

Talk is cheap, IR tools can be too



INVESTORS
IN PEOPLE



By Royal Charter



Background

- Networks are breached by four main groups:
 - State Targeted
 - E-Crime
 - Hacktivists
 - Script Kiddies
- The reported number of breaches is going up:
 - Either more attacks are occurring or
 - More attacks are being detected

Information Security Breaches Survey (HMGUK)

- 90% of large organizations breached
- 74% of small organizations breached
- £1.46m to £3.14m average cost of breach for large organization
- £75k to 311k average cost of breach for small organization

https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/432412/bis-15-302-information_security_breaches_survey_2015-full-report.pdf

Breaches

- US Office of Personal Management (OPM): 19.7m records
- FBI portal: Figures not disclosed
- Ashley Madison: 11m records
- TalkTalk: 150k records, cost over £60m
- Carphone Warehouse: 2.5m records

Key point: It can happen to anyone

Network Defence

- Signature based solutions do not work well
- Traditional anti-virus solutions do not work well
- Numerous different commercial solutions are available that are designed to “protect” the network and it’s hosts
- End point security is becoming increasingly common
- Devices are expensive
- End points are expensive

Problems

- Organizations have a finite budget
- Cloud based solutions may not be desired
- Hosts may not be externally visible or connected to the Internet
- Organizations may prefer to use in house security teams

Solutions

- There are a number of simple ways that can provide great benefits to defending a network
- You don't need to blow your budget on a black box with a nice blue flashing light...though they look cool

Solutions

- The key point is to always assume your network is breached and therefore the security team should always be on the hunt
- This presentation is aims to:
 - introduce Autorun Logger as a tool
 - cover other tools that can be implemented
 - describe other processes that can help detect attackers
 - briefly introduce two new BSI systems



AutoRun Logger (ARL)

AutoRun Logger (ARL) - Background

- Malicious software wants to have a persistence mechanism (Autoruns) so that it can survive reboots
- The go to tool for extracting autoruns is the SysInternals Autoruns software, written by Mark Russinovich (Microsoft)
- AutoRun-Logger is a three component system that allows organizations to have a network wide view of all of the autorun data of their Windows hosts.

AutoRun Logger (ARL) - Architecture

- ARL is a three component system:
 - Windows service that extracts autorun data periodically and sends to the Analysis server
 - Analysis server parses autorun data and compares to previous set
 - Any changes e.g. adds/edits/deletions are alerted on
 - Generates summary data for data stacking analysis
 - Alerts and summary data files viewed via UI server

AutoRun Logger – Architecture

- Windows service is written in .Net (C#)
 - Has service auto-install code
- Analysis and UI servers are written in golang and are designed to run on Linux
- The servers use a PostgreSQL database to store the data
- Unique instances of each domain/host autorun can be stored as archives



Other tools

Other techniques/defenses (Ransomware)

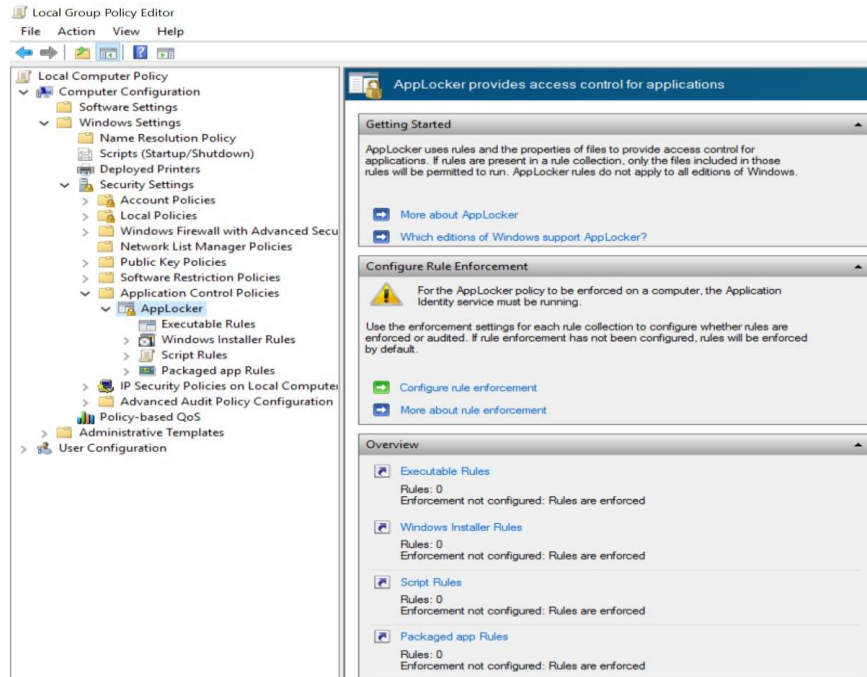
- Ransomware is one of the most common threats to an organization
- There are some simple and easy to implement strategies that can help limit the threat of ransomware malware:
 - Remove local administrator rights
 - Use AppLocker to prevent the execution of applications within the User profile directories



AppLocker

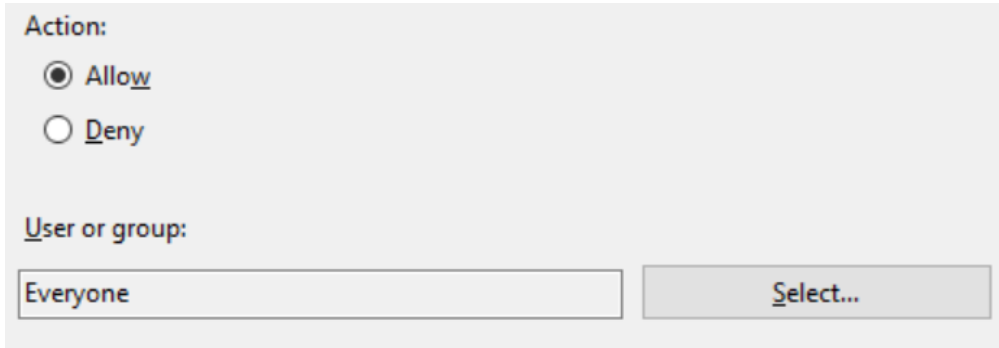
AppLocker

- GPO->Computer Configuration->Windows Settings->Security Settings->Application Control Policies->AppLocker



AppLocker

- Create a new **Executable Rule**: Right click on the **Executable Rules** tree item, select the **Create New Rule** menu item)
- Configure the rule:



The screenshot shows a configuration window for an AppLocker rule. It has a light gray background. At the top, the label "Action:" is in blue. Below it are two radio buttons: the first is selected and labeled "Allow" in blue, and the second is unselected and labeled "Deny" in blue. Below these is the label "User or group:" in blue. Underneath is a text box containing the word "Everyone" and a button to its right labeled "Select..." in blue.

- Next, **Allow** for **Everyone** (These are the defaults)

AppLocker

- Select the primary condition type as **Path**:

Select the type of primary condition that you would like to create.

☐ Publisher
Select this option if the application you want to create the rule for is signed by the software publisher.

☒ Path
Create a rule for a specific file or folder path. If you select a folder, all files in the folder will be affected by the rule.

☐ File hash
Select this option if you want to create a rule for an application that is not signed.

AppLocker

- Enter * as the Path and click **Next**:

Select the file or folder path that this rule should affect. If you specify a folder path, all files underneath that path will be affected by the rule.

Path:

AppLocker

- Add an Exception: Set the exception type as **Path**, and enter the exception path as **%OSDRIVE%\Users***, then click **Next**

To add an exception, select the type of exception and then click Add. Exceptions are optional and allow you to exclude files that would normally be included in the rule. To continue configuring this rule without adding an exception, click Next.

Primary condition:
*

Add exception:

Path

Exceptions:

Exception	Type
%OSDRIVE%\Users*	Path

Add... Edit Remove

AppLocker

- Enter the rule name e.g. Allow Executables Only Outside of User Profile:

Enter a name to identify this rule.

Name:

Allow Executables Only Outside of User Profile

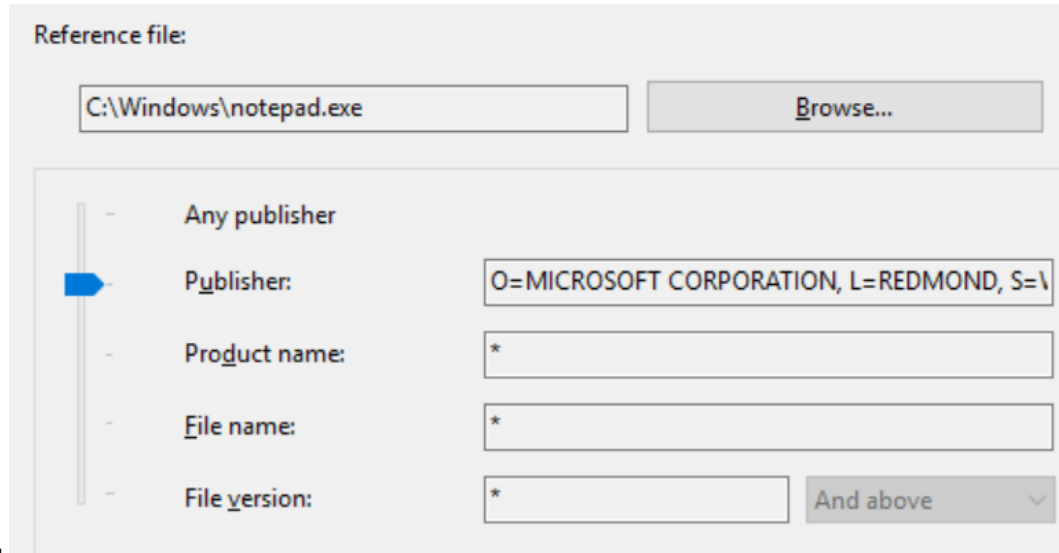
Description: (Optional)

AppLocker

- Some legitimate applications run from within the User profile directories, therefore exceptions are likely to be required
- Examples of such applications are:
 - Google Chrome
 - Dropbox
 - Microsoft OneDrive
- Rules can easily be created that are based on publisher, so that all applications that are signed by companies such as Microsoft, Google, Dropbox are permitted, regardless of location

AppLocker

- To define a **Publisher** rule, create the rule as before, but choose **Publisher**, rather than **Path**
- Choose a reference executable; for example **notepad.exe** and move the slider to **Publisher**



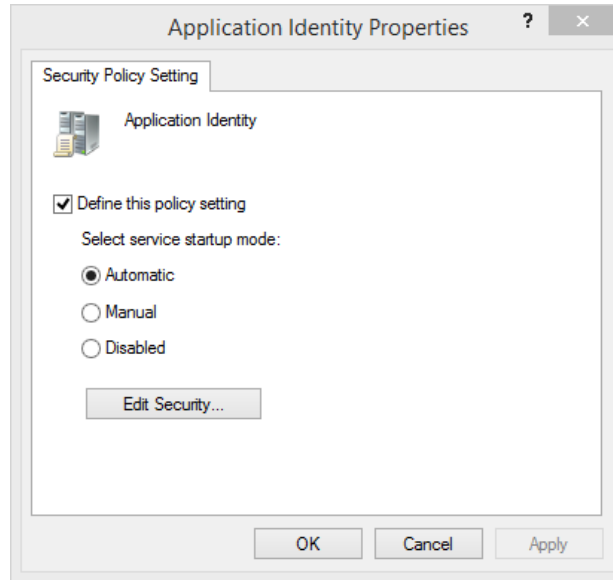
The screenshot shows the AppLocker rule configuration window. At the top, the "Reference file:" section contains a text box with "C:\Windows\notepad.exe" and a "Browse..." button. Below this, a vertical slider on the left allows selecting the rule type. The "Publisher" option is selected, indicated by a blue arrow. To the right of the slider, the "Publisher:" field contains "O=MICROSOFT CORPORATION, L=REDMOND, S=\". Below this are four more fields: "Product name:", "File name:", and "File version:", each containing an asterisk (*). At the bottom right, there is a dropdown menu labeled "And above" with a downward arrow.

AppLocker

- Add rules for each of the other required publishers
- Rules can be defined using the file hash value, which is fine for binaries that do not change often
- Rules based on path are not recommended as they can be used as launched points
- Applications such as Google Chrome can be installed in the Program Files directory which removes the need for a rule

AppLocker

- Ensure that the **Application Identity** service startup mode is **Automatic**:
 - Computer Configuration → Windows Settings → System Services → Application Identity





Enhanced Mitigation Experience Toolkit

Enhanced Mitigation Experience Toolkit (EMET)

- EMET is a utility that helps prevent vulnerabilities in software from being successfully exploited:

EMET Security Mitigations	Included
Attack Surface Reduction (ASR) Mitigation	✓
Export Address Table Filtering (EAF+) Security Mitigation	✓
Data Execution Prevention (DEP) Security Mitigation	✓
Structured Execution Handling Overwrite Protection (SEHOP) Security Mitigation	✓
NullPage Security Mitigation	✓
Heapspray Allocation Security Mitigation	✓
Export Address Table Filtering (EAF) Security Mitigation	✓
Mandatory Address Space Layout Randomization (ASLR) Security Mitigation	✓
Bottom Up ASLR Security Mitigation	✓
Load Library Check – Return Oriented Programming (ROP) Security Mitigation	✓
Memory Protection Check – Return Oriented Programming (ROP) Security Mitigation	✓
Caller Checks – Return Oriented Programming (ROP) Security Mitigation*	✓
Simulate Execution Flow – Return Oriented Programming (ROP) Security Mitigation*	✓
Stack Pivot – Return Oriented Programming (ROP) Security Mitigation	✓
Windows 10 untrusted fonts***	✓

EMET

- Current version is 5.5
- Released in January 2016
- Compatible with Windows 10
- Some security features within Windows 10 are supposed to be better than those provided by EMET
- CESG guidelines (Alpha) still recommend that EMET is used
<https://www.gov.uk/government/publications/end-user-devices-security-guidance-windows-10/end-user-devices-security-guidance-windows-10>
- Due to the way that EMET works it can interfere with the way applications work...e.g. it can break them! So it is worth testing

EMET

- The CESG EMET settings are:

Group Policy	Value(s)
Computer Configuration > Administrative Templates > Windows Components > EMET > Default Action and Mitigation Settings	Enabled Deep Hooks: Enabled Anti Detours: Enabled Banned Functions: Enabled Exploit Action: Stop Program
Computer Configuration > Administrative Templates > Windows Components > EMET > System DEP	Enabled DEP Setting: Always On



Powershell

PowerShell

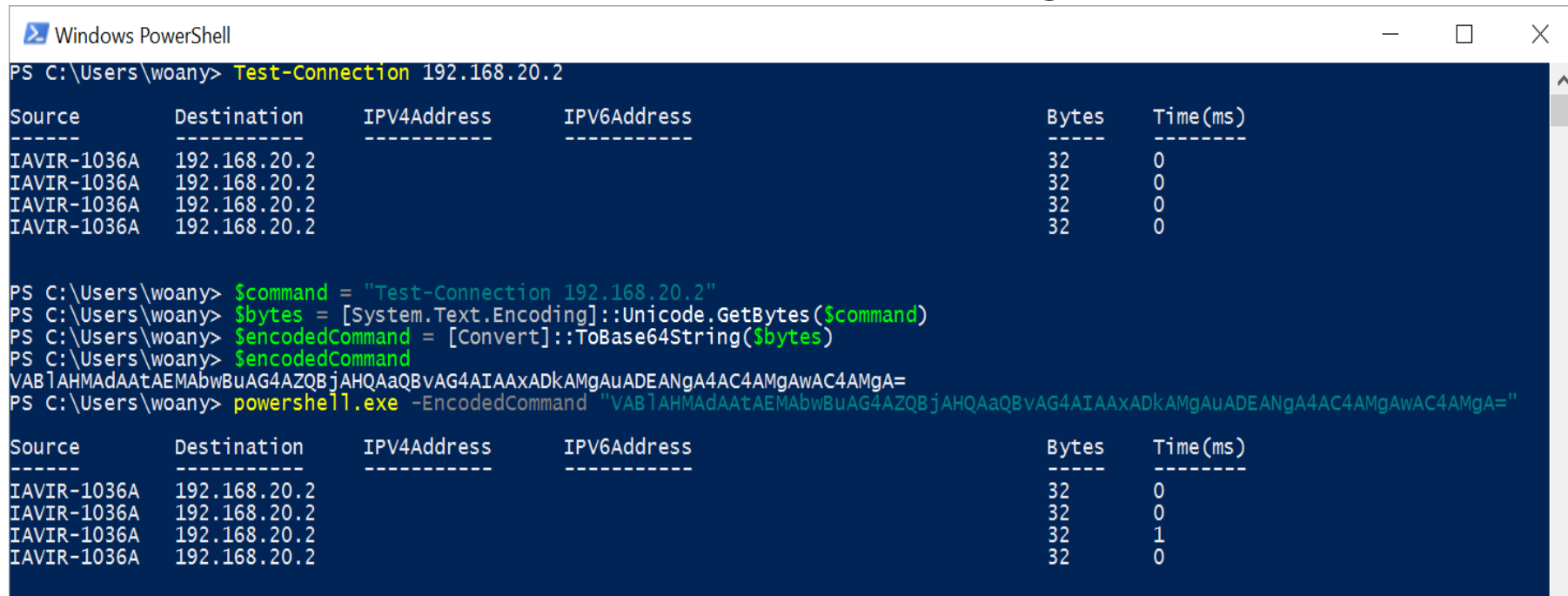
- PowerShell v1 was released in November 2006
- Current version is v5 (December 2016)
- Windows 7/Windows 10 ship with PowerShell v2
- Console based system that provides powerful automation, configuration and scripting capabilities
- Heavily tied into the .NET Framework
- Can access the .NET objects directly

PowerShell

- When discussing PowerShell in the context of network defence, the phrase **“with great power comes great responsibility”** is particularly applicable
- Attackers would rather not introduce their tools onto a network as it increases the chances of being identified
- PowerShell provides an attacker with a great platform for privilege escalation, lateral movement, credential theft, persistence mechanism, data exfiltration
- PowerShell in a default state can leave little in forensic artefacts

PowerShell

- Attackers often obfuscate PowerShell commands using Base64:



The screenshot shows a Windows PowerShell window with a dark blue background. The title bar reads 'Windows PowerShell'. The command prompt shows the user 'woany' at 'C:\Users\woany>'. The first command is 'Test-Connection 192.168.20.2', which outputs a table of connection statistics. The second command is a sequence of PowerShell commands to obfuscate the 'Test-Connection' command using Base64 encoding. The final command is 'powershell.exe -EncodedCommand "VAB1AHMAdAAtAEMabwBuAG4AZQBjAHQAaQBVAG4AIAAXADkAMgAuADEANgA4AC4AMgAwAC4AMgA="', which then outputs the same connection statistics table.

```
PS C:\Users\woany> Test-Connection 192.168.20.2
```

Source	Destination	IPv4Address	IPv6Address	Bytes	Time (ms)
IAVIR-1036A	192.168.20.2			32	0
IAVIR-1036A	192.168.20.2			32	0
IAVIR-1036A	192.168.20.2			32	0
IAVIR-1036A	192.168.20.2			32	0

```
PS C:\Users\woany> $command = "Test-Connection 192.168.20.2"
PS C:\Users\woany> $bytes = [System.Text.Encoding]::Unicode.GetBytes($command)
PS C:\Users\woany> $encodedCommand = [Convert]::ToBase64String($bytes)
PS C:\Users\woany> $encodedCommand
VAB1AHMAdAAtAEMabwBuAG4AZQBjAHQAaQBVAG4AIAAXADkAMgAuADEANgA4AC4AMgAwAC4AMgA=
PS C:\Users\woany> powershell.exe -EncodedCommand "VAB1AHMAdAAtAEMabwBuAG4AZQBjAHQAaQBVAG4AIAAXADkAMgAuADEANgA4AC4AMgAwAC4AMgA="
```

Source	Destination	IPv4Address	IPv6Address	Bytes	Time (ms)
IAVIR-1036A	192.168.20.2			32	0
IAVIR-1036A	192.168.20.2			32	0
IAVIR-1036A	192.168.20.2			32	1
IAVIR-1036A	192.168.20.2			32	0

PowerShell

The Good:

- PowerShell v5 has improved logging capabilities:
 - Script Block Logging
 - Constrained Mode
 - Antimalware Scan Interface Integration (AMSI) (only Win10)
 - Protected Event Logging (only Win10)

The Bad:

- You need to install PowerShell v5
- Ideally you want to install Windows 10 as well

Both of the **bad** items are actually good, but not that feasible in the short term for most large organizations

PowerShell

- A PowerShell “script block” is the base level of executable code in PowerShell
- A script block could be a command typed interactively in the PowerShell console, supplied through the command line (“PowerShell -Command <...>”), or wrapped in a function or script
- Attack tools and commands are commonly are obfuscated, often using Base64 encoding, which makes it more difficult to detect or identify what code actually ran

PowerShell

- Script block logging logs the actual code delivered to the PowerShell engine before
- Script block logging de-obfuscates the code and logs the code that is executed
- As the code is de-obfuscated it can be logged to a central SIEM and thus can be alerted on

PowerShell

Example of how this functionality can help unwrap a malicious attempt to encrypt and obfuscate a script:

```
## Malware
function SuperDecrypt
{
    param($script)
    $bytes = [Convert]::FromBase64String($script)

    ## XOR "encryption"
    $xorKey = 0x42
    for($counter = 0; $counter -lt $bytes.Length; $counter++)
    {
        $bytes[$counter] = $bytes[$counter] -bxor $xorKey
    }
    [System.Text.Encoding]::Unicode.GetString($bytes)
}

$decrypted = SuperDecrypt "FUIwQitCNkInQm9CCKItQjFCNkJiQmVCEkI1QixCJkJlQg=="
Invoke-Expression $decrypted
```

PowerShell

Running this generates the following log entries:

```
Compiling Scriptblock text (1 of 1):
function SuperDecrypt
{
    param($script)
    $bytes = [Convert]::FromBase64String($script)
    ## XOR "encryption"
    $xorKey = 0x42
    for($counter = 0; $counter -lt $bytes.Length; $counter++)
    {
        $bytes[$counter] = $bytes[$counter] -bxor $xorKey
    }
    [System.Text.Encoding]::Unicode.GetString($bytes)
}
ScriptBlock ID: ad8ae740-1f33-42aa-8dfc-1314411877e3

Compiling Scriptblock text (1 of 1):
$decrypted = SuperDecrypt "FUIwQitCNkInQm9CCKItQjFCNkjiQmVCEkI1QixCjKj1Qg=="
ScriptBlock ID: ba11c155-d34c-4004-88e3-6502ecb50f52

Compiling Scriptblock text (1 of 1):
Invoke-Expression $decrypted
ScriptBlock ID: 856c01ca-85d7-4989-b47f-e6a09ee4eeb3

Compiling Scriptblock text (1 of 1):
Write-Host 'Pwnd'
ScriptBlock ID: 5e618414-4e77-48e3-8f65-9a863f54b4c8
```

PowerShell

- PowerShell v5 automatically locks down when AppLocker is deployed in **Allow** mode
- AppLocker **Allow** mode performs binary whitelisting and can prevent any unauthorized binary from being executed
- When AppLocker **Allow** mode is in effect, the PowerShell language is set to **Constrained Mode**
- Constrained Mode PowerShell only allows core PowerShell functionality and prevents execution of the extended language features often used by offensive PowerShell tools (direct .NET scripting, invocation of Win32 APIs via the Add-Type cmdlet, and interaction with COM objects)
- Enterprise signed code or scripts in a trusted directory are executed in full PowerShell mode and not the Constrained PowerShell environment

PowerShell

- In Windows 10, PowerShell submits all script content (interactive or otherwise) to the registered anti-malware engine
- Includes additional calls for scripts that employ obfuscation or layer dynamic code evaluation

PowerShell

- With the increased logging there could be concerns that sensitive content such as credentials could be logged
- To prevent this, Windows 10 has Protected Event Logging
- Protected Event Logging lets participating applications encrypt sensitive data as the data is written to the event log
- Events can be decrypted and processed once they have been moved to a secure, centralised log collector
- Protected Event Logging protects event log content through the IETF Cryptographic Message Syntax (CMS) standard
- The CMS encryption standard implements public key cryptography, where the keys used to encrypt content (the *public key*) and the keys used to decrypt content (the *private key*) are separate

PowerShell

- When implementing a protected event logging policy, a public key is deployed to all machines that have event log data you want to protect
- The corresponding private key is retained to post-process the event logs
- To enable the 'Enable Protected Event Logging' feature in Group Policy through Windows Components -> Administrative Templates -> Event Logging

PowerShell

- This setting requires an encryption certificate, which you can provide in one of several forms:
 - The content of a base-64 encoded X.509 certificate
 - The thumbprint of a certificate that can be found in the Local Machine certificate store
 - The full path to a certificate (can be local, or a remote share)
 - The path to a directory containing a certificate or certificates (can be local, or a remote share)
 - The subject name of a certificate that can be found in the Local Machine certificate store (usually deployed by PKI infrastructure)
- The resulting certificate must have 'Document Encryption' as an enhanced key usage (1.3.6.1.4.1.311.80.1), as well as either Data Encipherment or Key Encipherment key usages enabled

PowerShell

- Because the CMS format is an IETF standard, PowerShell supports the decryption of content generated by other conforming tools, and the content it generates can be decrypted by other conforming tools
- One of the more popular implementations to support the CMS message format is the OpenSSL library and command-line toolchain
- The primary challenge when exchanging data with the OpenSSL library comes from the OpenSSL assumption that the content is contained within an email message body in the P7M format
- Fortunately, these text-based headers are relatively easy to add and remove

PowerShell

PowerShell CMS format:

```
-----BEGIN CMS-----
MIIBqAYJKoZIhvcNAQcDoIIbMTCCAzuCAQAxggFQMIIbTAIBADA0MCAxHjAcBgNVBAMMFwx1ZWwhv
bG1AbWljcm9zb2Z0LmNvbQIQYHsbcXnjIJCtH+OhGmc1DANBgkqhkiG9w0BAQcwAASCAQAnkFHM
proJnFy4geFGfyNmXH3yeoPvwEYzdnsOVqqDPAd8D3wao77z70hJEXwz9GeFLnxD6djKV/tF4PxR
E27aduKSLbnxfpf/sepZ4fUkuGibnwWFrXGE3B1G26MCenHWjYQiqv+Nq32Gc97qEAERrhLv6S4R
G+2dJEnesW8A+z9QPo+DwYU5FzD0Td0ExrksWVckpLNR6j17Yaags3ltNVmbdEXekhi6Psf2MLMP
TS0791v2L0KeXFGuPOrdzPAwCkV0vNEqTEBeDnZGrjv/5766bM3GW34FXApod9u+VSFpBnqVOCBA
DVDraA6k+xwBt66cV840HLkh0kT02SIHMDwGCSqGSIB3DQEhATAdbglghkgBZQMEASoEEJbJaiRl
KMnBoD1dkb/FzSWAEBaL8xkFwCu0e1ZtDj7nSJc=
-----END CMS-----
```

OpenSSL P7M format:

```
MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/pkcs7-mime; smime-type=enveloped-data; name="smime.p7m"
Content-Transfer-Encoding: base64
```

```
MIIBqAYJKoZIhvcNAQcDoIIbMTCCAzuCAQAxggFQMIIbTAIBADA0MCAxHjAcBgNVBAMMFwx1ZWwhv
bG1AbWljcm9zb2Z0LmNvbQIQYHsbcXnjIJCtH+OhGmc1DANBgkqhkiG9w0BAQcwAASCAQAnkFHM
proJnFy4geFGfyNmXH3yeoPvwEYzdnsOVqqDPAd8D3wao77z70hJEXwz9GeFLnxD6djKV/tF4PxR
E27aduKSLbnxfpf/sepZ4fUkuGibnwWFrXGE3B1G26MCenHWjYQiqv+Nq32Gc97qEAERrhLv6S4R
G+2dJEnesW8A+z9QPo+DwYU5FzD0Td0ExrksWVckpLNR6j17Yaags3ltNVmbdEXekhi6Psf2MLMP
TS0791v2L0KeXFGuPOrdzPAwCkV0vNEqTEBeDnZGrjv/5766bM3GW34FXApod9u+VSFpBnqVOCBA
DVDraA6k+xwBt66cV840HLkh0kT02SIHMDwGCSqGSIB3DQEhATAdbglghkgBZQMEASoEEJbJaiRl
KMnBoD1dkb/FzSWAEBaL8xkFwCu0e1ZtDj7nSJc=
```



OS Query

osquery

- osquery is a framework for performing real-time OS querying
- osquery exposes an operating system as a high-performance relational database
- This design allows you to write SQL-based queries efficiently and easily to explore operating systems
- With osquery, SQL tables represent the current state of operating system attributes, such as:
 - Running processes
 - Loaded kernel modules
 - Open network connections

osquery

- osquery is available for various operating systems:
 - CentOS/RHEL
 - Ubuntu
 - OS X
 - Windows (Currently being developed)

osquery

- osquery can work in two modes:
 - Adhoc
 - Query
- The **Adhoc** mode can be used for DFIR use cases
 - The end point is installed and used to run real-time queries against the host
 - Cannot be used for long term detection e.g. what processes connected to the host with the IP X.X.X.X one week ago
 - The end point is lightweight and should not impact the host

osquery

- Query mode can be used for long term monitoring and for active threat hunting
- Requires the installation of the **osqueryd** daemon
- Pre-defined queries are run at specific intervals
- If the data has changed from the previous run, then the data is logged
- Data from the queries can be sent to a aggregating server
- Examples for log aggregation include:
 - Logstash
 - Splunk
 - Rsyslog

osquery

Example of a query using the osqueryi command line tool:

```
$ osqueryi
osquery> SELECT DISTINCT
...>   process.name,
...>   listening.port,
...>   process.pid
...> FROM processes AS process
...> JOIN listening_ports AS listening
...> ON process.pid = listening.pid
...> WHERE listening.address = '0.0.0.0';
```

```
+-----+-----+-----+
| name   | port  | pid   |
+-----+-----+-----+
| Spotify | 57621 | 18666 |
| ARDAgent | 3283  | 482   |
+-----+-----+-----+
osquery>
```

osquery

- Kolide is an agent-less osquery web interface
- Uses the osquery remote API's to:
 - Perform adhoc distributed queries
 - Modify osqueryd configurations
 - Collection and processing of scheduled queries
- Written by Dustin Webber, developer of the snort web UI snorby



Passive DNS

Passive DNS

- Merike Kaeo from Internet Systems Consortium (ISC) defines Passive DNS as:
 - A technology invented in 2004 by Florian Weimer
 - Inter-server DNS messages are captured by sensors and forwarded to a collection point for analysis
 - After being processed, individual DNS records are stored in a database where they can be indexed and queried
- It allows for the following questions to be answered:
 - Where did this domain name point to in the past?
 - What domain names are hosted by a given nameserver?
 - What domain names point into a given IP network?
 - What subdomains exist below a certain domain name?

Passive DNS

- Install a Passive DNS sensor on the internal network
 - passiveDNS (<https://github.com/gamelinux/passivedns>)
 - sie-dns-sensor (<https://github.com/farsightsec/sie-dns-sensor>)
- The **sie-dns-sensor** sends encrypted, anonymised data to a global store
- Users that participate using the **sie-dns-sensor** can access the Passive DNS web UI @ <https://www.dnsdb.info>

Passive DNS

- From Presentation:
[http://files.sans.org/summit/Threat Hunting Incident Response Summit 2016/PDFs/Hunting-on-the-Cheap-Butler-Ahuja-Morris-Endgame.pdf](http://files.sans.org/summit/Threat%20Hunting%20Incident%20Response%20Summit%202016/PDFs/Hunting-on-the-Cheap-Butler-Ahuja-Morris-Endgame.pdf)
- Key techniques:
 - Identify fast flux DNS used by malware e.g. large numbers of IP addresses associated with a single domain
 - Identify Domain Generated Algorithm (DGA) domains used by malware e.g. ptiautjtthpnxdcw.com
 - Identify traffic volumes e.g. unusual volumes @ unusual times and periodicity such as beaconing



Memory Forensics

Memory Forensics

- Memory forensics can be:
 - Performed relatively easily
 - Performed relatively quickly
 - Can be automated
- Memory forensics can:
 - Show artefacts that only exist in memory
 - Help identify root kits and memory resident malware

Memory Forensics

- There are two frameworks for performing memory analysis
 - <https://github.com/volatilityfoundation/volatility>
 - <https://github.com/google/rekall>
- Rekall is a fork of the Volatility project, created in 2013
- Rekall is designed to be modular, which allows for it's integration into Google's Rapid Response (GRR) project

Memory Forensics

- The Rekall project has the **pmem*** tools that can be used to extract memory from various operating systems:
 - Windows
 - Linux
 - OSX

<https://github.com/google/rekall/tree/master/tools/pmem>

Memory Forensics

- Identify the top 10 key compromise targets such as file servers, email servers, remote access servers, key employee laptops/desktops
- Periodically extract memory and perform memory analysis
- Once the target OS/memory profile is known, then key information can be automatically extracted such as:
 - Processes
 - Network Connections
 - Drivers
 - Services

Memory Forensics

- Ensure your staff have had appropriate memory forensics training
- Perform full memory forensics on each of the targets periodically (bi-monthly, quarterly, half yearly)



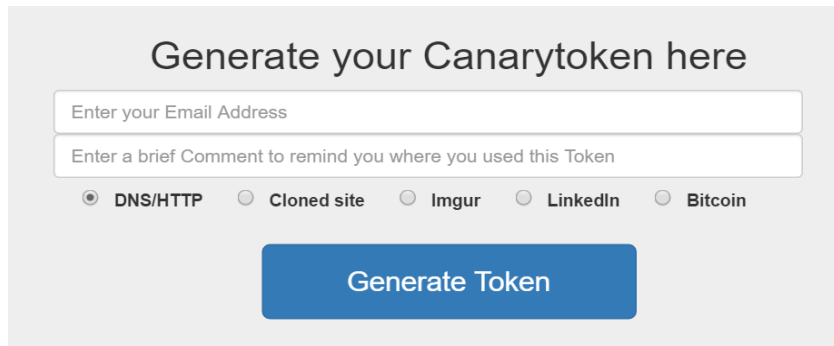
Canary Tokens

Canary Tokens

- Canary tokens are data or computer resources that are used create triggers within production infrastructure
- Created by Thinkst as an open source and commercial offering
- Unique values (tokens) are generated and embed within documents (Word, PDF), Windows folders (desktop.ini), email messages, databases
- When the triggers are actioned, alerts are sent

Canary Tokens

- Canary tokens are a quick and cheap (via open-source) way to provide network defenders a way to determine if they have been breached
- The easiest method is to use the <http://canarytokens.org/> site:
 - Generate a new token



The screenshot shows the 'Generate your Canarytoken here' interface. It features two input fields: 'Enter your Email Address' and 'Enter a brief Comment to remind you where you used this Token'. Below these fields are five radio button options: 'DNS/HTTP' (selected), 'Cloned site', 'Imgur', 'LinkedIn', and 'Bitcoin'. A blue 'Generate Token' button is positioned at the bottom of the form.

Canary Tokens

- Embed the token within a MS Word, PDF etc
- Wait for an alert
- Issues with this method is that the domain contained within the token cannot be controlled or configured
- The alternative is to host the canary server yourself
 - Use a third party hosting provider that supports virtual private Linux hosts
 - Install the canary tools via a Docker instance
 - Configure the domains and email account

Canary Tokens

- There are various ways to embed canary tokens:
 - A Windows file share: Modify the desktop.ini file to include the token value

```
[.ShellClassInfo]
```

```
IconResource=\\%USERNAME%.%USERDOMAIN%.INI.mgx45ipiak.canarytokens.com\resource.dll
```

- This method can also be used in conjunction with zip archive file
- The canary tokens web site creates MS Word and PDF documents containing the trigger and the token

Canary Tokens

- The following shows an example of an email trigger alert:

Canarytoken Mailer <noreply@canarytokens.org>

to me 

One of your canarydrops was triggered.

Channel: DNS

Time : 2016-07-18 13:37:52.408029

Memo : email

Source IP : 81.

Manage your settings for this Canarydrop:

<http://canarytokens.org/manage?token=sksa9twu2fqscj9lvqngx62s1&auth=91172082de1e83acc4f0f42bf145482f>