prefetch\
  - at.exe-{hash}.pf
  - schtasks.exe-{hash}.pf

```
at \\host 13:00 "c:\temp\evil.exe"
schtasks /CREATE /TN taskname /TR c:\temp\evil.exe /SC once /RU "SYSTEM" /ST 13:00 /S host /U username
```

EVENT LOGS

- security.evtx
  - 4624 Logon Type 3
    - Source IP/Logon User Name
  - 4672
    - Logon User Name
    - Logon by a user with administrative rights
    - Requirement for accessing default shares such as C$ and ADMIN$
  - 4698 – Scheduled task created
  - 4702 – Scheduled task updated
  - 4699 – Scheduled task deleted
  - 4700/4701 – Scheduled task enabled/disabled
- Microsoft-Windows-TaskScheduler%4Maintenance.evtx
  - 106 – Scheduled task created
  - 140 – Scheduled task updated
  - 141 – Scheduled task deleted
  - 200/201 – Scheduled task executed/completed

REGISTRY

- SOFTWARE
  - Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks
  - Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\
- ShimCache – SYSTEM
  - evil.exe
- AmCache.hve – First Time Executed
  - evil.exe

FILE SYSTEM

- File Creation
  - evil.exe
  - Job files created in C:\Windows\Tasks
  - XML task files created in C:\Windows\System32\Tasks
    - Author tag under "RegistrationInfo" can identify:
      - Source system name
      - Creator username
- Prefetch – C:\Windows\Prefetch\
  - evil.exe-{hash}.pf

Services

EVENT LOGS

REGISTRY

- ShimCache – SYSTEM
  - sc.exe
- BAM/DAM – SYSTEM – Last Time Executed
  - sc.exe
- AmCache.hve – First Time Executed
  - sc.exe

FILE SYSTEM

- Prefetch – C:\Windows\Prefetch\
  - sc.exe-{hash}.pf

```
sc \\host create servicename binpath= "c:\temp\evil.exe"
sc \\host start servicename
```

EVENT LOGS

- security.evtx
  - 4624 Logon Type 3
    - Source IP/Logon User Name
  - 4697
    - Security records service install, if enabled
    - Enabling non-default Security events such as ID 4697 are particularly useful if only the Security logs are forwarded to a centralized log server
- system.evtx
  - 7034 – Service crashed unexpectedly
  - 7035 – Service sent a Start/Stop control
  - 7036 – Service started or stopped
  - 7040 – Start type changed (Boot | On Request | Disabled)
  - 7045 – A service was installed on the system

REGISTRY

- SYSTEM
  - \CurrentControlSet\Services\
  - New service creation
- ShimCache – SYSTEM
  - evil.exe
  - ShimCache records existence of malicious service executable, unless implemented as a service DLL
- AmCache.hve – First Time Executed
  - evil.exe

FILE SYSTEM

- File Creation
  - evil.exe or evil.dll malicious service executable or service DLL
- Prefetch – C:\Windows\Prefetch\
  - evil.exe-{hash}.pf

WMI/WMIC

EVENT LOGS

- security.evtx
  - 4648 – Logon specifying alternate credentials
    - Current logged-on User Name
    - Alternate User Name
    - Destination Host Name/IP
    - Process Name

REGISTRY

- ShimCache – SYSTEM
  - wmic.exe
- BAM/DAM – SYSTEM – Last Time Executed
  - wmic.exe
- AmCache.hve – First Time Executed
  - wmic.exe

FILE SYSTEM

- Prefetch – C:\Windows\Prefetch\
  - wmic.exe-{hash}.pf


```
wmic /node:host process call create "evil.exe"
Invoke-WmiMethod -Computer host -Class Win32_Process -Name create -Argument "c:\temp\evil.exe"
```

EVENT LOGS

- security.evtx
  - 4624 Logon Type 3
    - Source IP/Logon User Name
  - 4672
    - Logon User Name
    - Logon by an a user with administrative rights
- Microsoft-Windows-WMI-Activity%4Operational.evtx
  - 5857
    - Indicates time of wmiprvse execution and path to provider DLL – attackers sometimes install malicious WMI provider DLLs
  - 5860, 5861
    - Registration of Temporary (5860) and Permanent (5861) Event Consumers. Typically used for persistence, but can be used for remote execution.

REGISTRY

- ShimCache – SYSTEM
  - wmiprvse.exe
  - evil.exe
  - mofcomp.exe
- AmCache.hve – First Time Executed
  - wmiprvse.exe
  - evil.exe
  - mofcomp.exe

FILE SYSTEM

- File Creation
  - evil.exe
  - evil.mof – .mof files can be used to manage the WMI Repository
- Prefetch – C:\Windows\Prefetch\
  - evil.exe-{hash}.pf
  - wmiprvse.exe-{hash}.pf
  - mofcomp.exe-{hash}.pf
- Unauthorized changes to the WMI Repository in C:\Windows\System32\wbem\Repository

https://digital-forensics.sans.org/media/SANS_Poster_2018_Hunt_Evil_FINAL.pdf

PsExec: Running sysinternals tools

- Many APT groups use sysinternal tools like psexec for lateral movement or privilege escalations.
- Sysinternal tools require users to accept a EULA.
- This makes them add an “EulaAccepted” value to the registry.
- We can hunt for this to see the first time a particular sysinternal tool was run on the system (from the registry key modification time).
- This works best for machines which should never run such tools (i.e. non-developer/sysadmin machines) with a clean build.
- Test this by running (this gives a system shell):

```
PsExec.exe -s -i cmd.exe
```

```
name: Windows.Registry.Sysinternals.Eulacheck
description: |
  Checks for the Accepted Sysinternals EULA from the registry key
  "HKCU\Software\Sysinternals\[TOOL]\".  When a Sysinternals tool is
  first run on a system, the EULA must be accepted. This writes a
  value called EulaAccepted under that key.

parameters:
  - name: Sysinternals_Reg_Key
    default: HKEY_USERS\*\Software\Sysinternals\*

sources:
  - precondition:
      SELECT OS From info() where OS = 'windows'

    queries:
  - LET users <= SELECT Name, UUID FROM Artifact.Windows.Sys.Users()
  - SELECT Key.Name as ProgramName,
          Key.FullPath as Key,
          timestamp(epoch=Key.Mtime.Sec) AS TimeAccepted,
          {
            SELECT Name FROM users WHERE UUID=regex_replace(
              source=Key.FullPath, re=".+\\\\(S-[^\\\\]+)\\\\.+", replace="$1")
          } as User,
          EulaAccepted
    FROM read_reg_key(globs=split(string=Sysinternals_Reg_Key, sep=',[\\s]*'))
```

Parameters can be overridden but have defaults

If the precondition returns no rows the artifact does not run.

One or more VQL statements. The last statement is a SELECT returning a sequence of rows

Who launched the artifact

Velociraptor

Search Box

0

mike@velocidex.com

TestComputer
Access reason: test
Status: 1 minutes ago
106.71.187.90:51137

Host Information

Start new flows

Collect Artifacts

Browse Virtual Filesystem

Manage launched flows

MANAGEMENT

Hunt Manager

Server Files

LINKS

Monitoring

State	Path	Flow Name	Creation Time	Last Active	Creator
	F.ae6dab50	ArtifactCollector	2019-03-20 12:49:15 UTC	2019-03-20 12:49:16 UTC	mike@velocidex.com
	F.9b37a73a	ArtifactCollector	2019-03-20 04:22:00 UTC	2019-03-20 04:22:04 UTC	

Flow Information | Download | Requests | Results | Log

Artifact Name

Windows.Registry.Sysinternals.Eulacheck

Show 10 entries

Search:

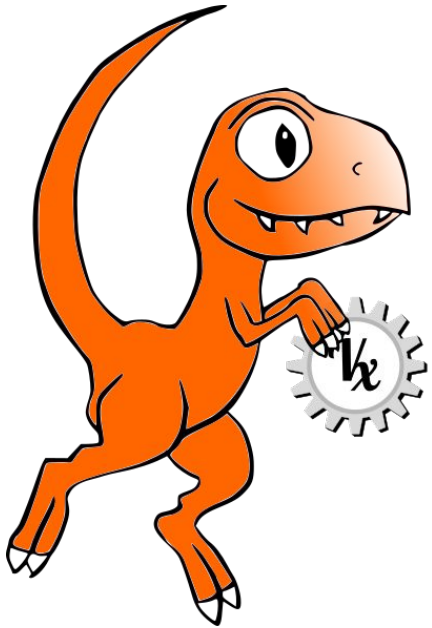
ProgramName	Key	TimeAccepted	User	EulaAccepted
AutoRuns	HKEY_USERS\S-1-5-21-546003962-2713609280-610790815-1001\Software\Sysinternals\AutoRuns	2019-02-07T03:42:29-08:00	test	1
Handle	HKEY_USERS\S-1-5-21-546003962-2713609280-610790815-1001\Software\Sysinternals\Handle	2019-01-06T18:04:12-08:00	test	1
Process Monitor	HKEY_USERS\S-1-5-21-546003962-2713609280-610790815-1001\Software\Sysinternals\Process Monitor	2018-12-31T04:31:26-08:00	test	1
PsExec	HKEY_USERS\S-1-5-21-546003962-2713609280-610790815-1001\Software\Sysinternals\PsExec	2018-11-09T22:35:11-08:00	test	1
sigcheck	HKEY_USERS\S-1-5-21-546003962-2713609280-610790815-1001\Software\Sysinternals\sigcheck	2019-02-03T09:01:45-08:00	test	1

Showing 1 to 5 of 5 entries

Previous 1 Next

Module 5

The Velociraptor Query Language



Why a query language?

- Able to dynamically adapt to changing requirements - without needing to rebuild clients or servers.
  - For example, a new IOC is released for detection of a specific threat
  - Immediately write a VQL artifact for the threat, upload the artifact and hunt everywhere for it.
  - Turn around from IOC to full hunt: A few minutes.
- Share artifacts with the community
  - VQL Artifacts are simply YAML files with VQL queries.
  - Can be easily shared and cross pollinate other Artifacts
  - Can be customized by users in the GUI in seconds.
- Public Artifact Reference here

What is VQL?

Column Selectors

VQL Plugin with Args

Filter Condition

SELECT X, Y, Z FROM plugin(arg=1) WHERE X = 1

Example - search files by glob

```
F:\>velociraptor.exe query "SELECT * from glob(globs='C:\\Users\\*\\ntuser.dat') limit 1"
[
 {
  "Atime": {
   "sec": 1553122388,
   "usec": 1553122388586794900
  },
  "Ctime": {
   "sec": 1551249159,
   "usec": 1551249159201822600
  },
  "Data": "",
  "FullPath": "\\C:\\Users\\someuser\\NTUSER.DAT",
  "GetLink": "",
  "IsDir": false,
  "IsLink": false,
  "ModTime": "2019-03-12T19:19:11.9014376-07:00",
  "Mode": 438,
  "Mtime": {
   "sec": 1552443551,
   "usec": 1552443551901437600
  },
  "Name": "NTUSER.DAT",
  "Size": 1048576,
  "Sys": {
   "FileAttributes": 8226,
```

```
F:\>velociraptor.exe query "SELECT FullPath from glob(globs='C:\\Users\\*\\ntuser.dat')" --format text
+-------------------------------+
|           FullPath            |
+-------------------------------+
| \C:\Users\test\NTUSER.DAT     |
| \C:\Users\user\NTUSER.DAT     |
| \C:\Users\Default\NTUSER.DAT  |
| \C:\Users\someuser\NTUSER.DAT |
+-------------------------------+
SELECT FullPath FROM
glob(globs="C:\\Users\\*\\ntuser.dat")

F:\>velociraptor.exe query "SELECT FullPath, Size from glob(globs='C:\\Users\\*\\ntuser.dat')" --format text
+-------------------------------+---------+
|           FullPath            |  Size   |
+-------------------------------+---------+
| \C:\Users\someuser\NTUSER.DAT | 1048576 |
| \C:\Users\test\NTUSER.DAT     | 1572864 |
| \C:\Users\user\NTUSER.DAT     | 1835008 |
| \C:\Users\Default\NTUSER.DAT  |  262144 |
+-------------------------------+---------+
SELECT FullPath, Size FROM
glob(globs="C:\\Users\\*\\ntuser.dat")

F:\>velociraptor.exe query "SELECT FullPath, Size, timestamp(epoch=Mtime.Sec) as Modified from glob(globs='C:\\Users\\*\\ntuser.dat')" --format text
+-------------------------------+---------+---------------------------+
|           FullPath            |  Size   |          Modified         |
+-------------------------------+---------+---------------------------+
| \C:\Users\someuser\NTUSER.DAT | 1048576 | 2019-03-12T19:19:11-07:00 |
| \C:\Users\test\NTUSER.DAT     | 1572864 | 2019-03-18T16:11:53-07:00 |
| \C:\Users\user\NTUSER.DAT     | 1835008 | 2019-03-19T19:42:48-07:00 |
| \C:\Users\Default\NTUSER.DAT  |  262144 | 2019-03-12T19:19:10-07:00 |
+-------------------------------+---------+---------------------------+
SELECT FullPath, Size, timestamp(epoch=Mtime.Sec) AS Modified FROM
glob(globs="C:\\Users\\*\\ntuser.dat")
```

VQL plugins return many rows and take various args

VQL functions return a single value and take args - operate one row at a time

Module 6

Data Collection

For triage and acquisition

Triage and data collection

- You get a call requesting to preserve user activity on a machine for an ongoing DFIR investigation.
- But you do not have Velociraptor deployed (and you do not have a server)!
- You can collect an artifact on the command line too.

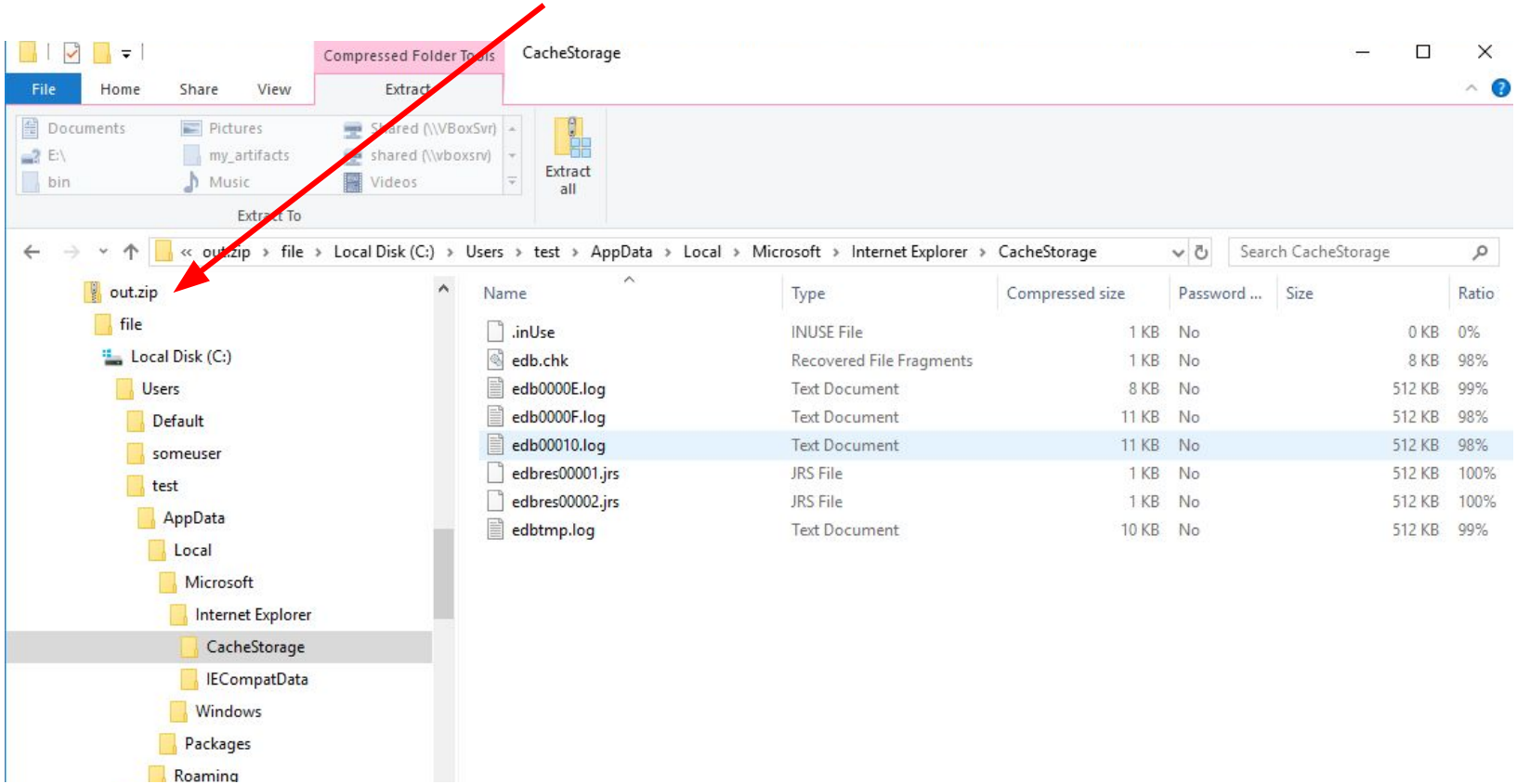
Velociraptor does not actually need a server to collect artifacts! We can collect artifacts into a zip file from the command line.

Collect this artifact (can be given multiple times to collect multiple artifacts). Triage artifacts just collect files.

Store output in this zip file (can be a file share).

```
F:\>velociraptor.exe artifacts -v collect Windows.Triage.WebBrowsers --output f:\output\out.zip
[INFO] 2019-03-21T10:24:19-07:00 Loaded 89 built in artifacts
[INFO] 2019-03-21T10:24:21-07:00 Collecting file \C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.defau
[INFO] 2019-03-21T10:24:21-07:00 Collecting file \C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.defau
[INFO] 2019-03-21T10:24:22-07:00 Collecting file \C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.defau
[INFO] 2019-03-21T10:24:22-07:00 Collecting file \C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wek
[INFO] 2019-03-21T10:24:22-07:00 Collecting file \C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wek
[INFO] 2019-03-21T10:24:22-07:00 Collecting file \C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wek
[INFO] 2019-03-21T10:24:22-07:00 Collecting file \C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wek
```

Zip file contains all the collected files as well as CSV with artifact result set



Artifact Windows.Triage.WebBrowsers.csv (read-only) - LibreOffice Calc

File Edit View Insert Format Styles Sheet Data Tools Window Help

A1 FullPath

	A	B	C	D	E	F
1	FullPath	Size	Modifed	Type	ZipPath	Md5
2	\C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.default\places.sqlite	5242880		Places	file/C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.default\places.sqlite	8e5b063c145ece34038def53
3	\C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.default\cookies.sqlite	524288		Cookies	file/C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.default\cookies.sqlite	e67845525eaf2b302dd7e64b
4	\C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.default\favicons.sqlite	5242880		Favicons	file/C:\Users\test\AppData\Roaming\Mozilla\Firefox\Profiles\cc9yc6pl.default\favicons.sqlite	6b3afdafb8a20ccce21ce269
5	\C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\roaming.lock	0		Edge folder	file/C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\roaming.lock	d41d8cd98f00b204e9800998
6	\C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat	8192		Edge folder	file/C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat	fad7631fd9ccb7a1b15ea80a
7	\C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat.LOG1	8192		Edge folder	file/C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat.LOG1	1ca5ac3ffcc4a4f99184627e4
8	\C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat.LOG2	8192		Edge folder	file/C:\Users\someuser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat.LOG2	794308be04ccdf3195f9d56a
9	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\roaming.lock	0		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\roaming.lock	d41d8cd98f00b204e9800998
10	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat	8192		Edge folder		
11	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat.LOG1	8192		Edge folder		
12	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat.LOG2	16384		Edge folder		
13	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\Dow61D1.tmp	459916		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\Dow61D1.tmp	85d57967e153ae3692a5ebe
14	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\Dow778D.tmp	459916		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\Dow778D.tmp	85d57967e153ae3692a5ebe
15	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL10A.tmp	90521		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL10A.tmp	6333d5f7cf90412442911364
16	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL114E.tmp	90521		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL114E.tmp	0c9fb5af469e19b971802564
17	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL140E.tmp	99185		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL140E.tmp	e3025b593c57865dae2d7d9
18	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL15F9.tmp	88713		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL15F9.tmp	1525ea93d32b3dcb971e0c5
19	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL175A.tmp	88661		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL175A.tmp	881e13ae4d10c64ac42edb1
20	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL179B.tmp	90521		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL179B.tmp	5cf978551194a67693d6dd63
21	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL1A2E.tmp	99181		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL1A2E.tmp	6fa72dcaeb64ba60f0469f89
22	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL1CF6.tmp	90521		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL1CF6.tmp	02c017c5d797e6680ac0251
23	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL21B2.tmp	90521		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL21B2.tmp	0a2ea76d96e5297e21c8cdc
24	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL22D9.tmp	100725		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL22D9.tmp	b20aff3bf584057ebba539f3d
25	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL2407.tmp	94269		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL2407.tmp	a8761426852fa438bb67b409
26	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL2434.tmp	99185		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL2434.tmp	259a78df6891e355bbc46bce
27	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL27C0.tmp	90521		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL27C0.tmp	60e4fb8bed8773912c16214f
28	\C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL2A82.tmp	99181		Edge folder	file/C:\Users\test\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\Temp\URL2A82.tmp	6f9f240fa60f2b70212918596

Exercise: Obtain a timeline of users home directory

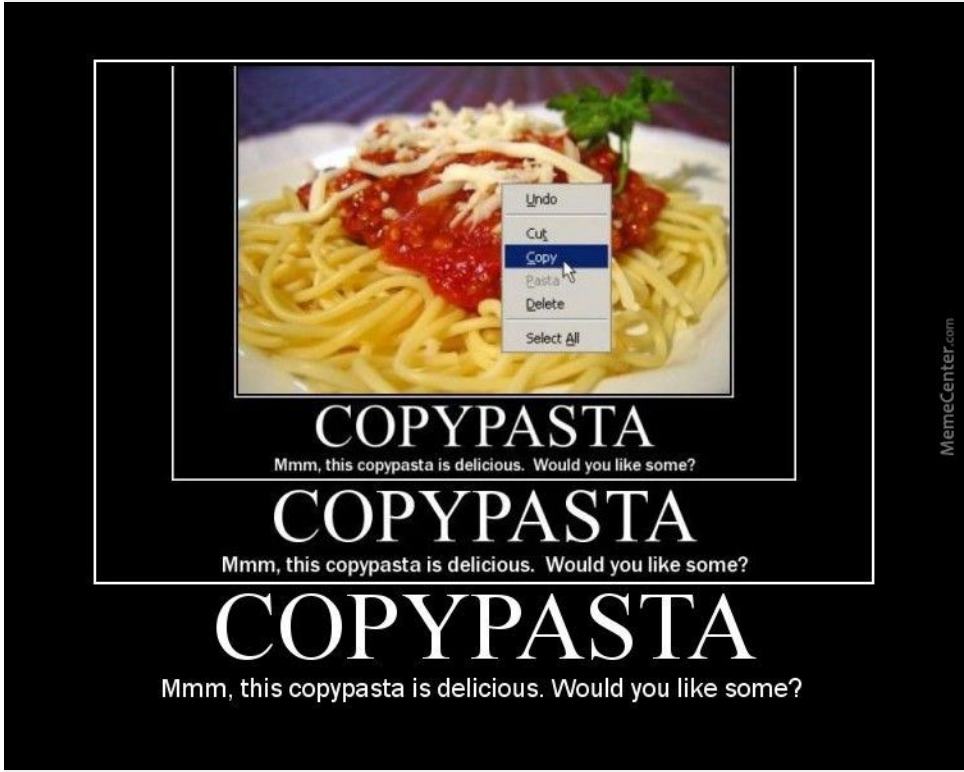
Generic.Forensic. Timeline

Collect timeline from all user home directory.

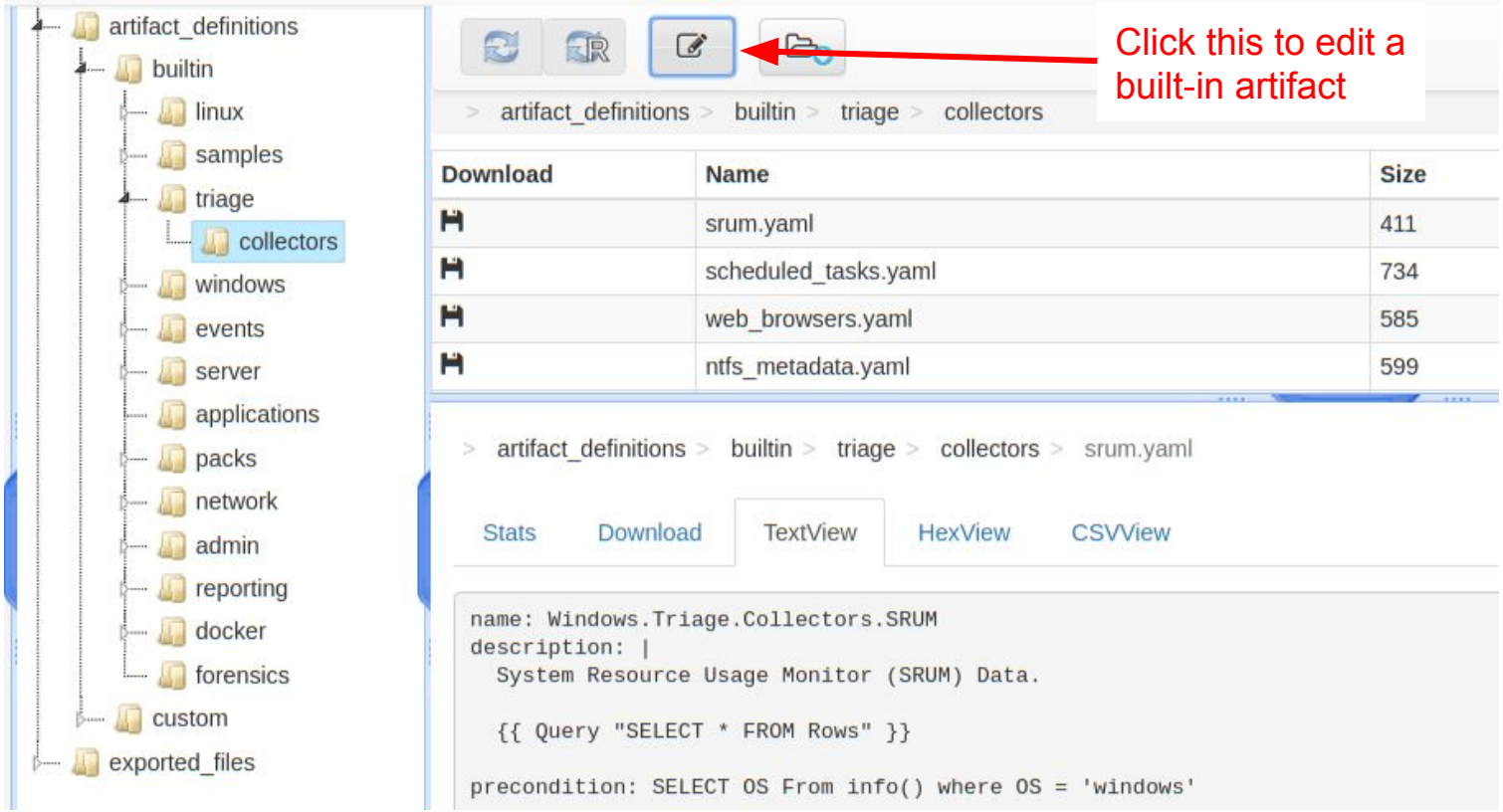
```
name: Generic.Forensic.Timeline
description: |
  This artifact generates a timeline of a file glob in bodyfile
  format. We currently do not calculate the md5 because it is quite
  expensive.

parameters:
  - name: timelineGlob
    default: C:\Users\**
  - name: timelineAccessor
    default: file

sources:
  # For NTFS accessors we write the MFT id as the inode. On windows
  # the file accessor does not give the inode at all.
  - precondition:
      SELECT OS From info() where OS = 'windows' AND timelineAccessor = 'ntfs'
    queries:
      - |
        SELECT 0 AS Md5, FullPath,
               Sys.mft as Inode,
               Mode.String AS Mode, 0 as Uid, 0 as Gid, Size,
               Atime.Sec, Mtime.Sec, Ctime.Sec
        FROM glob(globs=timelineGlob, accessor=timelineAccessor)

  # For linux we can get the Inode from Sys.Ino
  - precondition:
      SELECT * From scope() where timelineAccessor = 'file'
    queries:
      - |
        SELECT 0 AS Md5, FullPath,
               Sys.Ino as Inode,
               Mode.String AS Mode, Sys.Uid AS Uid, Sys.Gid AS Gid, Size,
               Atime.Sec, Mtime.Sec, Ctime.Sec
        FROM glob(globs=timelineGlob, accessor=timelineAccessor)
```

Tweaking existing Artifacts

Copypasta FTW



Find an artifact similar to what you need



Add/Modify an artifact

/artifact_definitions/custom/triage/collectors/srum.yaml

```yaml
name: Windows.Triage.Collectors.SRUM
description: |
  System Resource Usage Monitor (SRUM) Data.

  {{ Query "SELECT * FROM Rows" }}

ndition: SELECT OS From info() where OS = 'windows'

es:
ueries:
  - SELECT * FROM chain(
      a1={ SELECT * FROM Artifact.Triage.Collection.Upload(
           type="SRUM",
           path="C:\\Windows\\System32\\SRU\\**")
      }
    )
```

Save Artifact

New artifact will be written under the custom directory.

Change the artifact name to add a new one. If you do not change the name the custom definition will override the built-in.

Exercise: Collect timeline of recent files

Modify the Generic.Forensic.Timeline artifact to include a last modified time restriction. Only collect timeline of files changed within the last day.

The VQL condition is:

```
WHERE Mtime > now() - 24 * 60 * 60
```

Exercise: Customize triage artifacts

The Windows.Triage.Collectors.* artifacts simply collect relevant files.

- Modify one of the triage artifacts to collect all word documents in a user’s home directory that were created in the last month.

Running VQL interactively - the console

```
F:\>velociraptor.exe console --dump_dir f:\output\
VQL > SET FORMAT text
Setting FORMAT to text
VQL > SET PAGER more
Setting PAGER to more
VQL > SELECT * FROM netstat() LIMIT 1
+----+--------+------+--------------+--------------+--------+-----+--------------+--------------+------------+
| Fd | Family | Type |    Laddr     |    Raddr     | Status | Pid |  Timestamp   | FamilyString | TypeString |
+----+--------+------+--------------+--------------+--------+-----+--------------+--------------+------------+
|  0 |      2 |    1 | {"ip":"0.0.0 | {"ip":"0.0.0 | LISTEN | 976 | 2019-03-21T1 | IPv4         | TCP        |
|    |        |      | .0","port":1 | .0","port":0 |        |     | 1:28:55-07:0 |              |            |
|    |        |      | 35}          | }            |        |     |            0 |              |            |
+----+--------+------+--------------+--------------+--------+-----+--------------+--------------+------------+
SELECT * FROM netstat() LIMIT 1

VQL >
```

Exercise: Detect Att&ck Techniques

https://attack.mitre.org/techniques/T1183/

MITRE | ATT&CK

Matrices Tactics Techniques Groups Software Resources Blog Contact

Search site

Check out the results from our first round of ATT&CK Evaluations at attackevals.mitre.org!

ENTERPRISE

TECHNIQUES

All

Initial Access +

Execution +

Persistence -

.bash_profile and .bashrc

Home > Techniques > Enterprise > Image File Execution Options Injection

Image File Execution Options Injection

Image File Execution Options (IFEO) enable a developer to attach a debugger to an application. When a process is created, a debugger present in an application's IFEO will be prepended to the application's name, effectively launching the new process under the debugger (e.g., "C:\dbg\ntsd.exe -g notepad.exe"). [1]

IFEOs can be set directly via the Registry or in Global Flags via the GFlags tool. [2] IFEOs are represented as `Debugger` values in the Registry under

ID: T1183

Tactic: Privilege Escalation, Persistence, Defense Evasion

Platform: Windows

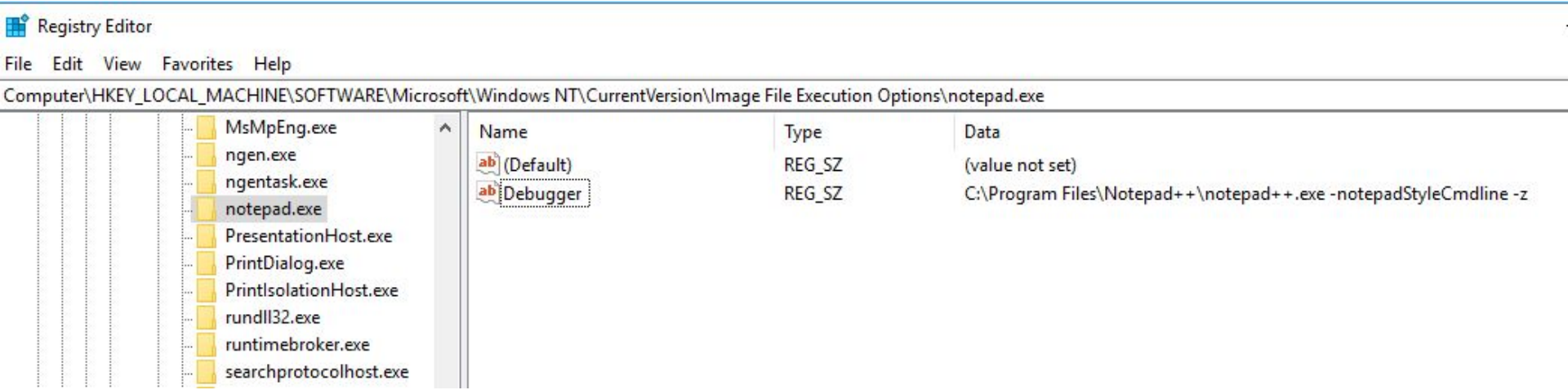
Permissions Required: Administrator, SYSTEM

https://attack.mitre.org/techniques/T1183/

First plant a signal on your machine

```
REG ADD "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File
Execution Options\notepad.exe" /v Debugger /t REG_SZ /d "C:\Program
Files\Notepad++\notepad++.exe -notepadStyleCmdline -z" /f
```

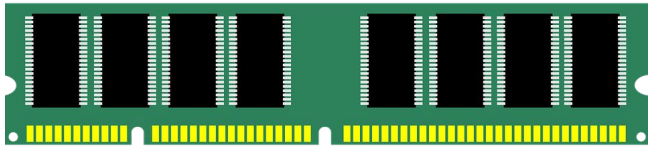
Test this: Type notepad - you get notepad++ (useful but….)



Solution: Windows.Persistence.Debug

```
SELECT Key.Name AS Program,
       Key.FullPath as Key,
       Debugger
FROM read_reg_key(
    globs="HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\*")
WHERE Debugger
```

Exercise: Dumping process memory



Dump process memory when yara sig matches

Advanced malware like the Cobalt Strike Beacon is only memory resident. It is very hard to detect on the network (due to maleable C&C) but it is very easy to detect on the endpoint by scanning the memory of running processes.

We will simulate something similar with notepad:

- Open notepad and write a secret message in it “This is a secret”
- Hunt for the process with the Windows.Detection.ProcessMemory artifact.
- Fetch the crash dump.

Windows.Detection.ProcessMemory

```
- LET processes = SELECT Name as ProcessName, CommandLine, Pid
      FROM pslist()
      WHERE Name =~ processRegex

- LET hits = SELECT * FROM foreach(
     row=processes,
     query={
       SELECT ProcessName, CommandLine, Pid,
              Strings.Offset as Offsets
       FROM proc_yara(rules=yaraRule, pid=Pid)
     })

 - SELECT * FROM foreach(
     row=hits,
     query={
       SELECT ProcessName, CommandLine, Pid, Offsets, FullPath,
              upload(file=FullPath) as CrashDump
       FROM proc_dump(pid=Pid)
     })
```

First find all processes with a name matching the regex

For each of those scan their memory with yara rules.

For each hit, create a process crash dump and then upload the crash dump to the server

Dump F:\output\dmp041786807.dmp - WinDbg:10.0.16299.91 AMD64

File Edit View Debug Window Help

Command

```
*************************************************************************

*** WARNING: Unable to verify checksum for notepad++.exe
*** ERROR: Module load completed but symbols could not be loaded for notepad++.exe
*** ERROR: Symbol file could not be found.  Defaulted to export symbols for KERNELBASE.dll -
GetUrlPageData2 (WinHttp) failed: 12030.

DUMP_CLASS: 2

DUMP_QUALIFIER: 400

FAULTING_IP:
+0
00000000`00000000 ??              ???

EXCEPTION_RECORD:  (.exr -1)
ExceptionAddress: 0000000000000000
   ExceptionCode: 80000003 (Break instruction exception)
  ExceptionFlags: 00000000
NumberParameters: 0

FAULTING_THREAD:  00002b34

BUGCHECK_STR:  BREAKPOINT

DEFAULT_BUCKET_ID:  BREAKPOINT

PROCESS_NAME:  notepad++.exe

ERROR_CODE: (NTSTATUS) 0x80000003 - {EXCEPTION}  Breakpoint  A breakpoint has been reached.

EXCEPTION_CODE: (HRESULT) 0x80000003 (2147483651) - One or more arguments are invalid

EXCEPTION_CODE_STR:  80000003

WATSON_BKT_PROCSTAMP:  5c7f29b4

WATSON_BKT_PROCVER:  7.6.4.0
```

Module 7

Security Auditing

Hunting for anomalies and baselining

Collect installed Chrome Extensions

We want to know what chrome extensions are installed by our user base.

Collect **Windows.Applications.Chrome.Extensions** on your own machine.

This is an example of a fairly complex artifact:

- We need to parse the manifest of extensions to map strings like name, description etc.
- Can you follow the VQL?

Velociraptor

Search Box

Search known artifacts

0 mic

TestComputer
Access reason: test
Status: 27 seconds ago
[::1]:51364

Host Information

Start new flows

Collect Artifacts

Browse Virtual Filesystem

Manage launched flows

MANAGEMENT

Hunt Manager

Server Files

Launch

Search

Windows.Applications.ChocolateyPackage
Windows.Applications.Chrome.Extensions
Windows.Applications.OfficeMacros
Windows.Events.DNSQueries
Windows.Events.FailedLogBeforeSuccess

Selected Artifacts: Add

Windows.Applications.Chrome.Extensions

Selected artifacts

Clear Remove

Artifact description, parameters and VQL sources (might include comments)

Fetch Chrome extensions.

Chrome extensions are installed into the user's home directory. We search for manifest.json files in a known path within each system user's home directory. We then parse the manifest file as JSON.

Many extensions use locale packs to resolve strings like name and description. In this case we detect the default locale and load those locale files. We then resolve the extension's name and description from there.

Parameters

name	extensionGlobs
default	\AppData\Local\Google\Chrome\User Data*\Extensions**\manifest.json

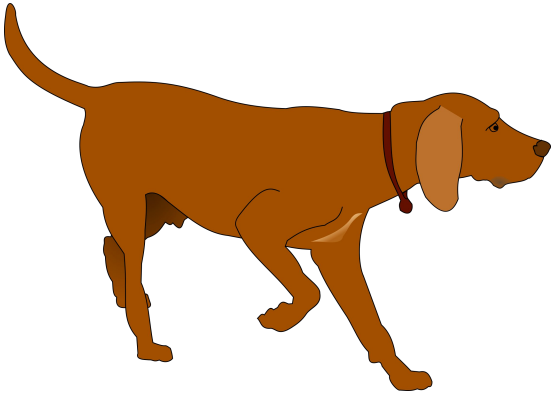
Artifact Sources

Precodition

```
SELECT OS From info() where OS = 'windows'
```

Queries

```
/* For each user on the system, search for extension manifes
   in their home directory. */
LET extension_manifests = SELECT * from foreach(
  row={
     SELECT Uid, Name AS User, Directory from Artifact.Windo
  },
  query={
     SELECT FullPath, Mtime, Ctime, User, Uid from glob(
       globs=Directory + extensionGlobs)
  })
```



Hunting for evil

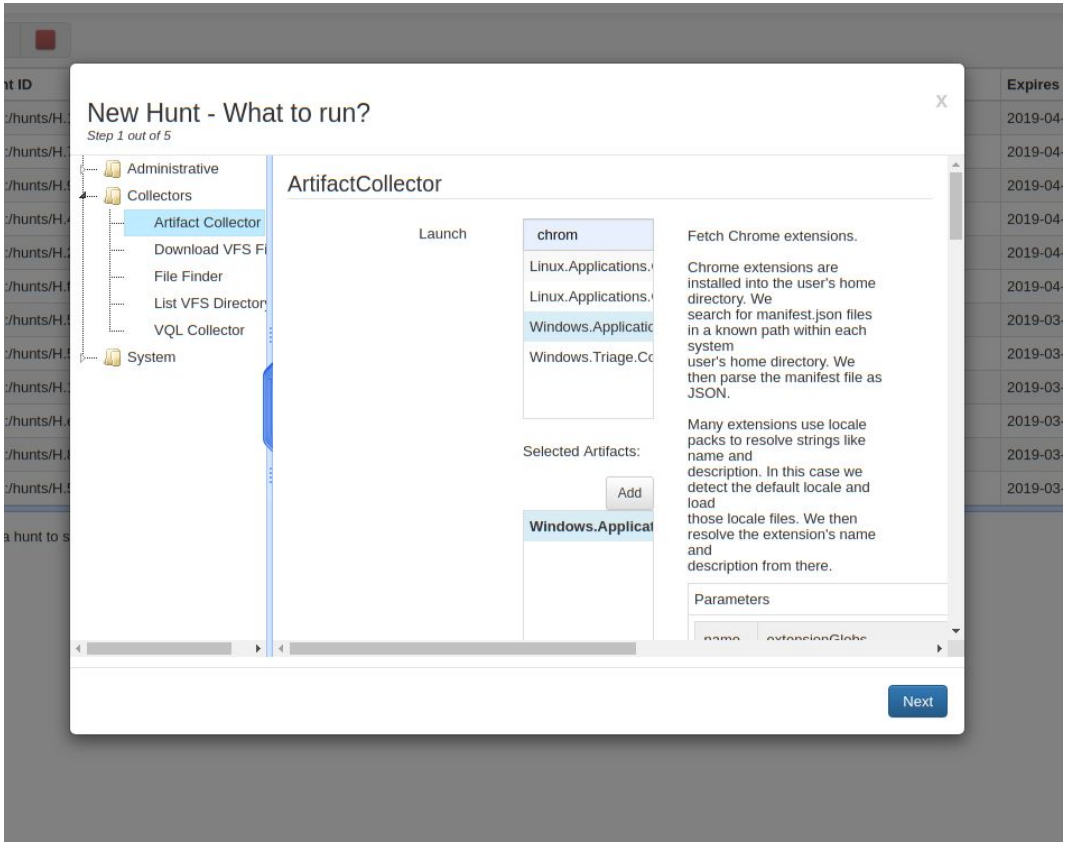
What is a hunt?

- A hunt is just an automated way to collect one or more artifacts across the entire deployment.
- It is just a management abstraction - each endpoint just collects the artifacts in the usual way.
  - The hunt just keeps count of endpoints that participate in the hunt.
  - It is possible to download all results from the hunt as one result set (zip file etc).
  - On the server we can issue VQL to interact with the hunt.
- Hunts are very fast:
  - All currently connected machines are scheduled immediately
  - We typically run a hunt in about 10-20 seconds for currently connected machines.
  - Velociraptor protects itself from too much concurrency - so server load is limited. Feel free to run as many hunts as you need to.

Exercise: Hunt for chrome extensions.

Prepare your hunt through the new hunt wizard:

- Select the artifacts to be collected
- Provide a useful description, the description will be visible in the hunt manager UI
- It is possible to restrict the hunt to a subset of end points:
  - By label
  - By OS
  - By an arbitrary VQL query
- Once the hunt is created we need to start it explicitly.



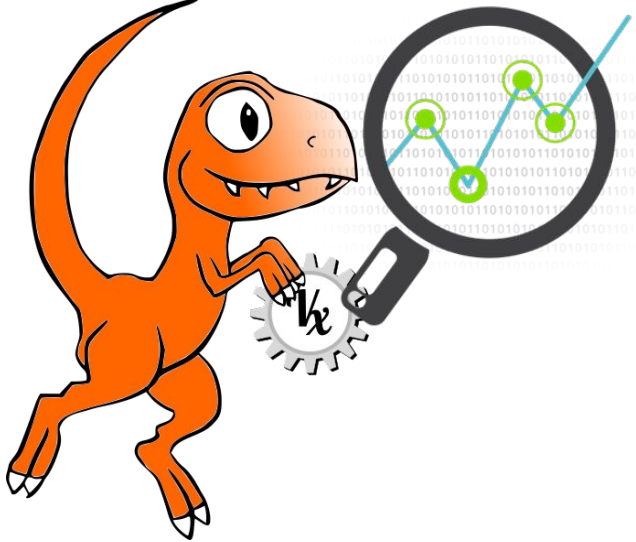
Post processing using VQL

- We can run VQL statements on the server.
  - When run on the server we gain access to additional VQL plugins:
    i. The clients() plugin lists all clients.
    ii. The hunt_flows() plugin lists all flows belonging to a hunt.
    iii. The hunt_results() plugin lists all results in the same hunt.
- Count the most popular chrome extensions in your deployment:

```
SELECT count(items=User) AS TotalUsers,
       Name, Description, Identifier
FROM hunt_results(
       hunt_id=huntId,
       artifact='Windows.Applications.Chrome.Extensions')
ORDER BY TotalUsers DESC
GROUP BY Identifier
```

```
VQL > select count(items=User) AS TotalUsers,Name, Description, Identifier from hunt_results(hunt_id='H
.69e1955e', artifact='Windows.Applications.Chrome.Extensions') ORDER BY TotalUsers DESC GROUP BY Identi
fier
[
 {
  "Description": "Provider for discovery and services for mirroring of Chrome Media Router",
  "Identifier": "pkedcjkdefgpdelpbcmbmeomcjbeemfm",
  "Name": "Chrome Media Router",
  "TotalUsers": 2491
 },
 {
  "Description": "Google Drive: create, share and keep all your stuff in one place.",
  "Identifier": "apdfllckaahabafndbhieahigkjlhalf",
  "Name": "Google Drive",
  "TotalUsers": 2321
 },
 {
  "Description": "Fast, searchable email with less spam.",
  "Identifier": "pjkljhegncpnkpknbcohdijeoejaedia",
  "Name": "Gmail",
  "TotalUsers": 2309
 },
 {
  "Description": "",
  "Identifier": "blpcfgokakmgnkcojhhkbfbldkacnbeo",
  "Name": "YouTube",
  "TotalUsers": 2304
 },
```

Velociraptor Monitoring



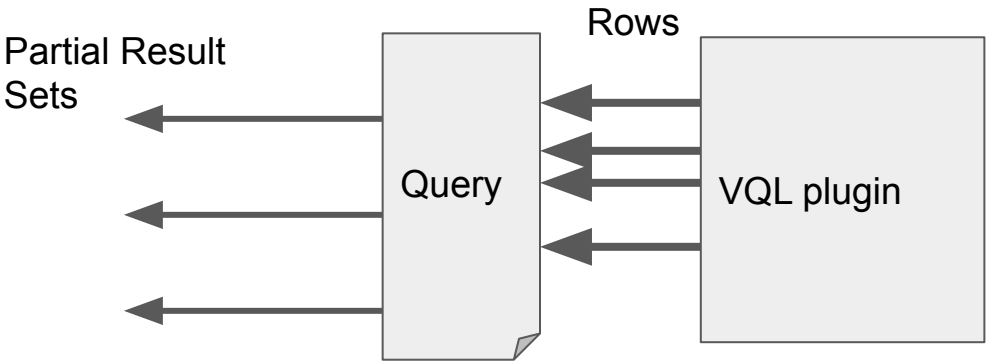
Module 8

VQL: Event Queries

- Normally a VQL query returns a result set and then terminates.
- However some VQL plugins can run indefinitely or for a long time.
  - These are called Event VQL plugins since they can be used to generate events.

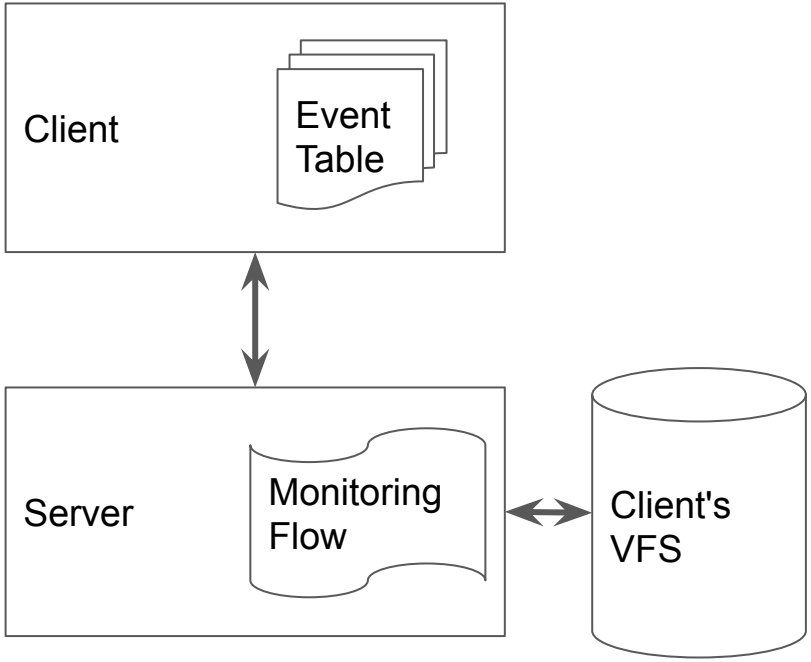
An Event query does not complete on its own - it simply returns partial results until cancelled.

- Wait time
- Max rows



Client monitoring architecture

- The client maintains an Event Table
  - A set of VQL Event Queries
  - All run in parallel.
- When any of the queries in the event table produces a result set, the client sends it to the Monitoring Flow.
- The Server's Monitoring Flow writes the events into log files in the client's VFS.
- The set of events the client should be monitoring is defined as a set of Event Artifacts in the server's config file.
- If the Event Table needs to be refreshed, existing event queries are cancelled and a new event table created.



Example Monitoring configuration

```
Events:
  artifacts:
  - Windows.Events.ServiceCreation
  - Windows.Events.ProcessCreation
  version: 1
```

Simply add new artifact names to the Events section in the config file.

Clients will update their monitoring artifacts when the version number is increased.

Currently monitoring is configured in the configuration file so we need to restart the server to pick up new artifacts.

NOTE: Artifacts are compiled on the server - The client does not need to have these artifact definitions.

Process Execution Logs

- file
- ntfs
- registry
- artifacts
- monitoring
  - Artifact Generic.Cli
  - Artifact Windows.E
  - Artifact Windows.E

> monitoring > Artifact Windows.Events.ProcessCreation

All Event artifacts are collected under the monitoring part of the VFS

Download	Name	Size	Mode	Timestamp
	2019-03-17	1022	-r--r--r--	2019-03-17T23:49:51.414039006+10:00
	2019-03-18	760	-r--r--r--	2019-03-18T00:11:17.166140874+10:00
	2019-03-22	2325	-r--r--r--	2019-03-22T00:44:27.142112709+10:00

> monitoring > Artifact Windows.Events.ProcessCreation > 2019-03-22

Stats | Download | TextView | HexView | CSVView

You can download the CSV file or post process events via server side VQL or the API

Show 10 entries

Timestamp	PPID	PID	Name	Com	
2019-03-21T07:07:47-07:00	856	1232	unsecapp.exe	C:\WINDOWS\system32\wbem\unsecapp.exe -Embedding	C:\WINDOWS\system32\svchost.exe -k DcomLaunch -p
2019-03-21T07:13:34-07:00	872	3940	SearchProtocolHost.exe	"C:\WINDOWS\system32\SearchProtocolHost.exe" Global\UsGthrFltPipeMssGthrPipe40_ Global\UsGthrCtrlFltPipeMssGthrPipe40 1 -2147483646 "Software\Microsoft\Windows Search" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT; MS Search 4.0 Robot)" "C:\ProgramData\Microsoft\Search\Data\Temp\usgthrsvc" "DownLevelDaemon"	C:\WINDOWS\system32\SearchIndexer.exe /Embedding

2019-03-21 14:51:03 UT

Example: Log DNS queries on the endpoint

- file
- ntfs
- registry
- artifacts
- monitoring
  - Artifact Generic.Client.Stats
  - Artifact System.Hunt.Participation
  - Artifact Windows.Events.DNSQueries
  - Artifact Windows.Events.FailedLogBefo
  - Artifact Windows.Events.ProcessCreati
  - Artifact Windows.Events.ServiceCreatio
  - Artifact Windows.System.FailedLoginA

monitoring > Artifact Windows.Events.DNSQueries

Download	Name	Size	Mode	Timestamp
	2019-02-16	24833	-r--r--r--	2019-02-16T23:56:19.266107507+10:00
	2019-02-17	5284	-r--r--r--	

monitoring > Artifact Windows.Events.DNSQueries > 2019-02-16

Stats | Download | TextView | HexView | CSVView

Show 10 entries

Time	EventType	Name	CNAME	Answers
2019-02-16T01:10:36-08:00	A	licensing.mp.microsoft.com.	["licensing.md.mp.microsoft.com.akadns.net.", "licensing-asia.md.mp.microsoft.com.akadns.net.", "hk2.licensing.md.mp.microsoft.com.akadns.net."]	["111.221.29.174"]
2019-02-16T01:10:37-08:00	A	activation-v2.sls.microsoft.com.	["activation-v2.sls.trafficmanager.net."]	["65.52.98.233"]
2019-02-16T01:11:22-08:00	A	tile-service.weather.microsoft.com.	["wildcard.weather.microsoft.com.edgekey.net.", "e15275.g.akamaiedge.net."]	["104.67.248.74"]
2019-02-16T01:11:23-	A	cdn.onenote.net.	["cdn.onenote.net.edgekey.net.", "e1553.dspg.akamaiedge.net."]	["23.212.219.133"]

Historical record of IP/DNS mapping. Note: This is recorded on the end point so works even at Starbucks! Good for fast flux domains.

Exercise - Generic.Client.Statistics

Our users are concerned about the potential resource usage of the Velociraptor client.

The clock plugin generates an event periodically (every 10 sec)

```
SELECT * from foreach(
        row={
          SELECT UnixNano FROM clock(period=atoi(string=Frequency))
        },
        query={
          SELECT UnixNano / 1000000000 as Timestamp,
                 Times.user + Times.system as CPU,
                 MemoryInfo.RSS as RSS
          FROM pslist(pid=getpid())
        })
```

At each clock event we run this query and emit its results to the server event stream.

Example: Windows.Events.ServiceCreation.

```
SELECT System.TimeCreated.SystemTime as Timestamp,
       System.EventID.Value as EventID,
       EventData.ImagePath as ImagePath,
       EventData.ServiceName as ServiceName,
       EventData.ServiceType as Type,
       System.Security.UserID as UserSID,
       EventData as _EventData,
       System as _System
  FROM watch_evtx(filename=systemLogFile) WHERE EventID = 7045
```

Event ID 7045: A service was installed in the system

watch_evtx() VQL plugin can watch an event log for new events which it then emits as rows.

Event Viewer

File Action View Help

Event Viewer (Local)
- Custom Views
- Windows Logs
  - Application
  - Security
  - Setup
  - System
  - Forwarded Events
- Applications and Service
- Subscriptions

System Number of events: 41

Level	Date and Time	Source	Event ID	Tas
Information	11/5/2018 9:49:38 PM	Service...	7040	No
Information	11/5/2018 4:40:11 PM	Service...	7045	No
Error	11/5/2018 3:09:54 PM	Windo...	20	Wi
Information	11/5/2018 2:58:17 PM	Service...	7040	No
Information	11/5/2018 2:55:11 PM	Windo...	43	Wi

Event 7045, Service Control Manager

General Details

A service was installed in the system.

Service Name: Velociraptor
Service File Name: "C:\Program Files\Velociraptor\Velociraptor.exe" se
Service Type: user mode service
Service Start Type: auto start
Service Account: LocalSystem

Log Name:	System		
Source:	Service Control Manager	Logged:	11/5/
Event ID:	7045	Task Category:	None
Level:	Information	Keywords:	Class
User:	TESTCOMPUTER\test	Computer:	TestC
OpCode:	Info		
More Information:	Event Log Online Help		

Actions

System
- Open Saved Log...
- Create Custom View...
- Import Custom View...
- Clear Log...
- Filter Current Log...
- Properties
- Find...
- Save All Events As...
- Attach a Task To this L...
- View
- Refresh
- Help

Event 7045, Service Control...
- Event Properties
- Attach Task To This Ev...
- Copy
- Save Selected Events...
- Refresh
- Help

Let’s detect service installation

Administrator: Command Prompt - velociraptor.exe console --dump_dir f:\output\

```
]VQL
VQL > SELECT * FROM watch_evtx(filename="c:/windows/system32/winevt/logs/system.evtx") WHERE System.EventId.Value = 7045
[
 {
  "EventData": {
   "AccountName": "LocalSystem",
   "ImagePath": "\"C:\\Program Files\\Velociraptor\\Velociraptor.exe\" service run",
   "ServiceName": "Velociraptor",
   "ServiceType": "user mode service",
   "StartType": "auto start"
  },
  "System": {
   "Channel": "System",
   "Computer": "TestComputer",
   "Correlation": {},
   "EventID": {
    "Qualifiers": 16384,
    "Value": 7045
   },
   "EventRecordID": 131699,
   "Execution": {
    "ProcessID": 684,
    "ThreadID": 5056
   },
   "Keywords": 9259400833873739776,
   "Level": 4,
   "Opcode": 0,
   "Provider": {
    "EventSourceName": "Service Control Manager",
    "Guid": "{555908d1-a6d7-4695-8e1e-26931d2012f4}",
    "Name": "Service Control Manager"
   },
   "Security": {
    "UserID": "S-1-5-21-546003962-2713609280-610790815-100
   },
   "Task": 0,
   "TimeCreated": {
    "SystemTime": 1553142697.0653296
```

Administrator: Command Prompt

```
Microsoft Windows [Version 10.0.17134.648]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>f:

F:\>velociraptor.exe --config velo.config.yaml service install

F:\>velociraptor.exe --config velo.config.yaml service remove

F:\>
```

Type here to search

Watch the system event log file for new events with ID 7045 (service creation). Which fields are of interest?

In another terminal install and remove the velociraptor service

Let’s go back to psexec: Service creation

- PsExec works by copying itself to an admin share then creating a service remotely to start it.
- Test this with the previous artifact - you should see a new service created:

```
PsExec.exe -s -i cmd.exe
```

- But we can change the name of the created service using the -r flag.

```
PsExec.exe -s -r svchost -i cmd.exe
```

```
F:\>velociraptor.exe artifacts collect Windows.Events.ServiceCreation --format json
[][][
 {
  "EventID": 7045,
  "ImagePath": "%SystemRoot%\\svchost.exe",
  "ServiceName": "svchost",
  "Timestamp": 1553435192.2840033,
  "Type": "user mode service",
  "UserSID": "S-1-5-21-546003962-2713609280-610
  "_EventData": {
   "AccountName": "LocalSystem",
   "ImagePath": "%SystemRoot%\\svchost.exe",
   "ServiceName": "svchost",
   "ServiceType": "user mode service",
   "StartType": "demand start"
  },
  "_System": {
   "Channel": "System",
   "Computer": "TestComputer",
   "Correlation": {},
   "EventID": {
    "Qualifiers": 16384,
    "Value": 7045
   },
   "EventRecordID": 132150,
```

Administrator: C:\WINDOWS\system32\cmd.exe

```
C:\WINDOWS\system32>whoami
nt authority\system

C:\WINDOWS\system32>
```

\\TESTCOMPUTER: cmd.exe

```
Microsoft Windows [Version 10.0.17134.648]
(c) 2018 Microsoft Corporation. All rights reserved

C:\WINDOWS\system32>f:

F:\>bin\PsExec.exe -s -r svchost -i cmd.exe

PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com
```

Exercise: Detect psexec with renamed service.

We need to modify the Windows.Events.ServiceCreation artifact to detect psexec with renamed service name by yara scanning the service file but this has a race!

- Windows.Events.ServiceCreation watches the event log file. Windows Event logs are flushed lazily (~10 seconds or more). If a psexec process is terminated before the event hits the log file we will be unable to find the file 😞.
- We therefore need to use some more efficient mechanism to be notified of a service creation event - WMI. Still not perfect but better….
- Try this by closing the psexec window very quickly or running a very quick command like PsExec.exe -s -r svchost -i cmd.exe /c dir c:\

Exercise: Windows.Detection.PsexecService

```
- LET file_scan = SELECT File, Rule, Strings, now() AS Timestamp,
        Name, ServiceType
  FROM yara(
        rules=yaraRule,
        accessor="ntfs",
        files=PathName)
  WHERE Rule

- LET service_creation = SELECT Parse.TargetInstance.Name AS Name,
        Parse.TargetInstance.PathName As PathName,
        Parse.TargetInstance.ServiceType As ServiceType
  FROM wmi_events(
     query="SELECT * FROM __InstanceCreationEvent WITHIN 1 WHERE TargetInstance ISA 'Win32_Service'",
     wait=5000000,
     namespace="ROOT/CIMV2")

- SELECT * FROM foreach(
    row=service_creation,
    query=file_scan)
```

Register a WMI event for creation of new service objects. The WITHIN 1 reduces the race condition to 1 second.

The diff() plugin

The diff plugin is an event plugin which runs a non-event query periodically and reports the difference between each execution.

- Start with a simple query: Get all files in the user’s temp directory

```
SELECT FullPath FROM glob(globs='c:/Users/*/AppData/Local/Temp/**')
```

- Now diff it every 10 seconds

```
][
 {
  "FullPath": "\\C:\\Us
 },
 {
  "FullPath": "\\C:\\Us
 },
 {
  "FullPath": "\\C:\\Us
 }
]VQL > SELECT * FROM diff(query={SELECT FullPath FROM glob(globs='c:/Users/*/AppData/Local/Temp/**')
[
 {
  "Diff": "added",
  "FullPath": "\\C:\\Users\\test\\AppData\\Local\\Temp\\my_file.txt"
 }
]
```

Administrator: Command Prompt

```
F:\>
F:\>notepad %TEMP%\my_file.txt

F:\>
F:\>
```

C:\Users\test\AppData\Local\Temp\my_file.txt - Notepad++ [Administ

File Edit Search View Encoding Language Settings Tools M

my_file.txt | my_file2.txt

```
1  hello world
```

Normal te length : 11 lines : 1 Ln : 1 Col : 12 Sel : 0 | 0

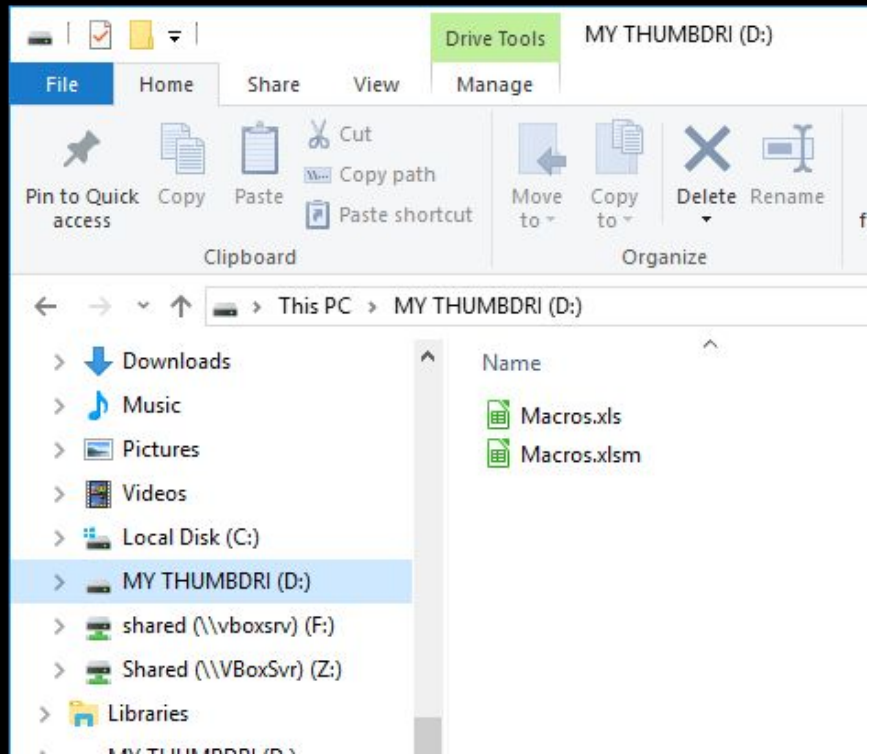
Example: Monitor insertion of USB thumb drives

Windows.Detection.Thumbdrives.List

```
LET removable_disks = SELECT Name AS Drive, atoi(string=Data.Size) AS Size
FROM glob(globs="/*", accessor="file")
WHERE Data.Description =~ "Removable" AND Size < maxDriveSize

LET file_listing = SELECT FullPath, timestamp(epoch=Mtime.Sec) As Modified, Size
FROM glob(globs=Drive+"\\**", accessor="file") LIMIT 1000

SELECT * FROM diff(
  query={ SELECT * FROM foreach(row=removable_disks, query=file_listing) },
  key="FullPath",
  period=10)
  WHERE Diff = "added"
```

Diff the file listing every 10 seconds and record added files.

```
F:\>velociraptor.exe artifacts collect -v --format json Windows.Detection.Thumbdrives.List
[INFO] 2019-03-25T19:09:30-07:00 Loaded 98 built in artifacts
[][][][
 {
  "Diff": "added",
  "FullPath": "\\D:\\Macros.xls",
  "Modified": "2019-03-25T18:23:26-07:00",
  "Size": 1205248
 },
 {
  "Diff": "added",
  "FullPath": "\\D:\\Macros.xlsm",
  "Modified": "2019-03-25T18:22:46-07:00",
  "Size": 3561814
 },
 {
  "Diff": "added",
  "FullPath": "\\D:\\System Volume Information",
  "Modified": "2019-03-25T17:27:26-07:00",
  "Size": 0
```



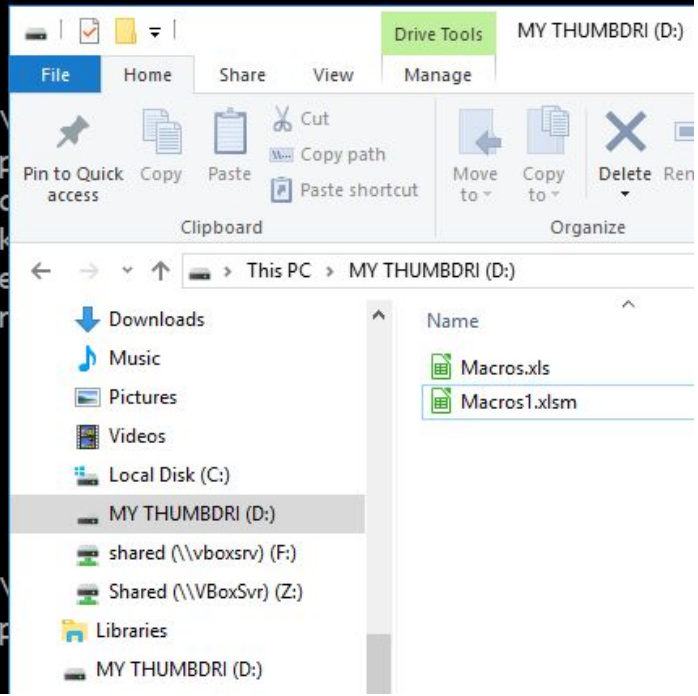
Exercise: Scan USB drives for Office Macros

Windows.Detection.Thumbdrives.OfficeMacros

```
SELECT * FROM foreach(
  row = {
    SELECT * FROM Artifact.Windows.Detection.Thumbdrives.List()
    WHERE FullPath =~ officeExtensions
  },
  query = {
    SELECT * from olevba(file=FullPath)
  })
```

We can just use the previous artifact directly.

```
F:\>velociraptor.exe artifacts collect -v --format json Windows.Detection.Thumbdrives.OfficeMacros
[INFO] 2019-03-25T19:18:29-07:00 Loaded 98 built in artifacts
[][
 {
  "Code": "Attribute VB_Name = \"Blad01\"\r\nAttribute VB_Base = \
e = False\r\nAttribute VB_PredeclaredId = True\r\nAttribute VB_Exp
lm; www.fhvzelm.com\r\n'Inhoud module Blad01\r\n'- Worksheet_Selec
\r\n'Macro seleceert 'niet-hyperlinkbare' bladen als op cel geklik
oTo Foutafhandeling\r\n  If Target = Range(\"C7\") Then\r\n    She
get = Range(\"C37\") Then\r\n    Sheets(\"xl4Macro\").Activate\r\n
  "ModuleName": "Blad01",
  "StreamName": "Blad01",
  "Type": "cls",
  "filename": "\\D:\\Macros1.xlsm"
 },
 {
  "Code": "Attribute VB_Name = \"Blad04\"\r\nAttribute VB_Base = \
e = False\r\nAttribute VB_PredeclaredId = True\r\nAttribute VB_Exp
r\n\r\n",
```



Exercise: Scanning Office Docs for keywords

Windows.Detection.Thumbdrives.OfficeKeywords

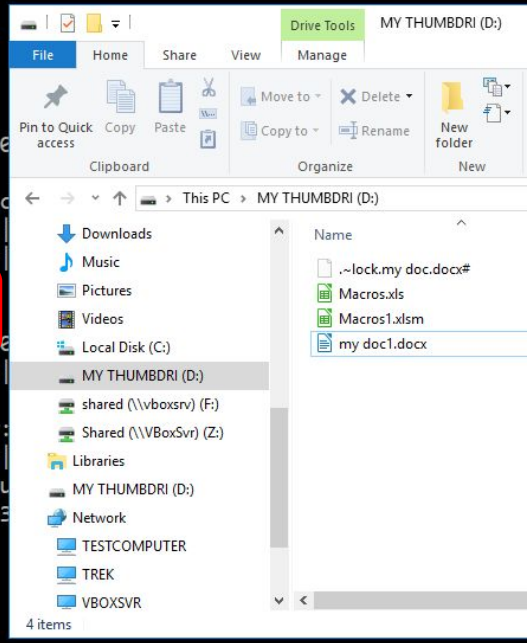
```
SELECT * FROM foreach(
  row = {
    SELECT * FROM Artifact.Windows.Detection.Thumbdrives.List()
    WHERE FullPath =~ officeExtensions
  },
  query = {
    SELECT * FROM Artifact.Generic.Applications.Office.Keywords(
    yaraRule=yaraRule, searchGlob=FullPath, documentGlobs="")
  })
```

Use this artifact to get events

Collect this artifact for each event. We can also provide parameters to the artifact.

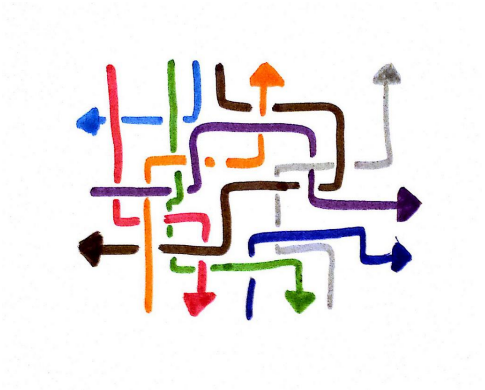
Artifact reuse FTW!

```
F:\>velociraptor.exe artifacts collect -v --format json Windows.Detection.Thumbdrives.OfficeKeywords
[INFO] 2019-03-25T19:30:57-07:00 Loaded 99 built in artifacts
[][
 {
  "HexContext": [
   [
    "00000000  6f 63 65 73 73 69 6e 67  44 72 61 77 69 6e 67 22  |ocessingDrawing\"|",
    "00000010  20 78 6d 6c 6e 73 3a 77  31 34 3d 22 68 74 74 70  | xmlns:w14=\"http|",
    "00000020  3a 2f 2f 73 63 68 65 6d  61 73 2e 6d 69 63 72 6f  |://schemas.micro|",
    "00000030  73 6f 66 74 2e 63 6f 6d  2f 6f 66 66 69 63 65 2f  |soft.com/office/|",
    "00000040  77 6f 72 64 2f 32 30 31  30 2f 77 6f 72 64 6d 6c  |word/2010/wordml|",
    "00000050  22 20 6d 63 3a 49 67 6e  6f 72 61 62 6c 65 3d 22  |\" mc:Ignorable=\"|",
    "00000060  77 31 34 20 77 70 31 34  22 3e 3c 77 3a 62 6f 64  |w14 wp14\"\u003e\u003cw:bod|",
    "00000070  79 3e 3c 77 3a 70 3e 3c  77 3a 70 50 72 3e 3c 77  |y\u003e\u003cw:p\u003e\u003cw:pPr\u00",
    "00000080  3a 70 53 74 79 6c 65 20  77 3a 76 61 6c 3d 22 4e  |:pStyle w:val=\"N|",
    "00000090  6f 72 6d 61 6c 22 2f 3e  3c 77 3a 72 50 72 3e 3c  |ormal\"/\u003e\u003cw:rPr\u003e\u003c",
    "000000a0  2f 77 3a 72 50 72 3e 3c  2f 77 3a 70 50 72 3e 3c  |/w:rPr\u003e\u003c/w:pPr\u003e\u003c|",
    "000000b0  77 3a 72 3e 3c 77 3a 72  50 72 3e 3c 2f 77 3a 72  |w:r\u003e\u003cw:rPr\u003e\u003c/w:r|",
    "000000c0  50 72 3e 3c 77 3a 74 3e  54 68 69 73 20 69 73 20  |Pr\u003e\u003cw:t\u003eThis is |",
    "000000d0  6d 79 20 73 65 63 72 65  74 e2 80 a6 2e 3c 2f 77  |my secret....\u003c/w|",
    "000000e0  3a 74 3e 3c 2f 77 3a 72  3e 3c 2f 77 3a 70 3e 3c  |:t\u003e\u003c/w:r\u003e\u003c/w:p\u0",
    "000000f0  77 3a 70 3e 3c 77 3a 70  50 72 3e 3c 77 3a 70 53  |w:p\u003e\u003cw:pPr\u003e\u003cw:pS|",
    "00000100  74 79 6c 65 20 77 3a 76  61 6c 3d 22 4e 6f 72 6d  |tyle w:val=\"Norm|",
    "00000110  61 6c 22 2f 3e 3c 77 3a  72 50 72 3e 3c 2f 77 3a  |al\"/\u003e\u003cw:rPr\u003e\u003c/w:",
    "00000120  72 50 72 3e 3c 2f 77 3a  70 50 72 3e 3c 77 3a 72  |rPr\u003e\u003c/w:pPr\u003e\u003cw:r|",
    "00000130  3e 3c 77 3a 72 50 72 3e  3c 2f 77 3a 72 50 72 3e  |\u003e\u003cw:rPr\u003e\u003c/w:rPr\u",
    "00000140  3c 2f 77 3a 72 3e 3c 2f  77 3a 70 3e 3c 77 3a 73  |\u003c/w:r\u003e\u003c/w:p\u003e\u003",
    "00000150  65 63 74 50 72 3e 3c 77  3a 74 79 70 65 20 77 3a  |ectPr\u003e\u003cw:type w:|",
    "00000160  76 61 6c 3d 22 6e 65 78  74 50 61 67 65 22 2f 3e  |val=\"nextPage\"/\u003e|",
    "00000170  3c 77 3a 70 67 53 7a 20  77 3a 77 3d 22 31 32 32  |\u003cw:pgSz w:w=\"122|",
    "00000180  34 30 22 20 77 3a 68 3d  22 31 35 38 34 30 22 2f  |40\" w:h=\"15840\"/|",
    "00000190  3e 3c 77 3a 70 67 4d 61  72 20 77 3a 6c 65 66 74  |\u003e\u003cw:pgMar w:left|",
    "000001a0  3d                                                |=|",
    ""
```



Module 9

Server VQL and the Velociraptor API



VQL can be run on the server!

```
VQL > select count(items=client_id), agent_information.version AS version from clients() where last_seen_at / 1000000 > now() - 60*60*24*30  group by version
+-----------------------+---------------------------+
| count(items=client_id) |          version          |
+-----------------------+---------------------------+
|                  1165 | 2019-02-21T21:34:46+10:00 |
+-----------------------+---------------------------+
SELECT count(items=client_id),
agent_information.version AS version FROM clients()
WHERE last_seen_at / 1000000 > now - 60 * 60 * 24 * 30
GROUP BY version
VQL > select count(items=client_id), agent_information.version AS version from clients() where last_seen_at / 1000000 > now() - 60*60*24*7  group by version
+-----------------------+---------------------------+
| count(items=client_id) |          version          |
+-----------------------+---------------------------+
|                  1082 | 2019-02-21T21:34:46+10:00 |
+-----------------------+---------------------------+
SELECT count(items=client_id),
agent_information.version AS version FROM clients()
WHERE last_seen_at / 1000000 > now - 60 * 60 * 24 * 7
GROUP BY version
VQL > select count(items=client_id), agent_information.version AS version from clients() where last_seen_at / 1000000 > now() - 60*60*24  group by version
+-----------------------+---------------------------+
| count(items=client_id) |          version          |
+-----------------------+---------------------------+
|                   846 | 2019-02-21T21:34:46+10:00 |
+-----------------------+---------------------------+
SELECT count(items=client_id),
```

30 Day active client count grouped by version

1 Day active client count

The Velociraptor API

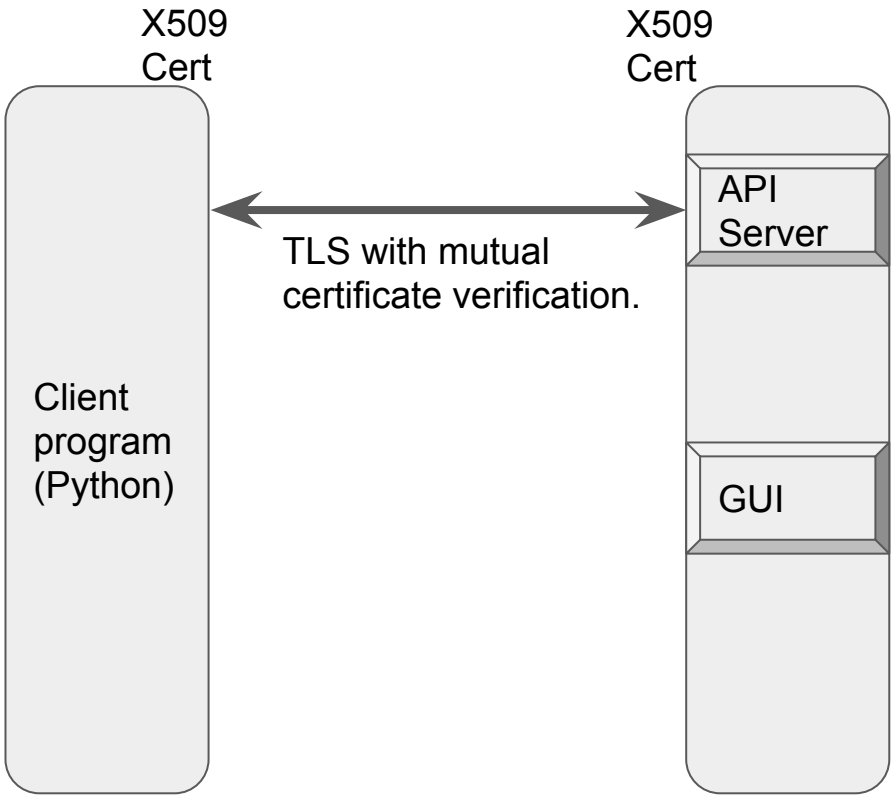
The API is extremely powerful!

Needs to be protected!

The point of an API is to allow a client program (written in any language) to interact with Velociraptor.

The server mints a certificate for the client program to use. This allows it to authenticate and establish a TLS connection with the API server.

By default the API server only listens on 127.0.0.1 - you need to reconfigure it to open it up.



Create a client API certificate

velociraptor.exe --config velo.config.yaml config api_client > api_client.config.yaml

F:\api_client.config.yaml - Notepad++ [Administrator]

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

api_client.config.yaml

```yaml
ca_certificate: |
client_cert: |
  -----BEGIN CERTIFICATE-----
  MIIDDTCCAfWgAwIBAgIBAjANBgkqhkiG9w0BAQsFADAaMRgwFgYDVQQKEw9WZWxv
  Y2lyYXB0b3IgQ0EwHhcNMTkwMzIxMTUxMzA0WhcNMjAwMzIwMTUxMzA0WjA5MRUw
  EwYDVQQKEwxWZWxvY2lyYXB0b3IxIDAeBgNVBAMTF1ZlbG9jaXJhcHRvciBBUEkg
  Q2xpZW50MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAuCjgaIzmeAZ4
  zf22YbyHd1u+9ncTDVP0x9V/BNU2ChXJixZr+bMQ4Kasr6BmlLiedEA9HATsH2gx
  f6/b51iNc5YUKLpVGgggXZSVBxWT2KFM6g+J6gq3Z/VOoY3F6FURjCmigBv8ffai
  8J7aEN+pVOueyDyPEU60+35FVWZ8Q3crnUmpEm5RHoV8lC/dt2HiP38soebpCFli
  eeq0LFdrOxwVxkDdWm7Jxl3iYQYAA/63FC/aeUeNOGSyjk+XklBlosMqJ6HXExq/
  FcCwzG+RIU0OqbYBOuj5YoK49XZArLEyDr6UXjeShd7vj2aWdJN+nwgK6RL97S8I
  T4AGcGg0lQIDAQABoz8wPTAOBgNVHQ8BAf8EBAMCBaAwHQYDVR0lBBYwFAYIKwYB
  BQUHAwEGCCsGAQUFBwMCMAwGA1UdEwEB/wQCMAAwDQYJKoZIhvcNAQELBQADggEB
  AAI6DwIKog3r6V/Ga+r3H6WnahnC9v93wRBImESTBb3XppgzEu0tlY09k+r5Yu9Y
  GUl2+qwq7Ag7zuWaR87h6Tw+uAnPz6X68yeSNspsJ1AiLi3BWCx1or27WEqX9PWC
  ++zretimplrYcTKE2JcH8cGLk/WWabRp/Nmfi4oKHYEhiDanYoSsmwjlwNvahgKB
  KUj+icL7zikPMYWX7iC/SNTbU1mm7T5C2mMRDlsvIIjmoI4se3HxVD9FuEu1EoXy
  enKB/ElV3vbojZ5gcbJEUMw6DqCaJnu60TQVMsDBXCWK0vnvtTy6FQpZpxHNKE0Q
  Sxuuj14/rDHzqAOj/Cvi9y8=
  -----END CERTIFICATE-----
client_private_key: |
api_connection_string: 127.0.0.1:8001
name: Velociraptor API Client
```

The API simply allows VQL to run on the server

```
creds = grpc.ssl_channel_credentials(
    root_certificates=config["ca_certificate"].encode("utf8"),
    private_key=config["client_private_key"].encode("utf8"),
    certificate_chain=config["client_cert"].encode("utf8"))

options = (('grpc.ssl_target_name_override', "VelociraptorServer",),)
with grpc.secure_channel(config["api_connection_string"], creds, options) as channel:
    stub = api_pb2_grpc.APIStub(channel)
    request = api_pb2.VQLCollectorArgs(
        max_wait=1,
        Query=[api_pb2.VQLRequest(
            VQL=" SELECT * from clients() ",
        )])

    for response in stub.Query(request):
        package = json.loads(response.Response)
        print (package)
```

Example of API program - fuse.

1. Download and install WinFSP - the fuse implementation for windows.
2. Start your client on another terminal - note its client ID. Make sure it is properly communicating with the frontend.
3. Start the fuse feature using the api_client.yaml and the client id
   a. Use q: as a drive letter to mount the client's virtual filesystem.

```
velociraptor.exe --api_config api_client.config.yaml -v fuse q: C.11a3013cca8f826e
```

API config we generated earlier.

Mount point (Drive)

Client ID

%5C%5C.%5CC%3A

File | Home | Share | View

Pin to Quick access | Copy | Paste | Cut | Copy path | Paste shortcut | Move to | Copy to | Delete | Rename | New folder | New item | Easy access | Properties | Open | Edit | History | Select all | Select none | Invert selection

Clipboard | Organize | New | Open | Select

This PC › Local Disk (Q:) › ntfs › %5C%5C.%5CC%3A

Search %5C%5C.%5CC%3A

- shared (\\vboxsrv) (F:)
- Local Disk (Q:)
  - artifacts
  - file
  - monitoring
  - ntfs
    - %5C%5C.%5CC%3A
      - $Extend
      - $Recycle.Bin
      - $Secure%3A$SDH
      - $Secure%3A$SII
      - DOCUME~1
      - Documents and Settings
      - gopath
      - MinGW
      - PerfLogs
      - PROGRA~1
      - PROGRA~2
      - PROGRA~3
      - Program Files
      - Program Files %28x86%29

Name	Date modified	Type	Size
Python27	11/1/2017 11:39 AM	File folder	
Recovery	11/9/2018 3:43 PM	File folder	
scripts	11/7/2018 6:03 AM	File folder	
System Volume Information	10/4/2017 8:17 AM	File folder	
SYSTEM~1	10/4/2017 8:17 AM	File folder	
tmp	11/14/2018 4:21 PM	File folder	
Users	4/11/2018 4:44 PM	File folder	
Windows	11/9/2018 6:11 PM	File folder	
$AttrDef	10/4/2017 9:10 AM	File	3 KB
$BadClus	10/4/2017 9:10 AM	File	0 KB
$BadClus%3A$Bad	10/4/2017 9:10 AM	File	32,562,532 ...
$Bitmap	10/4/2017 9:10 AM	File	994 KB
$Boot	10/4/2017 9:10 AM	File	8 KB
$LogFile	10/4/2017 9:10 AM	File	44,896 KB
$MFT	10/4/2017 9:10 AM	File	402,432 KB
$MFTMirr	10/4/2017 9:10 AM	File	4 KB
$Secure%3A$SDS	10/4/2017 9:10 AM	File	3,485 KB
$UpCase	10/4/2017 9:10 AM	File	128 KB
$UpCase%3A$Info	10/4/2017 9:10 AM	File	1 KB
$Volume	10/4/2017 9:10 AM	File	0 KB
pagefile.sys	2/13/2019 11:17 PM	System file	1,310,720 KB
Python27%3AWin32App_1	11/1/2017 11:39 AM	File	0 KB
swapfile.sys	11/9/2018 10:29 PM	System file	262,144 KB

38 items | 1 item selected

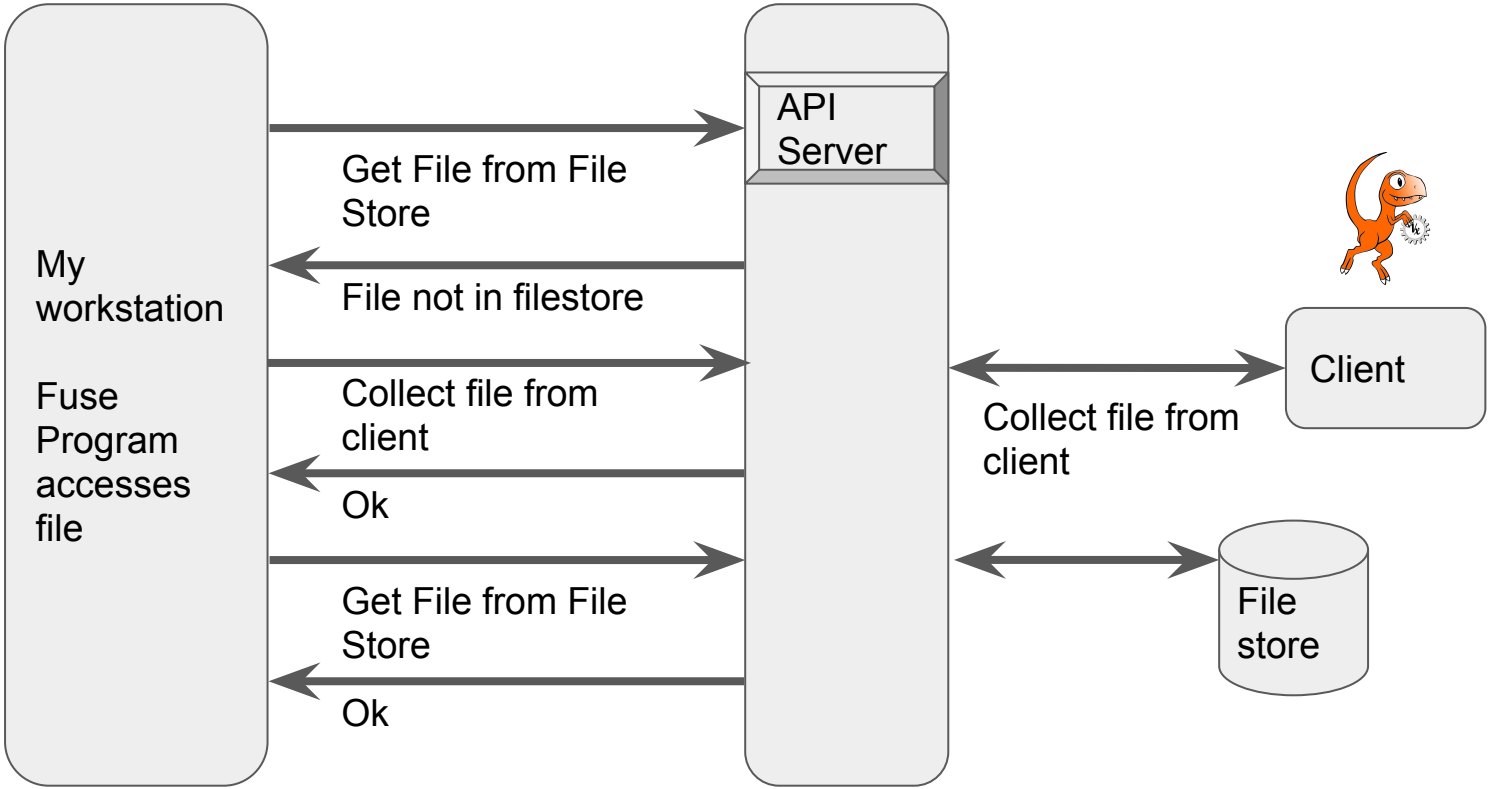
How does it work?

- When a file is accessed on **q:** drive, we make an API call to schedule a new file upload collection on the client
  - This is equivalent to the GUI's "Download this file" feature.
- When the file is received it can be passed to the fuse layer.
- When a directory is accessed on **q:** drive, we make an API call to list the directory from the client.
  - This is equivalent to the "refresh directory" in the GUI

Overall effect is that it feels like we are navigating the endpoint's filesystem directly! Almost as if it is mounted.

However: All accesses to the endpoint are logged and audited on the server!

The entire process is managed by the API client (Fuse program)



Using third party tools on the fuse mount

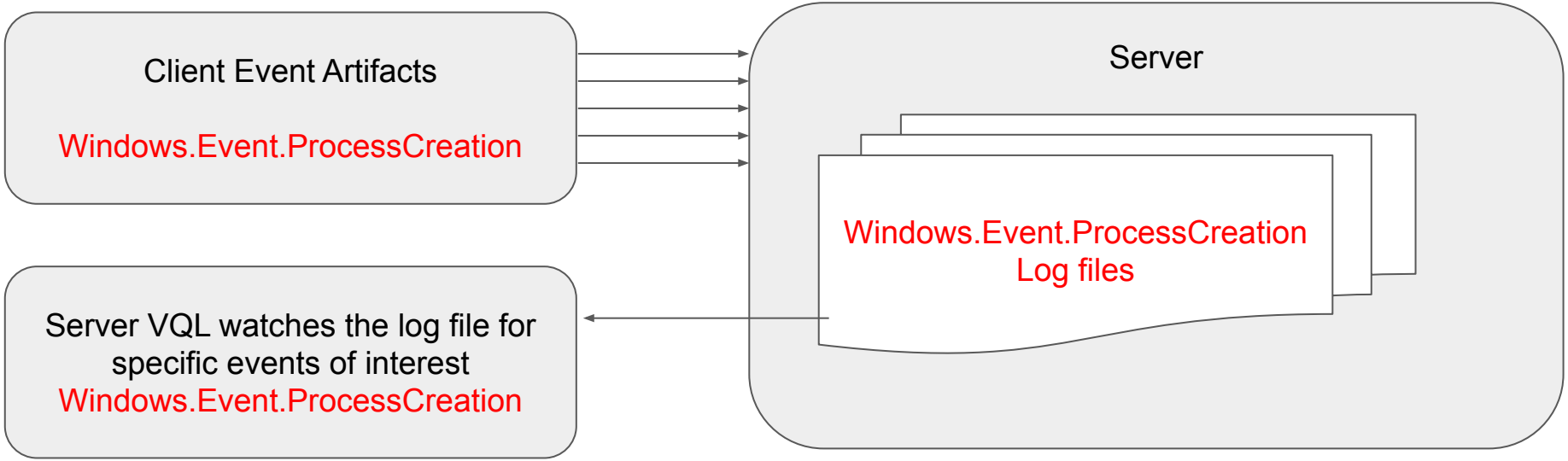
Any tool can be used on the fuse mount since it looks like a fixed disk.

Create a drive letter mapped into the file (or ntfs) path.

```
F:\>subst v: "q:\file\C%3A"

F:\>kape\KAPE\kape.exe --tsource V  --target Firefox --tdest "f:\output\Firefox"
KAPE version 0.8.0.0 Author: Eric Zimmerman (kape@kroll.com)
Command line: --tsource V --target Firefox --tdest f:\output\Firefox

Using Target operations
Found 18 targets. Expanding targets to file list...
        'V:\Users\Default\AppData\Roaming\Mozilla\Firefox\Profiles' does not exist. Skipping
        'V:\Users\Public\AppData\Roaming\Mozilla\Firefox\Profiles' does not exist. Skipping
        'V:\Users\someuser\AppData\Roaming\Mozilla\Firefox\Profiles' does not exist. Skipping
        'V:\Users\user\AppData\Roaming\Mozilla\Firefox\Profiles' does not exist. Skipping
        'V:\Users\Default\AppData\Roaming\Mozilla\Firefox\Profiles' does not exist. Skipping
```

Server side artifacts

- We can run event artifacts on the server. This allows us to act on client events



Exercise: Decode powershell encoded cmdline

- Powershell may accept a script on the command line which is base64 encoded. This makes it harder to see what the script does, therefore many attackers launch powershell with this option
- We would like to keep a log on the server with the decoded powershell scripts.
- Our strategy will be:
  - Watch the client's process execution logs as an event stream on the server.
  - Detect execution of powershell with encoded parameters
  - Decode the parameter and report the decoded script.
  - Store all results as another artifact.
- For testing use this:

```
powershell -encodedCommand ZABpAHIAIAAiAGMAOgBcAHAAcgBvAGcAcgBhAG0AIABmAGkAbABlAHMAIgAgAA==
```

VQL - Create an artifact with this query.

```
SELECT ClientId, ParentInfo, CommandLine, Timestamp,  utf16(
      string=base64decode(
           string=parse_string_with_regex(
                string=CommndLine,
                regex='-encodedcommand (?P<Encoded>[^ ]+)'
           ).Encoded
      )) AS Script
FROM watch_monitoring(artifact='Windows.Events.ProcessCreation')
WHERE CommandLine =~ '-encodedcommand'
```

Extract the encoded command from the command line, then base64 and utf16 decode it.

We only care about powershell command lines with encoded commands

Watch the monitoring logs (for all clients) for any new rows for this artifact.

Collect the artifact with a python program.

1. Copy the example python API client directory to your machine.
2. Install the required libraries:

c:\Python27\Scripts\pip.exe install -r requirements.txt

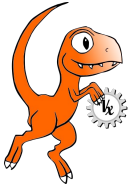
- Use the sample program to run the previous query.

c:\Python27\python.exe client_example.py api_client.config.yaml "`SELECT * FROM Artifact.Windows.Powershell.Decoded()` "

- Python programmers can now do whatever with the data live...

Conclusions

What it can do is only limited by your imagination!
What will you think of?



https://github.com/Velocidex/velociraptor

Appendix - Installing Grafana

Velociraptor | C.11a3013cca | Download Grafana | Grafan | Home - Grafana

localhost:3000/?orgId=1

Home

Home Dashboard

~~Install Grafana~~

Add data source

Create your first dashboard

Invite your team

Install apps & plugins

Starred dashboards

Recently viewed dashboards

Installed Apps

None installed. Browse Grafana.com

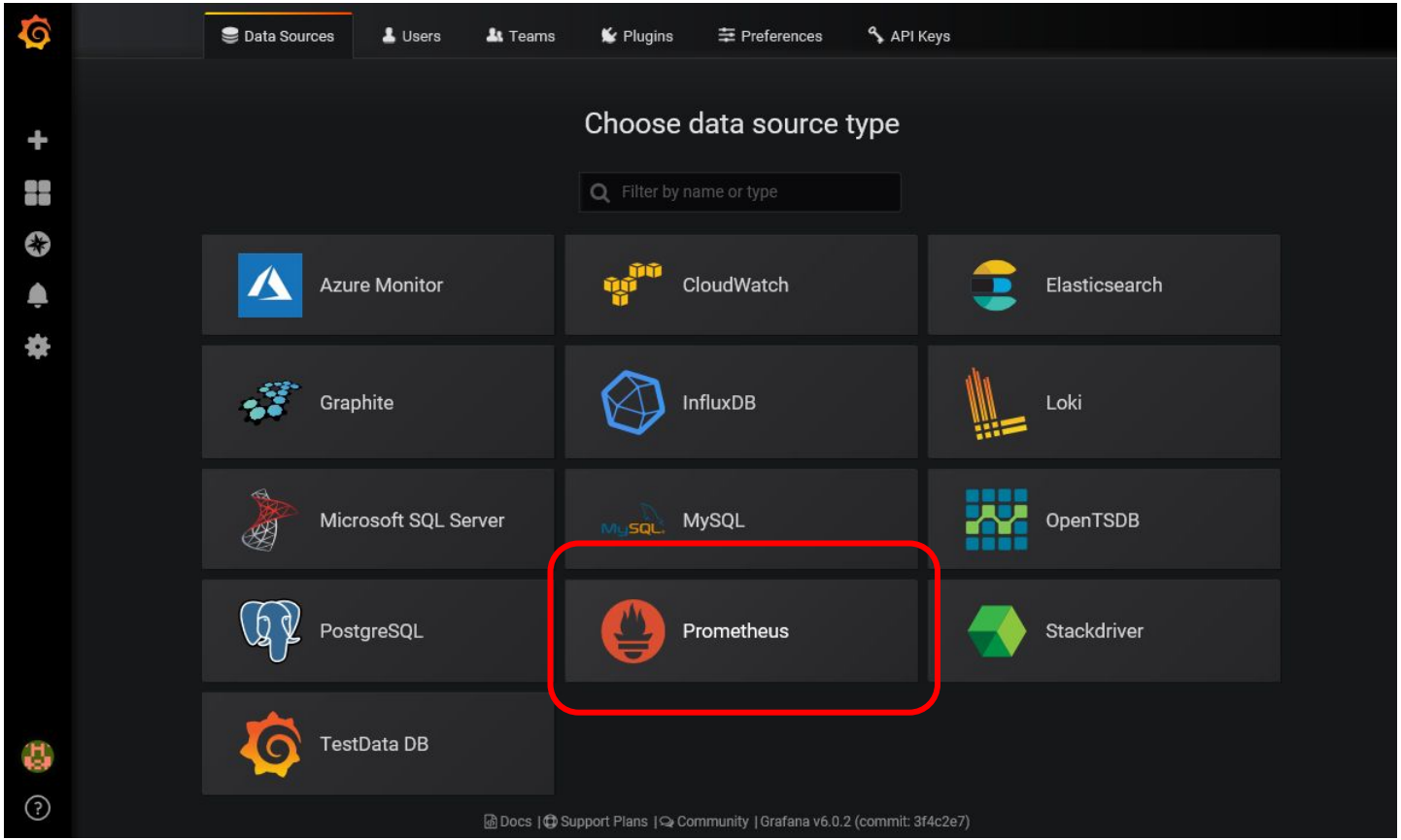
Installed Panels

None installed. Browse Grafana.com

Installed Datasources

None installed. Browse Grafana.com

http://localhost:3000/datasources/new?gettingstarted


Data Sources
Users
Teams
Plugins
Preferences
API Keys
Choose data source type
Filter by name or type
Azure Monitor
CloudWatch
Elasticsearch
Graphite
InfluxDB
Loki
Microsoft SQL Server
MySQL
OpenTSDB
PostgreSQL
Prometheus
Stackdriver
TestData DB
Docs | Support Plans | Community | Grafana v6.0.2 (commit: 3f4c2e7)

Data Sources / Prometheus

Type: Prometheus

Create

Dashboard

Folder

Import

Settings

Dashboards

me

Prometheus

Default

HTTP

URL

http://localhost:9090

Access

Server (Default)

Help

Whitelisted Cookies

Add Name

Auth

Basic Auth

With Credentials

TLS Client Auth

With CA Cert

Skip TLS Verify

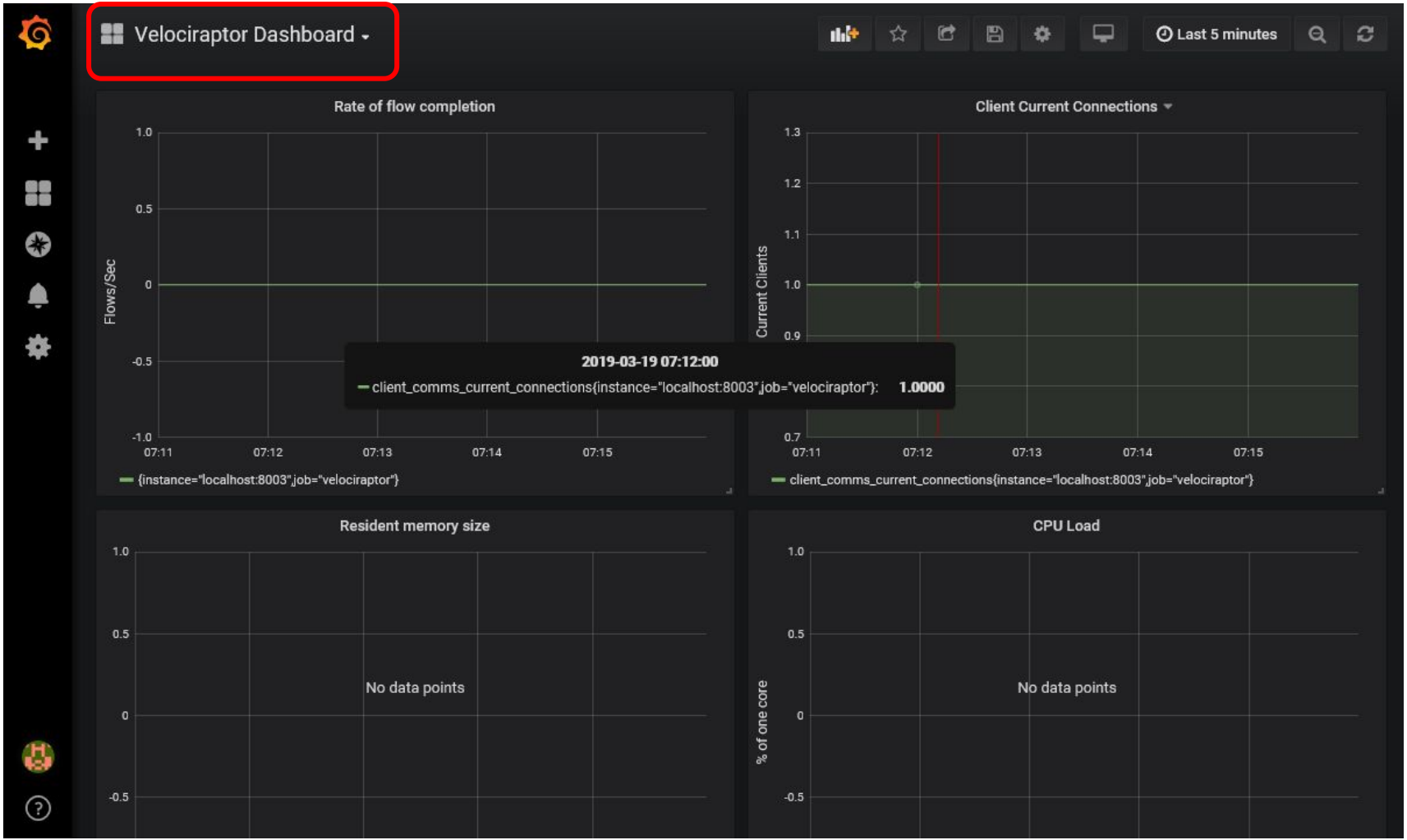
Scrape interval

15s

Query timeout

60s

Import the provided dashboard as a starting point. Feel free to tweak as needed

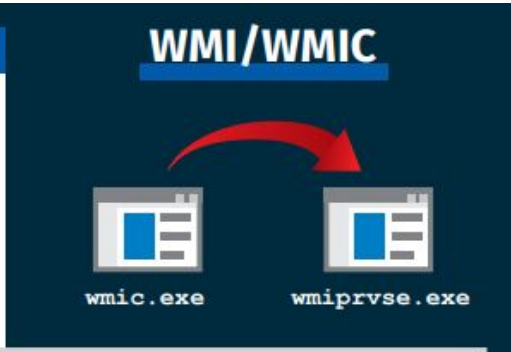

Velociraptor Dashboard
Last 5 minutes
Rate of flow completion
Flows/Sec
1.0
0.5
0
-0.5
-1.0
07:11
07:12
07:13
07:14
07:15
2019-03-19 07:12:00
client_comms_current_connections{instance="localhost:8003",job="velociraptor"}: 1.0000
{instance="localhost:8003",job="velociraptor"}
Client Current Connections
Current Clients
1.3
1.2
1.1
1.0
0.9
0.7
client_comms_current_connections{instance="localhost:8003",job="velociraptor"}
Resident memory size
No data points
CPU Load
% of one core
No data points

Lateral Movement wmi process creation

- WMI may be used to create processes remotely.
- Try it yourself

wmic process create cmd.exe

REGISTRY	FILE SYSTEM
■ ShimCache – SYSTEM • wmic.exe ■ BAM / DAM – SYSTEM – Last Time Executed • wmic.exe ■ AmCache.hve – First Time Executed • wmic.exe	■ Prefetch – C:\Windows\Prefetch\ • wmic.exe-{hash}.pf




```
wmic /node:host process call create "evil.exe"
Invoke-WmiMethod -Computer host -Class Win32_Process -Name create -Argument "c:\temp\evil.exe"
```