



IT-Security (ITS) B1

DIKU, E2021



Today's agenda

Intrusion detection

Next time: Forensics



If or when?

“There are two kinds of companies.

There are those who've been hacked and those who don't know they've been hacked.”

[Former FBI Director, James Comey](#)

Where should we focus?





Warm-up



Is this an incident?

[**] IIS vti_inf access attempt [**]

05/31-21-09:16:13 63.209.91.31:4791 -> 10.0.0.13:80

TCP TTL:116 TOS:0x0 ID:6075 DF

***PA* Seq:0x1CB4699 Ack:0x2AE6F9 Win:0x217C

← **IDS alert**

[Tue May 31 09:16:13 2021] [error] [client 63.209.91.31]

File does not exist: /usr/local/apache/htdocs/_vti_inf.html

[Tue May 31 09:16:14 2021] [error] [client 63.209.91.31]

File does not exist: /usr/local/apache/htdocs/_vti_bin/shtml.exe/_vti_rpc

← **Web server log**



Key characteristics in Intrusion Detection



Overall IT-security goals

Prevent as much as possible with *best practices* such as secure coding, whitelisting, patching, secure configurations and more

Anticipate breaches and **build to contain** with defence in depth, segmentation, least privilege, etc.

Detect and **respond** when things go wrong

Learn and **repeat**



Key activities in Intrusion Detection

Threat Assessment

How are we exposed (as a company, our business processes, and underlying IT)?

Visibility

What is the right level of insight we need in our systems and applications to detect intrusions?

Data Collection

How do we collect data to support our visibility needs?

Data Analysis

How do we analyse the data for signs of intrusions?

Incident Response

What do we do when we discover an attack?

Overall timeline in Intrusion Detection

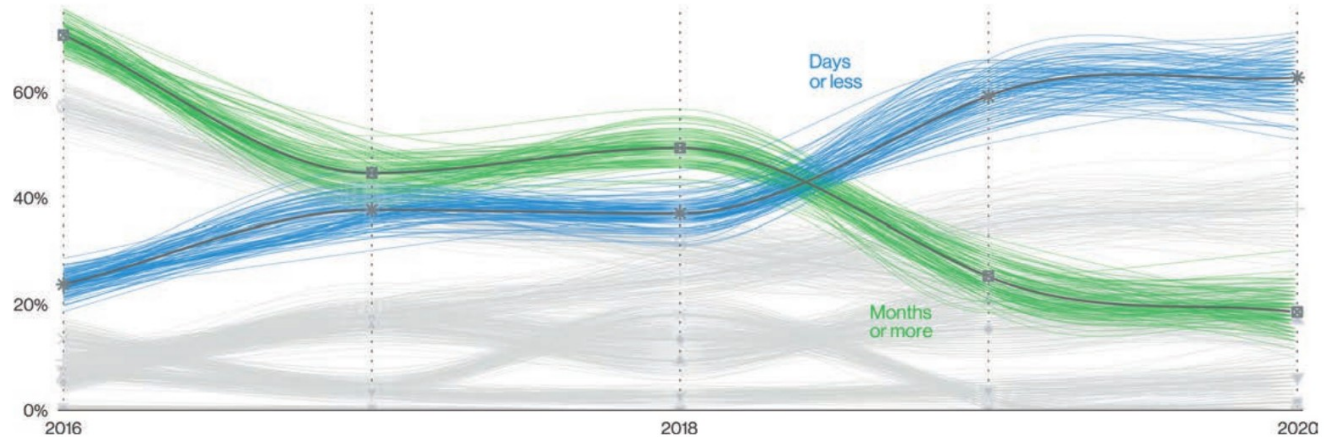
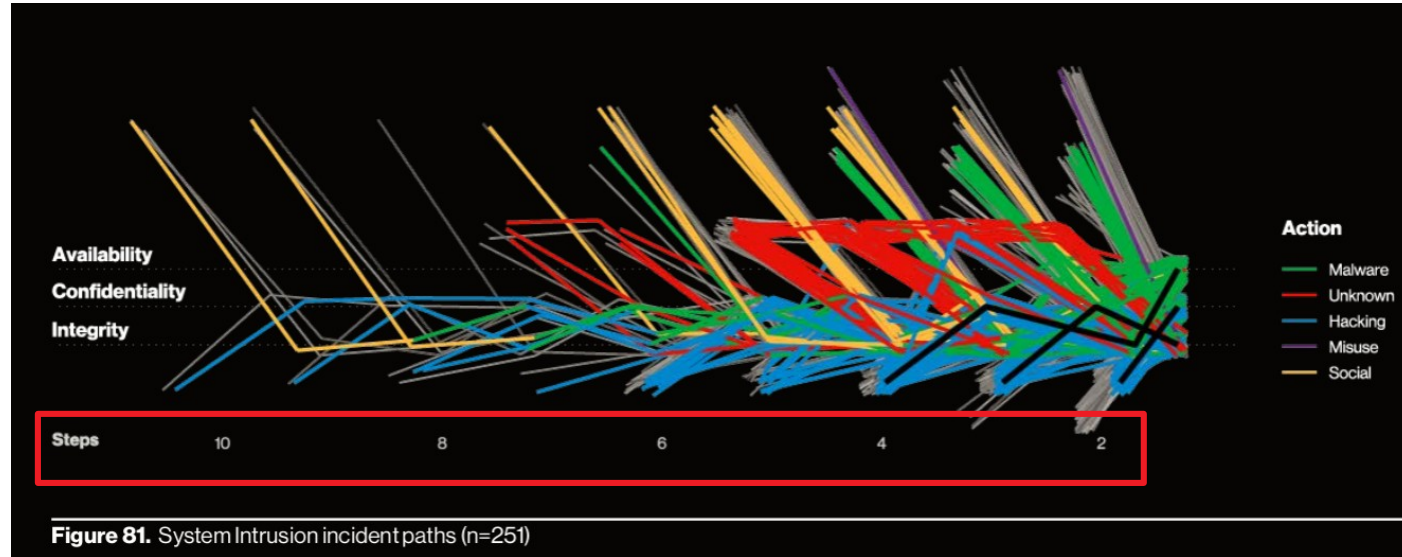


Figure 39. Discovery over time in breaches

Overall timeline in Intrusion Detection



NIST Security Incident Handling process

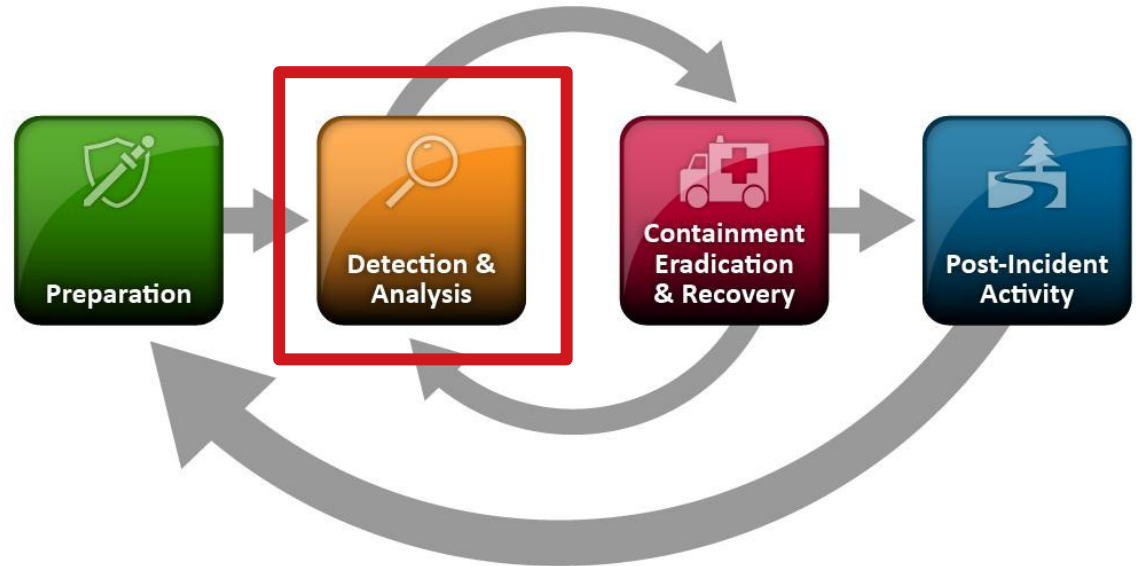
NIST
National Institute of
Standards and Technology
U.S. Department of Commerce

Special Publication 800-61
Revision 2

Computer Security Incident Handling Guide

Recommendations of the National Institute
of Standards and Technology

Paul Cichonski
Tom Millar
Tim Grance
Karen Scarfone





Visibility

Visibility

Q: What is the right level of insight we need in our systems and applications to detect intrusions?

A: Study how hackers actually hack: The **Cyber Kill Chain**:



MITRE ATT&CK

The Cyber Kill Chain is a good resource, but somewhat high-level.

MITRE ATT&CK to the rescue:

ATT&CK Matrix for Enterprise

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	Binary Padding	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	BITS Jobs	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 from Information Repositories	Custom Command and Control Protocol	Data Transfer Size Limits	Disk Content Wipe
Replication Through Removable Media	Control Panel Items	AppInit DLLs	Application Shimming	Clear Command History	Credentials in Files	File and Directory Discovery	Logon Scripts	Data from Local System	Custom Cryptographic Protocol	Exfiltration Over Alternative Protocol	Disk Structure Wipe
Spearphishing Attachment	Dynamic Data Exchange	Application Shimming	Bypass User Account Control	CMSTP	Credentials in Registry	Network Service Scanning	Pass the Hash	Data from Network Shared Drive	Data Encoding	Exfiltration Over Command and Control Channel	Endpoint Denial of Service
Spearphishing Link	Execution through API	Authentication Package	DLL Search Order Hijacking	Code Signing	Exploitation for Credential Access	Network Share Discovery	Pass the Ticket	Data from Removable Media	Data Obfuscation	Exfiltration Over Other Network Medium	Firmware Corruption
Spearphishing via Service	Execution through Module Load	BITS Jobs	Dylib Hijacking	Compile After Delivery	Forced Authentication	Network Sniffing	Remote Desktop Protocol	Data Staged	Domain Fronting	Exfiltration Over Physical Medium	Inhibit System Recovery
Supply Chain Compromise	Exploitation for Client Execution	Bootkit	Exploitation for Privilege Escalation	Compiled HTML File	Hooking	Password Policy Discovery	Remote File Copy	Email Collection	Domain Generation Algorithms	Scheduled Transfer	Network Denial of Service
Trusted Relationship	Graphical User Interface	Browser Extensions	Extra Window Memory Injection	Component Firmware	Input Capture	Peripheral Device Discovery	Remote Services	Input Capture	Fallback Channels		Resource Hijacking
Valid Accounts	Installable	Change Default File Association	File System Permissions	Component Object Model	Input Prompt	Permission Groups Discovery	Replication Through	Man in the Browser	Multi-Use Proxy		Runtime Data



What are ATT&CK TTPs?

TTPs = Tactics, Techniques, and Procedures

Tactics

The “why” of an ATT&CK technique. It is the adversary’s tactical goal: the reason for performing an action. For example, an adversary may want to achieve credential access.

Techniques

The “how” an adversary achieves a tactical goal by performing an action. For example, an adversary may dump credentials to achieve credential access.

Procedures

The “specific” implementation the adversary uses for a technique. For example, a procedure could be an adversary using PowerShell to inject into lsass.exe to dump credentials by scraping LSASS memory on a victim machine.



Example: SDelete

SDelete is an application that securely deletes data in a way that makes it unrecoverable. It is part of the Microsoft Sysinternals suite of tools.

Tactic:	Impact	(ID: TA0040)
Technique:	Data Destruction	(ID: T1485)
Procedure:	Sdelete	(ID: S0195)

Threat actor groups observed in the wild using this procedure:

- * G0053FIN5
- * G0080Cobalt Group
- * G0016APT29
- * G0091Silence

<https://attack.mitre.org/software/S0195/>

On Linux, **shred** is comparable. (Forensics is the topic of next lecture.)

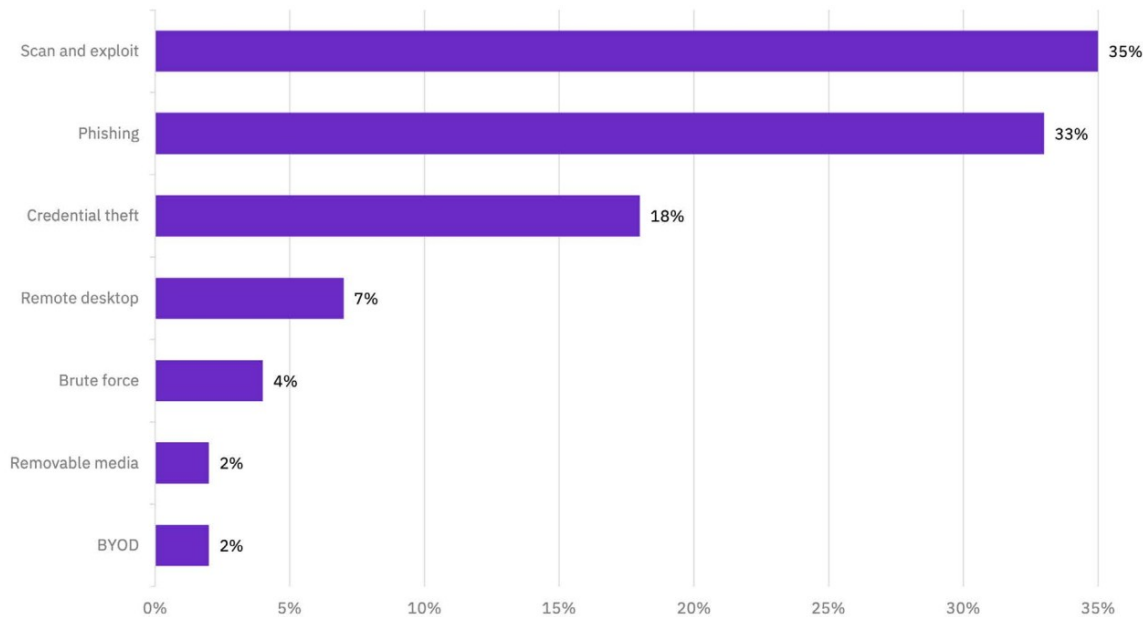
Which ATT&CK Techniques to focus on?

All?

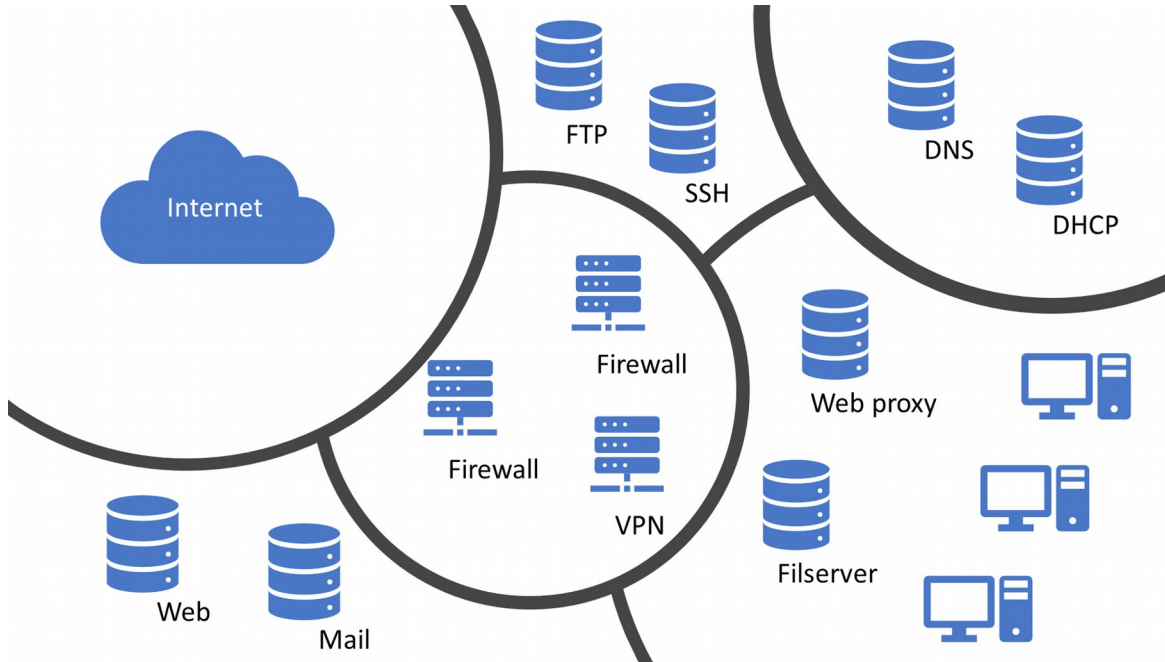
Or, **some**? And if so, then **which**?

Look at the evidence, i.e. **techniques observed in the wild** - either by ourself or reported in freely available information aka **(open source) threat intelligence**.

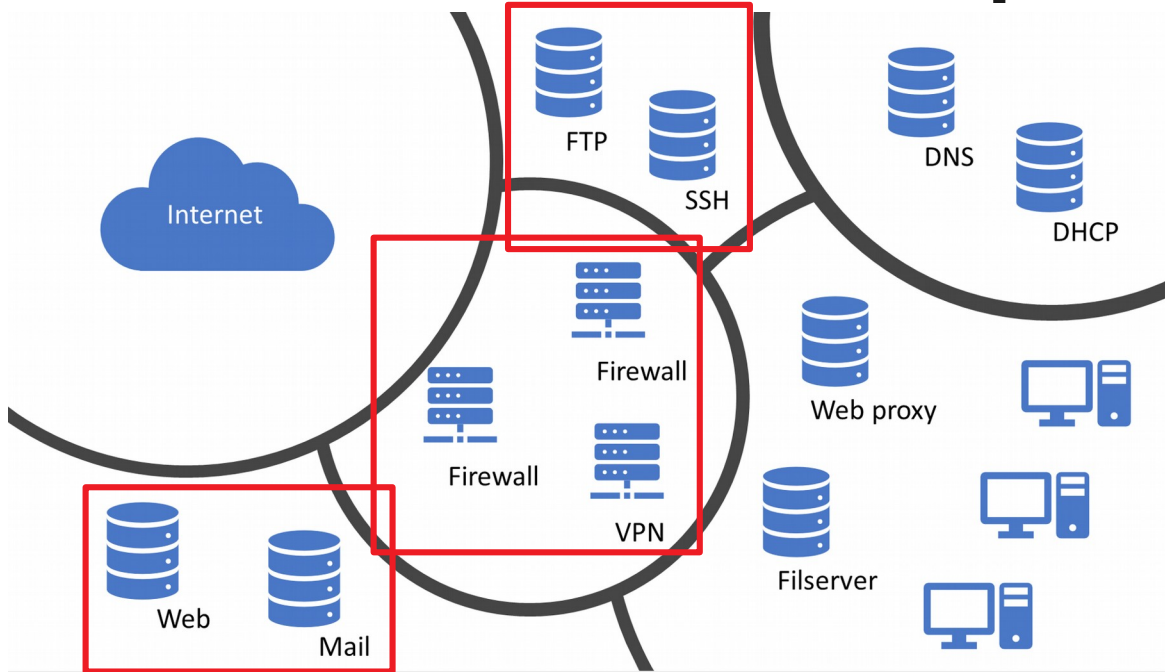
For example, in their 2021 X-Force Threat Intelligence Index, IBM notes their observations on the Initial Access tactic:



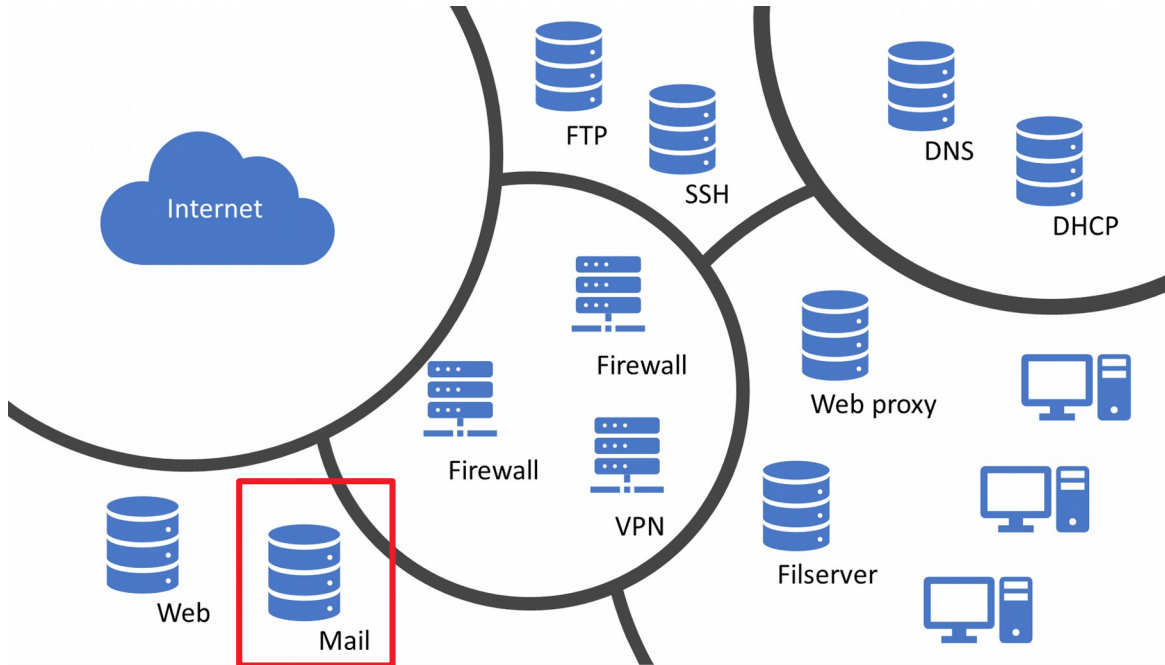
Possible log sources include:



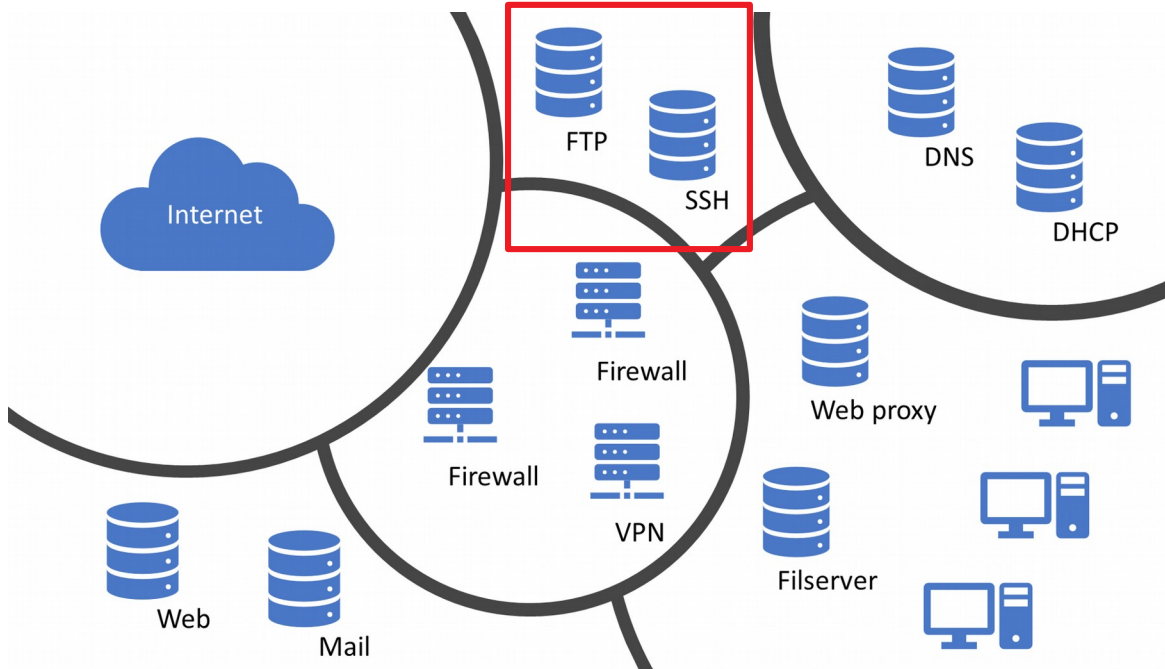
#1 Initial Access: Scan and exploit



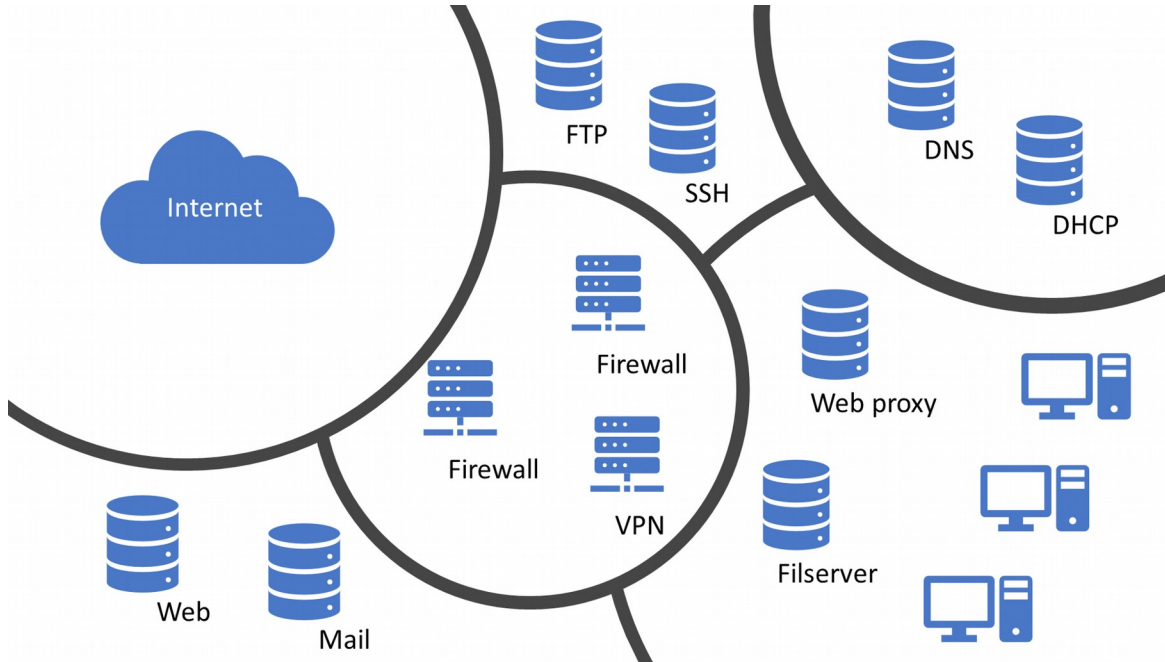
#2 Initial Access: Phishing



#3 Initial Access: Credential theft



Possible log sources include:



Example mail server log

```
2021-09-21T08:29:49.0000000Z,user@company.com,UserLoggedIn,[Details]
2021-09-21T08:31:34.0000000Z,user@company.com,UserLoggedIn,[Details]
2021-09-21T08:31:42.0000000Z,user@company.com,FilePreviewed,[Details]
2021-09-21T08:31:45.0000000Z,user@company.com,UserLoggedIn,[Details]
2021-09-21T08:31:47.0000000Z,user@company.com,UserLoggedIn,[Details]
2021-09-21T08:32:44.0000000Z,user@company.com,UserLoggedIn,[Details]
2021-09-21T08:32:54.0000000Z,user@company.com,UserLoggedIn,[Details]
2021-09-21T08:42:30.0000000Z,user@company.com,Set-Mailbox,[Details]
2021-09-21T08:55:24.0000000Z,user@company.com,New-InboxRule,[Details]
2021-09-21T11:28:39.0000000Z,user@company.com,UserLoggedIn,[Details]
```





Example web server log

```
[Oct 1 12:47:57 2021] 87.118.116.103:46928 [200]: /pressroom.php
[Oct 1 12:47:57 2021] 87.118.116.103:46930 [404]: /favicon.ico - No such file or directory
[Oct 1 12:47:57 2021] Notice: Undefined index: tag in /tmp/php/pressroom.php on line 17
[Oct 1 12:48:05 2021] 87.118.116.103:46932 [200]: /pressroom.php?tag=news
[Oct 1 12:48:14 2021] 87.118.116.103:46934 [200]: /pressroom.php?tag=events
[Oct 1 12:48:14 2021] 87.118.116.103:46936 [200]: /pressroom.php?tag=research
[Oct 1 12:48:18 2021] 87.118.116.103:46938 [200]: /pressroom.php?tag=foo
[Oct 1 12:48:18 2021] Notice: Non-existent tag requested: foo
[Oct 1 12:48:55 2021] 87.118.116.103:46946 [200]: /pressroom.php?tag=error.log
[Oct 1 12:49:10 2021] 87.118.116.103:46950 [200]: /pressroom.php?tag=../../etc/passwd
```

Sidebar: Recent Apache path traversal CVE

critical: Path traversal and file disclosure vulnerability

A flaw was found in a change made to path normalization in Apache 2.4.49. If files outside of these directories are not protected by the usual Alias directives, this could allow for remote code execution.

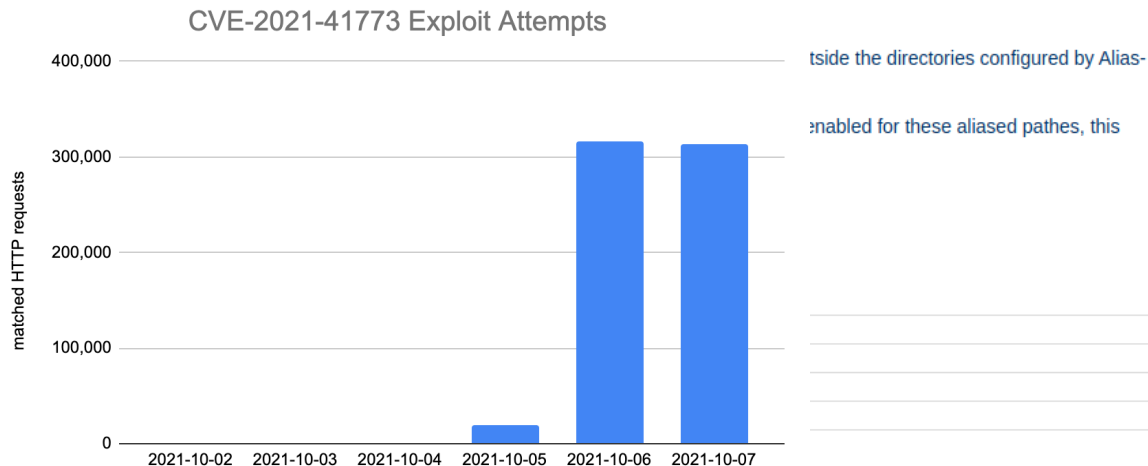
If files outside of these directories are not protected by the usual Alias directives, this could allow for remote code execution.

This issue is known to be exploited in the wild.

This issue only affects Apache 2.4.49 and not earlier versions.

Acknowledgements: This issue was reported by Ash Daulton

Reported to security team	2021-09-29
fixed by r1893775 in 2.4.x	2021-10-01
Update 2.4.50 released	2021-10-04
Affects	2.4.49



`$hostname/cgi-bin/.%2e/%2e%2e/%2e%2e/etc/passwd`



Example DNS log

```
30-09-2021 01:29:55 UDP Rcv 10.232.65.43    Q (3)www(7)gstatic(3)com(0)
30-09-2021 01:29:55 UDP Snd 10.232.65.43   R Q (3)www(7)gstatic(3)com(0)
30-09-2021 01:29:55 UDP Rcv 10.201.120.30   Q (5)login(4)live(3)com(0)
30-09-2021 01:29:55 UDP Snd 10.201.120.30   R Q (5)login(4)live(3)com(0)
30-09-2021 01:29:55 UDP Rcv 10.230.20.106   Q (2)gg(6)google(3)com(0)
30-09-2021 01:29:55 UDP Snd 10.230.20.106   R Q (2)gg(6)google(3)com(0)
30-09-2021 01:29:55 UDP Rcv 10.201.100.45   Q (4)pool(3)ntp(3)org(0)
30-09-2021 01:29:55 UDP Snd 10.201.100.45   R Q (4)pool(3)ntp(3)org(0)
30-09-2021 01:29:55 UDP Rcv 10.201.100.65   Q (5)yahoo(3)com(0)
30-09-2021 01:29:55 UDP Snd 10.201.100.65   R Q (5)yahoo(3)com(0)
```



Example DHCP log

```
10,2021/09/09,08:30:01,Assign,10.201.22.101,WS10012A,8c164566564e
10,2021/09/09,08:33:12,Assign,10.201.22.108,WS10022A,8c1645665a4b,
10,2021/09/09,08:33:55,Assign,10.201.22.109,WS10052A,8c164566779e,
10,2021/09/09,08:34:01,Assign,10.201.22.110,WS10044A,8c164566464c,
11,2021/09/09,08:34:32,Renew,10.201.22.122,VM10081A,005056c00001,
10,2021/09/09,08:34:34,Assign,10.201.22.130,WS10012A,8c16456651aa
11,2021/09/09,08:35:45,Renew,10.201.22.133,VM10110A,005056cee001,
10,2021/09/09,08:35:53,Assign,10.201.22.134,WS10072A,8c16456ab1a4b,
12,2021/09/09,08:37:01,Release,10.201.22.110,WS10048A,8c16456694c,
10,2021/09/09,08:37:10,Assign,10.201.22.110,WS10097A,8c164561239e,
```



Example firewall log

```
Mar  1 11:28:47 Built inbound UDP id 4253 from 192.38.84.35/7179 to 130.226.237.14/53
Mar  1 11:28:47 Teardown TCP id 4198 duration 0:00:00 bytes 7194 TCP FINs from in
Mar  1 11:28:47 Deny TCP from 10.150.96.249/54735 to 130.226.237.153/4433 flags RST ACK
Mar  1 11:28:47 Built inbound UDP id 4254 from 192.38.84.42/61918 to 130.226.237.14/53
Mar  1 11:28:47 Built inbound UDP id 4257 from 10.202.55.102/64651 to 130.226.237.14/53
Mar  1 11:28:47 Built outbound UDP id 4259 from 130.226.142.7/53 to 130.226.237.14/20238
Mar  1 11:28:47 Built inbound UDP id 4258 from 10.202.55.21/53921 to 130.226.237.14/53
Mar  1 11:28:47 Built outbound UDP id 4255 from 130.226.237.173/53 to 130.226.237.14/27800
Mar  1 11:28:47 Teardown TCP id 4210 duration 0:00:00 bytes 0 TCP FINs
Mar  1 11:28:47 Built inbound id 4260 TCP from 10.209.100.121/62921 to 130.226.237.153/4433
```



Example firewall log

```
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=60 syn [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=60 ack syn [Details]
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=52 ack [Details]
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=39 ack psh [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=52 ack [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=67 ack psh [Details]
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=52 ack [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=64 ack psh [Details]
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=52 ack [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=52 ack fin [Details]
```



Example Windows server Security Log

```
Information 10-06-2021 05:00:00 Microsoft Windows security auditing. 4624 Logon
Information 10-06-2021 05:00:00 Microsoft Windows security auditing. 4648 Logon
Information 10-06-2021 04:55:29 Microsoft Windows security auditing. 4625 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4624 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4648 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4673 Sensitive Privilege Use
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4624 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4648 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4673 Sensitive Privilege Use
Information 10-06-2021 04:55:10 Microsoft Windows security auditing. 4673 Sensitive Privilege Use
```



Log analysis



Log analysis approach

The Question

The specific question we are trying to answer

The Understanding

Understand the log

The Pattern

Find a pattern in the log that helps answer the question

The Search

Find all log entries that contain the pattern

The Extract

Output all or some elements of the log entries

The Clustering

If needed, aggregate the output to answer the question

For example, if we want to find all successful logins to our mail server (**the question**), and we know that the corresponding mail server log entry looks like this (**the understanding**):

```
2021-09-21T08:29:49.00Z,user@company.com,UserLoggedIn,[Details]
```

Then we need to search for “UserLoggedIn” (**the pattern**) to find all entries relevant to answer our question.



Find the pattern

Logs are **different**.

Patterns for successful logins in other logs will look different.

Which patterns should we search for:

FTP: Sep 30 13:09:52 - ftpuser (80.62.115.191) ! Successfully logged in.

SSH: Sep 30 13:31:07 Accepted password for root from 62.44.128.103 port 35901 ssh2

DHCP: Sep 30 13:53:00 User REGH IP 80.62.115.191 IPv4 192.168.200.103 assigned to session



Find the pattern

Logs are **different**.

Patterns for successful logins in other logs will look different.

Which patterns should we search for:

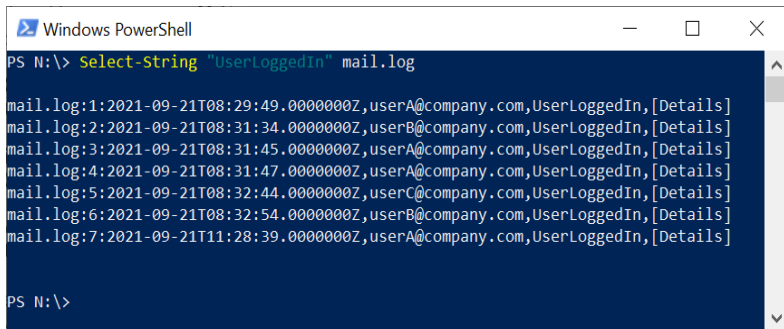
FTP: Sep 30 13:09:52 - ftpuser (80.62.115.191) ! **Successfully logged in.**

SSH: Sep 30 13:31:07 **Accepted password** for root from 62.44.128.103 port 35901 ssh2

DHCP: Sep 30 13:53:00 User REGH IP 80.62.115.191 IPv4 192.168.200.103 **assigned to session**

The Search

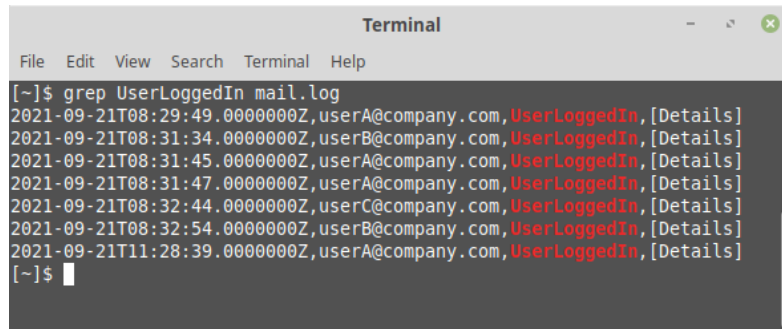
Given a pattern, we can use **custom-built tools** (like SIEMs), **write our own tools** using more or less well-chosen string matching algorithms or use **command-line shells**.



```
Windows PowerShell
PS N:\> Select-String "UserLoggedIn" mail.log

mail.log:1:2021-09-21T08:29:49.000000Z,userA@company.com,UserLoggedIn,[Details]
mail.log:2:2021-09-21T08:31:34.000000Z,userB@company.com,UserLoggedIn,[Details]
mail.log:3:2021-09-21T08:31:45.000000Z,userA@company.com,UserLoggedIn,[Details]
mail.log:4:2021-09-21T08:31:47.000000Z,userA@company.com,UserLoggedIn,[Details]
mail.log:5:2021-09-21T08:32:44.000000Z,userC@company.com,UserLoggedIn,[Details]
mail.log:6:2021-09-21T08:32:54.000000Z,userB@company.com,UserLoggedIn,[Details]
mail.log:7:2021-09-21T11:28:39.000000Z,userA@company.com,UserLoggedIn,[Details]

PS N:\>
```



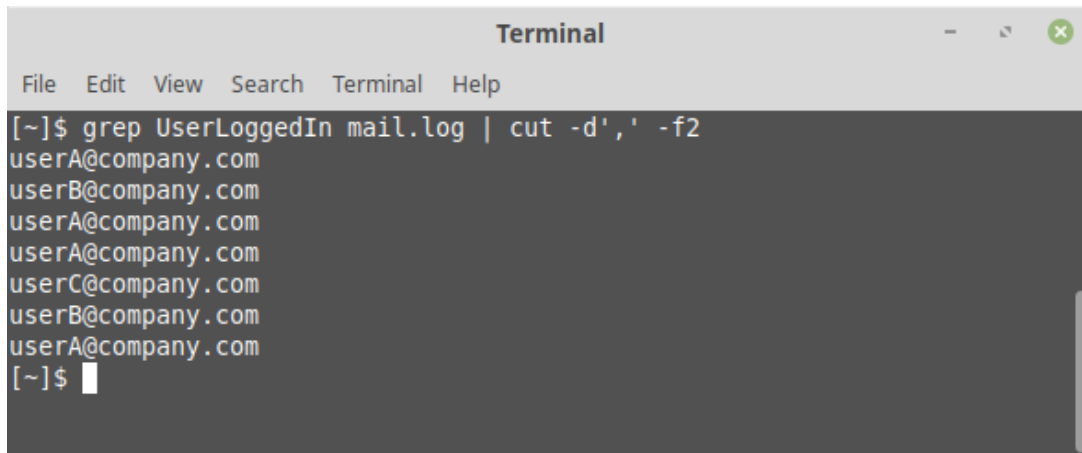
```
Terminal
File Edit View Search Terminal Help

[~]$ grep UserLoggedIn mail.log
2021-09-21T08:29:49.000000Z,userA@company.com,UserLoggedIn,[Details]
2021-09-21T08:31:34.000000Z,userB@company.com,UserLoggedIn,[Details]
2021-09-21T08:31:45.000000Z,userA@company.com,UserLoggedIn,[Details]
2021-09-21T08:31:47.000000Z,userA@company.com,UserLoggedIn,[Details]
2021-09-21T08:32:44.000000Z,userC@company.com,UserLoggedIn,[Details]
2021-09-21T08:32:54.000000Z,userB@company.com,UserLoggedIn,[Details]
2021-09-21T11:28:39.000000Z,userA@company.com,UserLoggedIn,[Details]
[~]$
```

The Extract

Suppose we want to zoom in on **who logged on**.

Then, in the **terminal**, we can use **cut** to output the second column (**-f2**) delimited by comma (**-d','**):


A terminal window titled "Terminal" with a menu bar (File, Edit, View, Search, Terminal, Help). The command `[~]$ grep UserLoggedIn mail.log | cut -d',' -f2` is entered. The output shows a list of email addresses: `userA@company.com`, `userB@company.com`, `userA@company.com`, `userA@company.com`, `userC@company.com`, `userB@company.com`, and `userA@company.com`. The prompt `[~]$` is shown at the bottom with a cursor.

```
Terminal
File Edit View Search Terminal Help
[~]$ grep UserLoggedIn mail.log | cut -d',' -f2
userA@company.com
userB@company.com
userA@company.com
userA@company.com
userC@company.com
userB@company.com
userA@company.com
[~]$
```

The Clustering

Suppose we want to count **how many times** each user logged on.

Then, in the **terminal**, we can **sort** the output of cut and use **uniq (-c)** to occurrence of each user:

A terminal window titled "Terminal" with a menu bar (File, Edit, View, Search, Terminal, Help). The command `grep UserLoggedIn mail.log | cut -d',' -f2 | sort | uniq -c` is entered and executed. The output shows the count of logins for three users: 4 for userA@company.com, 2 for userB@company.com, and 1 for userC@company.com. The prompt `[~]$` is visible at the bottom.

```
Terminal
File Edit View Search Terminal Help
[~]$ grep UserLoggedIn mail.log | cut -d',' -f2 | sort | uniq -c
    4 userA@company.com
    2 userB@company.com
    1 userC@company.com
[~]$
```



Log analysis of an FTP log

```
G:\homedir\sap-ftp\monitoring_status.data|156|2021-08-22 11:50:06|sap-ftp(10.209.131.12)
G:\homedir\sap-ftp\monitoring_status.data|156|2021-08-22 12:20:02|sap-ftp(10.209.131.12)
G:\homedir\backup\export_20210821.csv|234135|2021-08-22 12:50:01|backup(10.209.131.12)
G:\homedir\netops\zone12.txt|146|2021-08-22 13:20:01|netops(10.209.131.54)
G:\homedir\srv-pki\CA01.crl|520|2021-08-22 13:50:01|srv-pki(10.209.131.72)
G:\homedir\sap-ftp\monitoring_status.data|146|2021-08-22 14:20:03|sap-ftp(10.209.131.12)
G:\homedir\sql-ftp\md5_checksum.txt|36|2021-08-22 14:50:01|sql-ftp(10.209.131.12)
G:\homedir\netops\zone12.txt|146|2021-08-22 15:20:01|netops(10.209.131.54)
G:\homedir\vmware\postgres_init.gz|12908|2021-08-22 15:50:01|vmware(10.209.131.12)
G:\homedir\vmware\db_backup.gz|74448897|2021-08-22 15:50:01|vmware(10.209.131.12)
```

With **grep** and **cut**, build a command that searches for and extracts **time stamps** for to the **vmware** user:

```
grep vmware ftp.log | cut -d'|' -f3
```



Log analysis of a firewall log

Demo.



Indicators of compromise in Intrusion Detection



Indicators of compromise (IOCs)

Technical characteristics that identify a known threat, attacker methodology, or other evidence of compromise, e.g.:

- C2 domains

- IPs used in attack

- Special GET requests

- Malware file system locations

- Malware hashes

- Memory artifacts

Duqu IOCs

W32.Flamer

VS W32.Stuxnet and W32.Duqu

A quick comparison of the three threats.

All three threats appear to be developed by teams of attackers, rather than a lone individual.



All three threats were discovered within the Middle East

The code base behind Stuxnet and Duqu are similar.



Stuxnet



Duqu

The code base from Flamer is different from the other two.



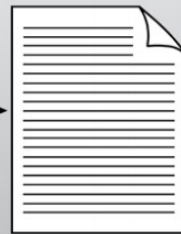
Flamer



All three threats were advanced persistent threats that targeted industrial or government systems.

The file size of Flamer is significantly larger than either Stuxnet or Duqu.

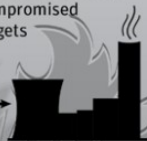
Stuxnet Duqu



The purpose of both Flamer and Duqu appear to be to gather information from the compromised computer. In contrast, Stuxnet targets industrial control systems.



Stuxnet



Flamer



Duqu



Duqu IOCs

W32.Flamer

VS W32.Stuxnet and W32.Duqu

A quick comparison of the three threats.

All three threats appear to be developed by teams of attackers, rather than a lone



The code base behind Stuxnet and Duqu



All three threats were advanced persistent threats that targeted industrial or government systems.

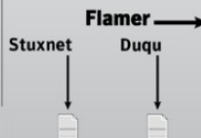
```
https://securelist.com/files/2015/06/7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f.ioc - Opera
Menu https://securelist.com/file X +
securelist.com/files/2015/06/7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f.ioc
<ioc xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.mandiant.com/2010/ioc"
id="7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f" last-modified="2015-06-10T11:48:29">
<short_description>TheDuqu 2.0 IOCs</short_description>
<description>
Indicators of compromise for the Duqu 2.0 https://securelist.com/blog/research/70504/the-mystery-
of-duqu-2-0-a-sophisticated-cyberespionage-actor-returns/
</description>
<authored_by>Kaspersky Lab</authored_by>
<authored_date>2015-06-09T21:47:32</authored_date>
<links/>
<definition>
<Indicator operator="OR" id="ad9e4858-9a36-4bf3-822f-04aad37e4887">
<IndicatorItem id="aa142b0a-c795-4a01-ad86-a938910091ea" condition="is">
<Context document="FileItem" search="FileItem/Md5sum" type="md5">
<Content type="md5">089a14f69a31ea5e9a5b375dc0c46e45</Content>
</IndicatorItem>
<IndicatorItem id="87853206-5a78-4260-a4ac-2a9b1e82c1f3" condition="is">
<Context document="FileItem" search="FileItem/Md5sum" type="md5">
<Content type="md5">16ed790940a701c813e0943b5a27c6c1</Content>
</IndicatorItem>
<IndicatorItem id="7fe336c9-c70c-43c1-a6c9-dce88dae9c40" condition="is">
<Context document="FileItem" search="FileItem/Md5sum" type="md5">
<Content type="md5">26c48a03a5f3218b4a10f2d3d9420b97</Content>
</IndicatorItem>
```



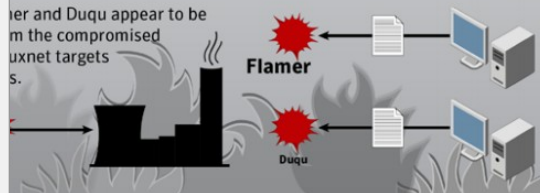
ner is different two.



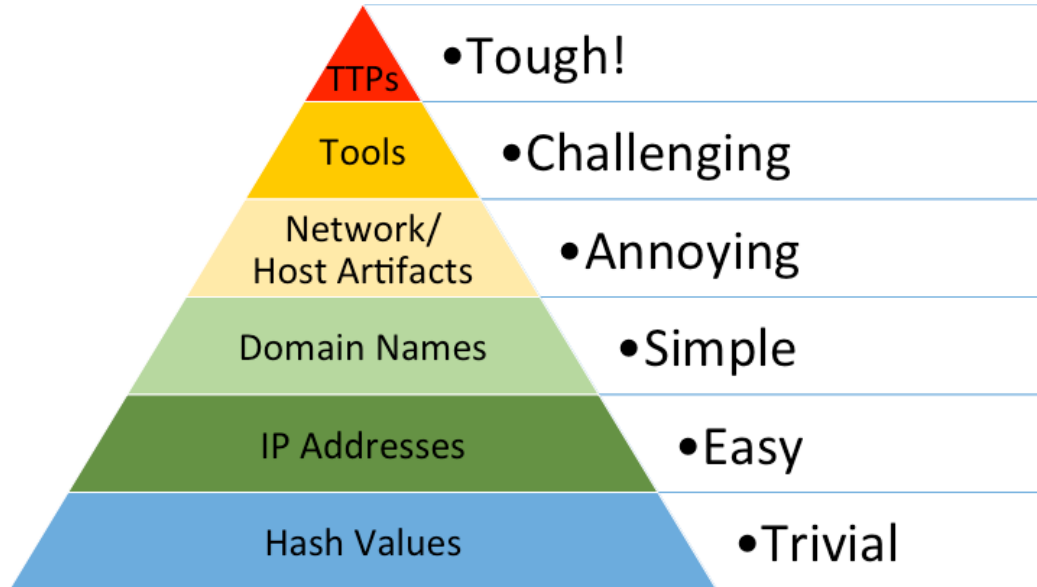
The file size of Flamer is significantly larger than either Stuxnet or Duqu.




ner and Duqu appear to be m the compromised uxnet targets s.



IOCs and “The Pyramid of Pain”





IOC (hash) strategy

Collect IOC hashes

For each host in my network:

- Calculate file hashes

- Match against IOC list

Problems:

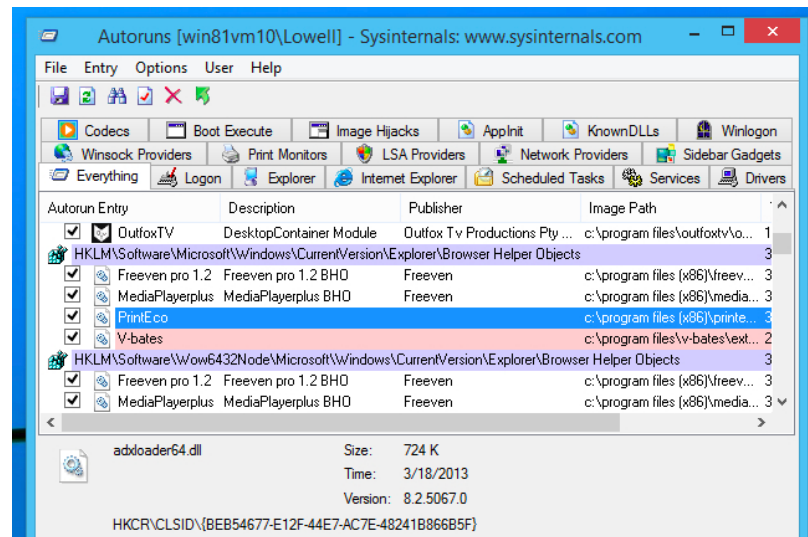
- What if attacker updates the malware?

- What if we get a match = IOC fidelity

Refined approach

Same as before but instead of all files, calculate only for executables that **autostart**

Plus: Look for new entries or hosts with entries unlike most, i.e. **anomalies** instead of IOCs only



Virustotal and its API

```
$ cat hashes
```

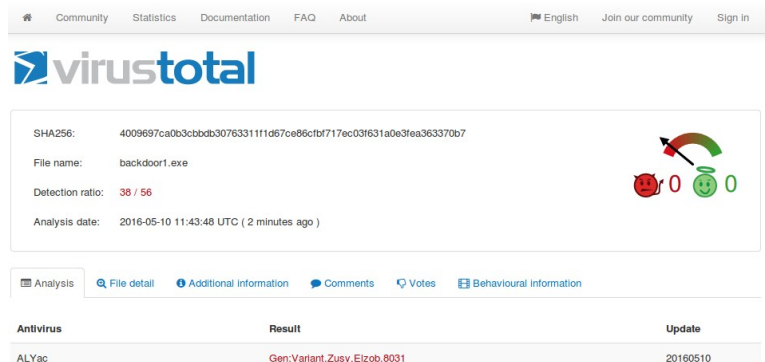
```
DC1E56092CC57FB4605B088D3DCCBF7A  
6F8842584D868174E24CACFD18B366B5  
0DA1C970D9AA3CCCCFBA7EE90876CBAB
```

```
$ cat vt.py
```

```
import requests  
import sys  
import time  
with open(sys.argv[1], 'r') as fd:  
    for line in fd.readlines():  
        params = {'apikey': 'key', 'resource': line.rstrip()}  
        response = requests.get('https://www.virustotal.com/vtapi/v2/file/report', params=params)  
        response_json = response.json()  
        print line.rstrip(), response_json['positives'], response_json['total']
```

```
$ python vt.py hashes
```

```
DC1E56092CC57FB4605B088D3DCCBF7A 0 66  
6F8842584D868174E24CACFD18B366B5 0 68  
0DA1C970D9AA3CCCCFBA7EE90876CBAB 26 57
```



The screenshot displays the VirusTotal website's file analysis page. At the top, there is a navigation bar with links for Community, Statistics, Documentation, FAQ, and About, along with language and login options. The main header features the VirusTotal logo. The central content area shows the following details for the file 'backdoor1.exe':
- SHA256: 4009697ca0b3cbbdb307633111d67ce86c6bf717ec03f631a0e3ea363370b7
- File name: backdoor1.exe
- Detection ratio: 38 / 56 (indicated by a red and green circular progress bar)
- Analysis date: 2016-05-10 11:43:48 UTC (2 minutes ago)
Below this information is a tabbed interface with 'Analysis' selected. The analysis results are presented in a table with three columns: Antivirus, Result, and Update.

Antivirus	Result	Update
ALYac	Gen:Variant.Zusy.Elzob.8031	20160510

In addition, submit file for analysis

Using malware sandboxes (automated malware analysis) - data points for machine learning

Detected signatures	
i	The executable contains unknown PE section names indicative of a packer (could be a false positive) 1 event
i	The file contains an unknown PE resource name possibly indicative of a packer 1 event
!	Performs some HTTP requests 21 events
!	Allocates read-write-execute memory (usually to unpack itself) 1 event
⊗	Communicates with host for which no DNS query was performed 1 event
⊗	Connects to an IP address that is no longer responding to requests (legitimate services will remain up-and-running usually) 1 event
⊗	File has been identified by 39 AntiVirus engines on VirusTotal as malicious 39 events



Wrap-up