

Threat Modelling

Acknowledgements: Ivan Fléchais, Shamal Faily

Outline

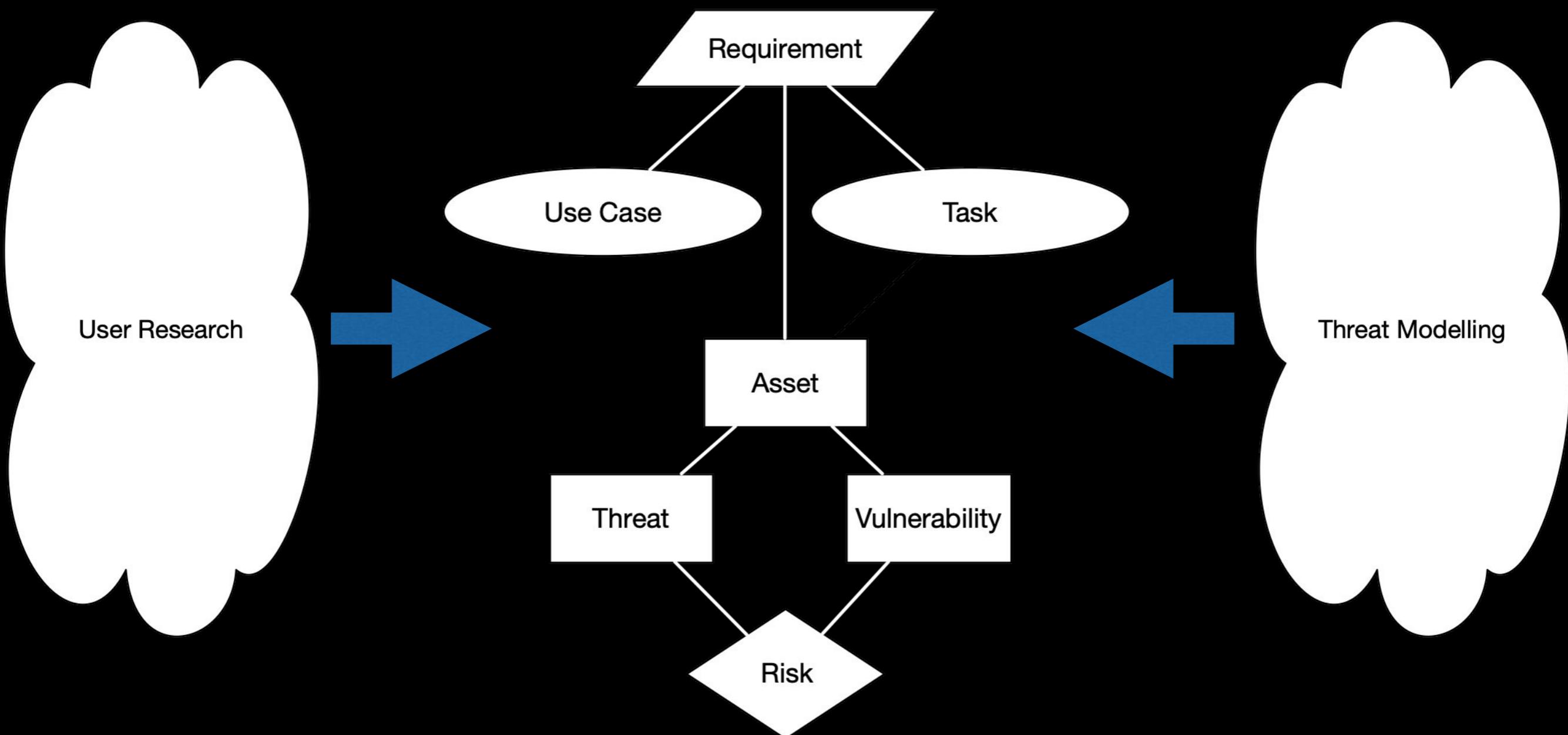
1. Threat Modelling Frameworks

- STRIDE
- The Diamond Model
- Other prominent frameworks

2. Threat Libraries

- OWASP
- CAPEC
- MITRE ATT&CK
- ATT&CK

Threat Modelling in context



What is a Threat Model

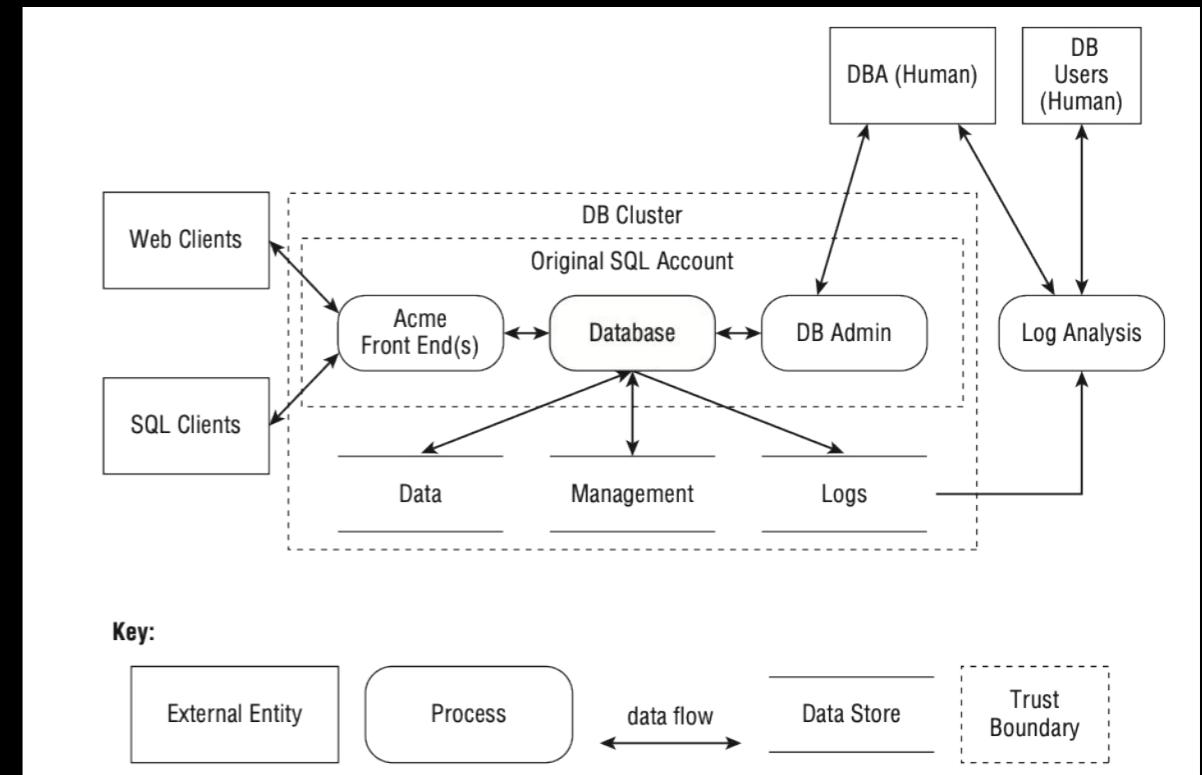
For a given system, the enumeration of the credible attack vectors, their impacts, and their mitigations.

Threat modelling requires:

- a knowledge of how systems are configured and deployed
- A broad understanding of attack methods
- Example techniques: attack patterns, attack trees, attacker personas

Strategy for threat modelling

- Model the system you're building
- Find threats using the model, and with complementary approaches
- Address threats
- Validate models for completeness and effectiveness



Approaches for finding threats

- STRIDE
- Attack trees
- Attack libraries

STRIDE

- Designed to help software developers identify typical attacks against software
 - S poofing
 - T ampering
 - R eputation
 - I nformation Disclosure
 - D enial of Service
 - E levation of Privilege

STRIDE

Threat	Property Violated
Spoofing	Authentication
Tampering	Integrity
Repudiation	Non-repudiation
Information Disclosure	Confidentiality
Denial of Service	Availability
Elevation of Privilege	Authorisation

Spoofing

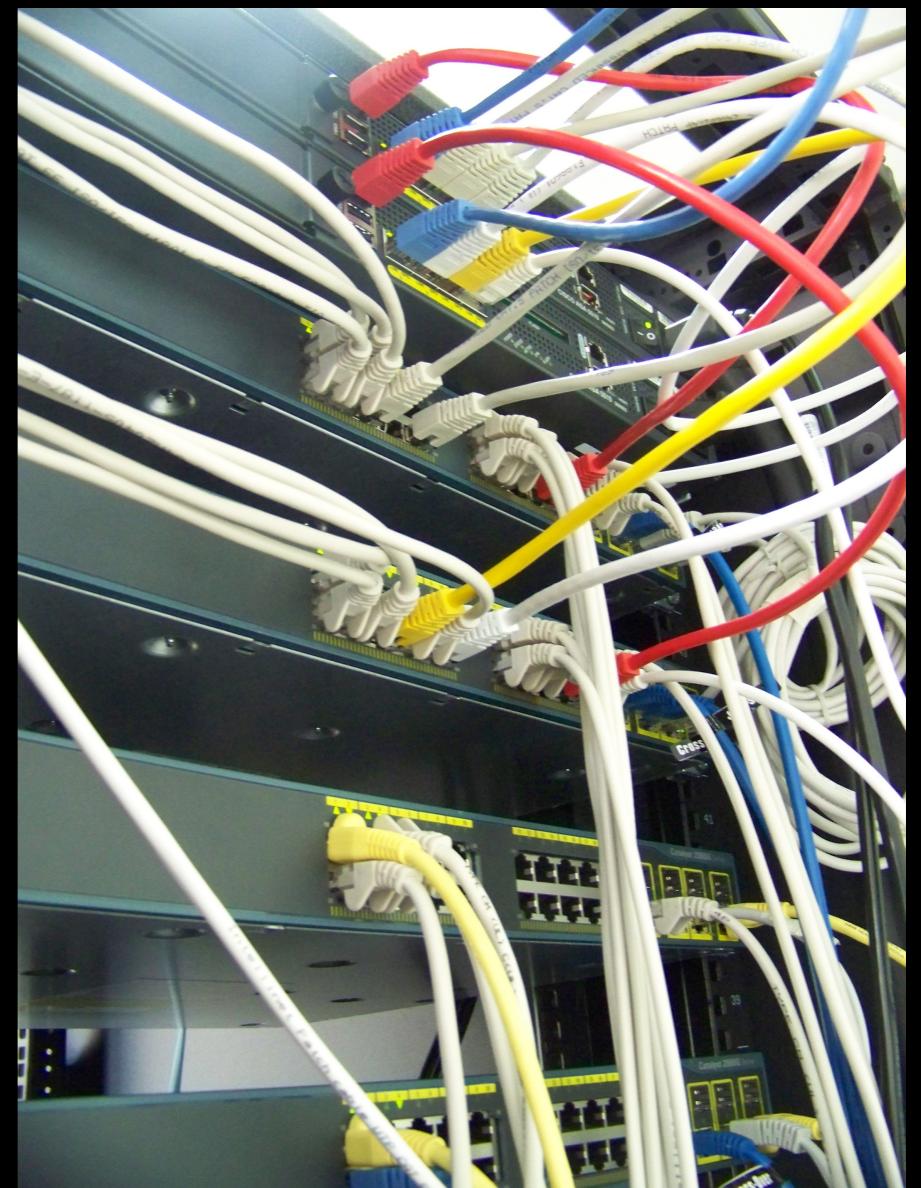
- Pretending to be something or someone other than yourself.
 - Spoofing **processes or files**
 - Spoofing **machines**
 - Spoofing **people**



<https://www.flickr.com/photos/stitchlily/2735596261>

Tampering

- Modification of system resources
 - Tampering with files
 - Tampering with memory
 - Tampering with network



<https://www.flickr.com/photos/airforceone/3035915842>

Repudiation

- Claiming you didn't do or are not responsible for something.
 - Repudiating actions
 - Attacking the logs



Photo by Tim Gouw on Unsplash

Information Disclosure

- Allow people to see information they are not authorised to see.
 - From processes
 - From data stores
 - From data flows



Photo by Daan Mooij on Unsplash

Denial of Service

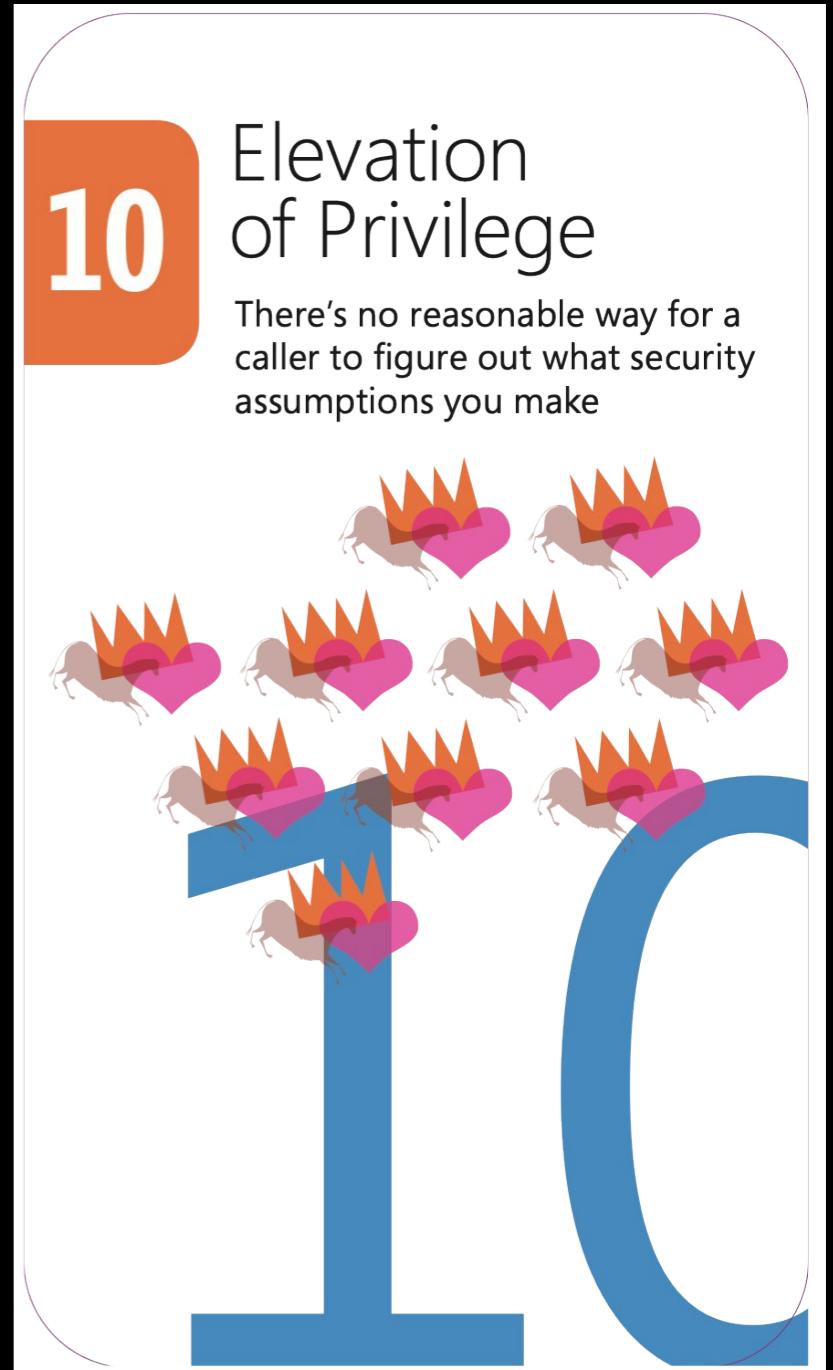
- Absorbing resources needed to provide services
 - Against processes
 - Against data stores
 - Against data flows



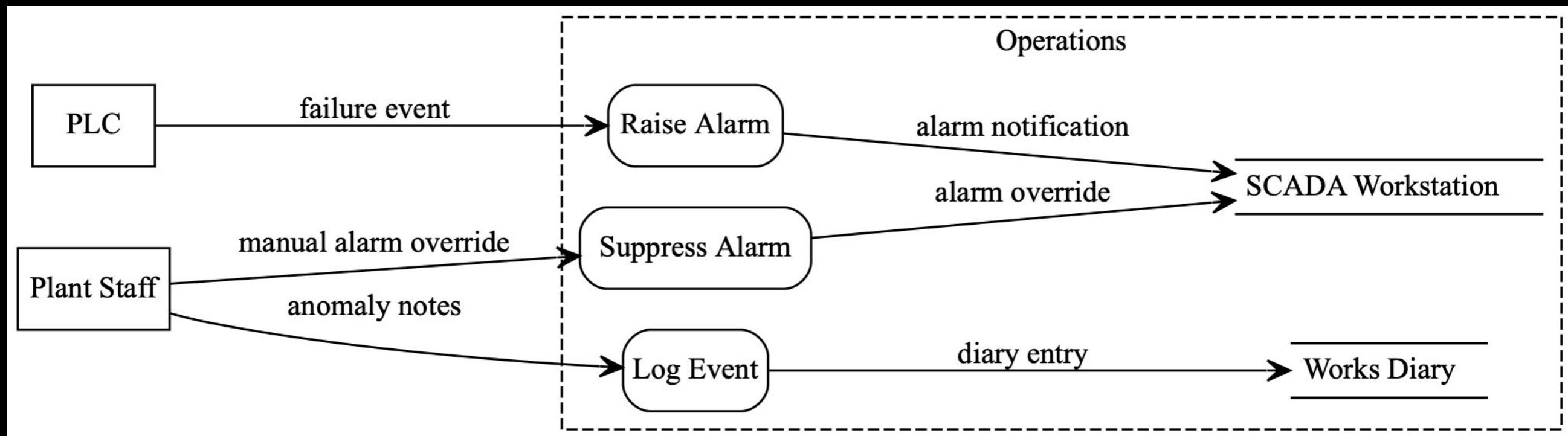
Photo by Markus Spiske on Unsplash

Elevation of Privilege

- Allowing someone to do something they're not authorised to do.
 - Corrupting processes
 - Bypass authorisation checks



How to use STRIDE

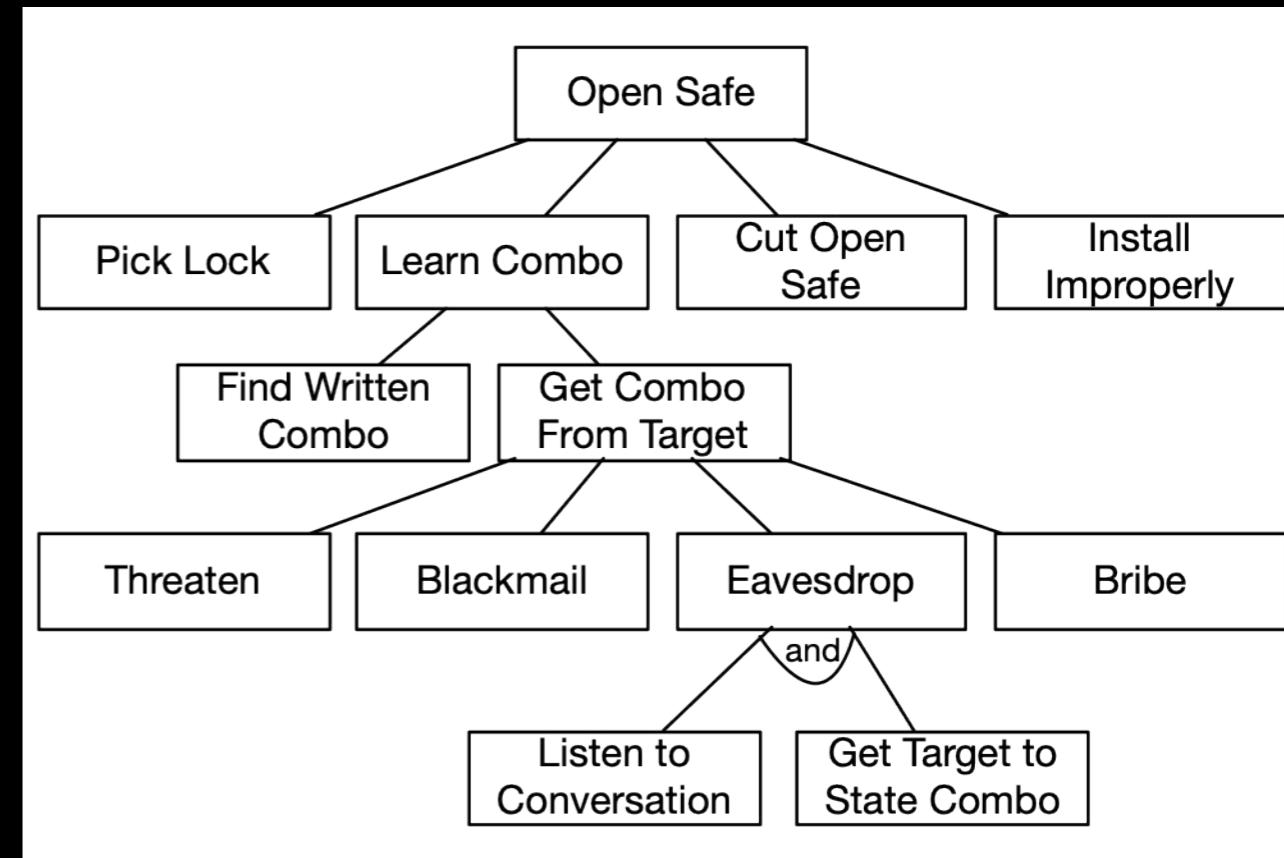


- Review your data flow model and consider
 - Threats to entities, processes, and data stores
 - Threats to data flows

Add slide explaining the components of a DfD

STRIDE and Attack Trees

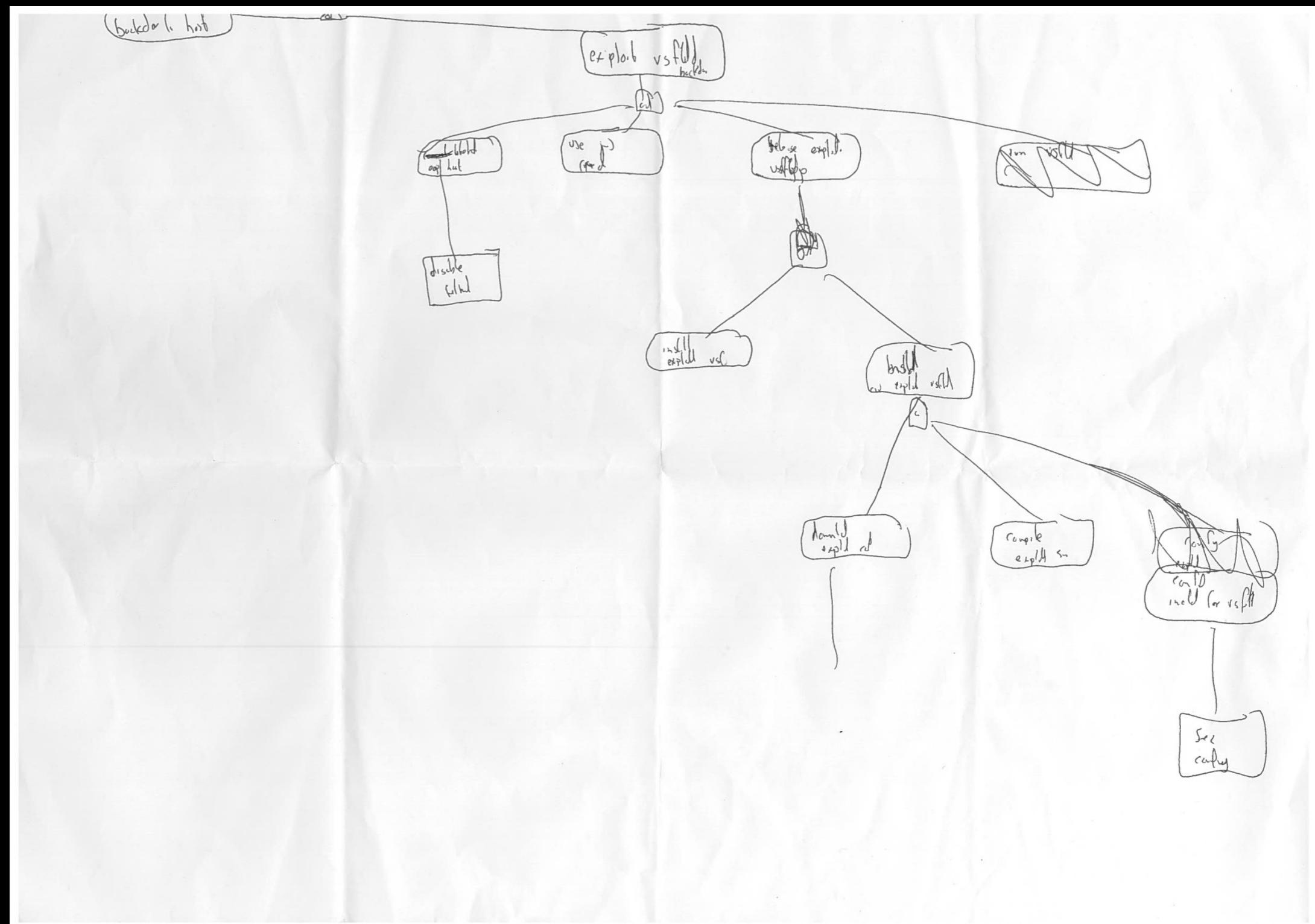
- STRIDE threats might be quite abstract
- Additional research needed to identify root cause vulnerabilities and attacks
- Attack trees can drive and visualise this research.



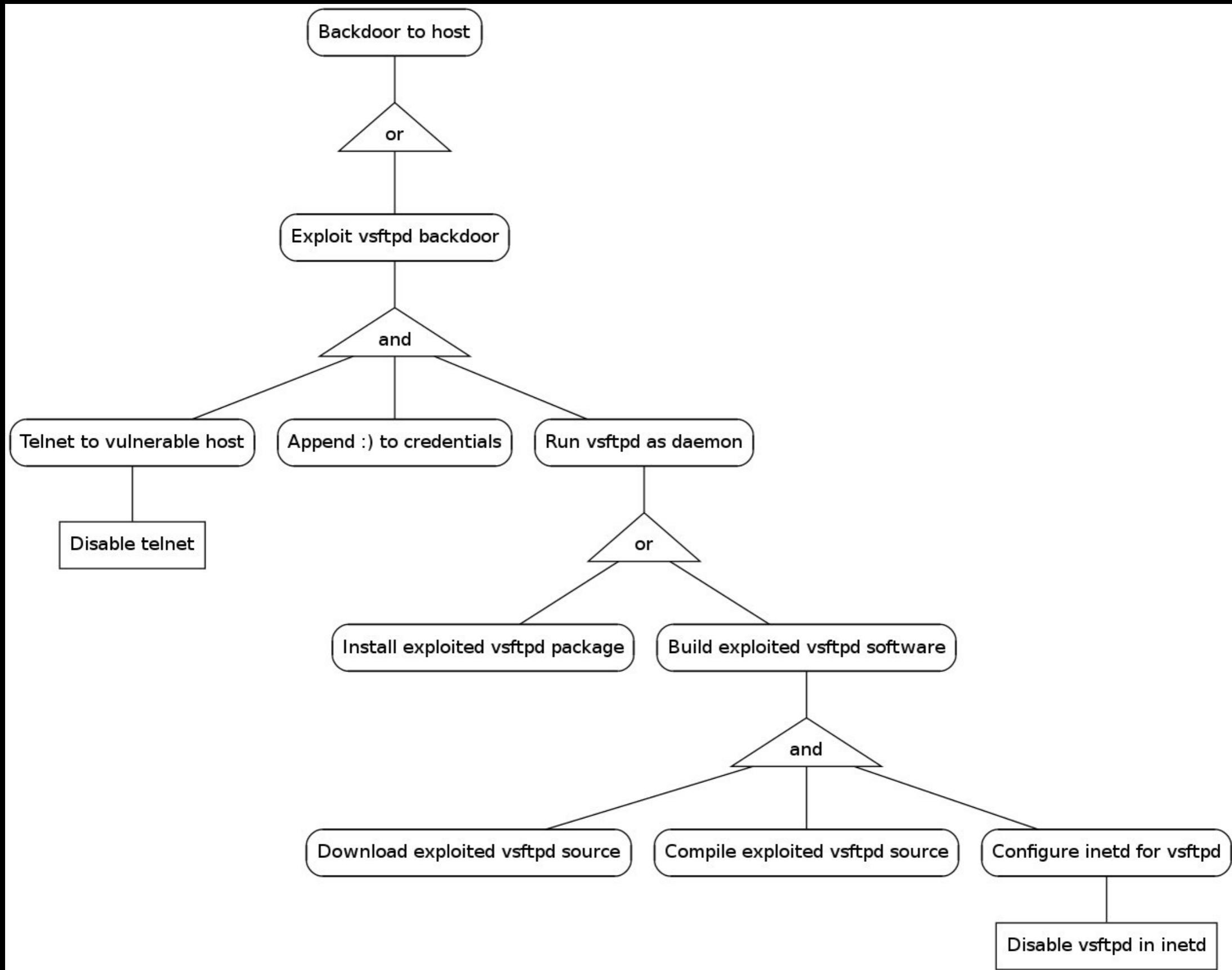
Attack Trees

- Represents attacks using a tree structure
 - Root node is the **goal**
 - Leaf nodes are different **ways of achieving** that **goal**
 - Nodes might be satisfied by AND or OR goals
 - Nodes can be added to indicate countermeasures to attack goals.

Attack Tree sketch



Attack Tree: Graphviz



Host Discovery

```
root@kali:~/Desktop# netdiscover -i eth0 -r 10.0.0.1/24
```

```
Currently scanning: Finished! | Screen View: Unique Hosts
```

```
4 Captured ARP Req/Rep packets, from 4 hosts. Total size: 240
```

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
<hr/>				
10.0.0.1	52:54:00:12:35:00	1	60	Unknown vendor
10.0.0.2	52:54:00:12:35:00	1	60	Unknown vendor
10.0.0.3	08:00:27:99:22:3c	1	60	PCS Systemtechnik GmbH
10.0.0.8	08:00:27:df:24:5a	1	60	PCS Systemtechnik GmbH

```
root@kali:~/Desktop#
```

Stealth Scan ALL Ports

```
root@kali:~/Desktop# nmap -sS -A -T4 -p- 10.0.0.8 -oN metasploitable2_nmap.txt
```

```
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-16 20:59 EDT
Nmap scan report for 10.0.0.8
Host is up (0.00035s latency).
Not shown: 65505 closed ports
PORT      STATE SERVICE      VERSION
21/tcp      open  ftp          vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
|     FTP server status:
|       Connected to 10.0.0.4
|       Logged in as ftp
|       TYPE: ASCII
|       No session bandwidth limit
|       Session timeout in seconds is 300
|       Control connection is plain text
|       Data connections will be plain text
|       vsFTPD 2.3.4 - secure, fast, stable
|_End of status
22/tcp      open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|   2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp      open  telnet        Linux telnetd
25/tcp      open  smtp         Postfix smptd
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES,
|_ssl-date: 2020-07-17T01:01:53+00:00; +2s from scanner time.
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
```

FTP Enumeration

```
root@kali:~# ls -al /usr/share/nmap/scripts/ | grep -e "ftp-"  
-rw-r--r-- 1 root root 4530 Mar 10 12:52 ftp-anon.nse  
-rw-r--r-- 1 root root 3253 Mar 10 12:52 ftp-bounce.nse  
-rw-r--r-- 1 root root 3108 Mar 10 12:52 ftp-brute.nse  
-rw-r--r-- 1 root root 3272 Mar 10 12:52 ftp-libopie.nse  
-rw-r--r-- 1 root root 3290 Mar 10 12:52 ftp-proftpd-backdoor.nse  
-rw-r--r-- 1 root root 3768 Mar 10 12:52 ftp-syst.nse  
-rw-r--r-- 1 root root 6021 Mar 10 12:52 ftp-vsftpd-backdoor.nse  
-rw-r--r-- 1 root root 5923 Mar 10 12:52 ftp-vuln-cve2010-4221.nse  
-rw-r--r-- 1 root root 5736 Mar 10 12:52 tftp-enum.nse  
root@kali:~# █
```

FTP Enumeration

```
root@kali:~# nmap -sV -p 21 --script ftp-vsftpd-backdoor 10.0.0.8
```

```
root@kali:~# nmap -sV -p 21 --script ftp-vsftpd-backdoor 10.0.0.8
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-16 21:10 EDT
Nmap scan report for 10.0.0.8
Host is up (0.00025s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.3.4
|  ftp-vsftpd-backdoor:|
|    VULNERABLE:
|      vsFTPD version 2.3.4 backdoor
|        State: VULNERABLE (Exploitable)
|        IDs: BID:48539  CVE:CVE-2011-2523
|          vsFTPD version 2.3.4 backdoor, this was reported on 2011-07-04.
|        Disclosure date: 2011-07-03
|        Exploit results:
|          Shell command: id
|          Results: uid=0(root) gid=0(root)
|        References:
|          https://www.securityfocus.com/bid/48539
|          http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
|          https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
|          https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/unix/ftp/vsftpd_234_backdoor.rb
MAC Address: 08:00:27:DF:24:5A (Oracle VirtualBox virtual NIC)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.71 seconds
root@kali:~#
```

FTP Enumeration

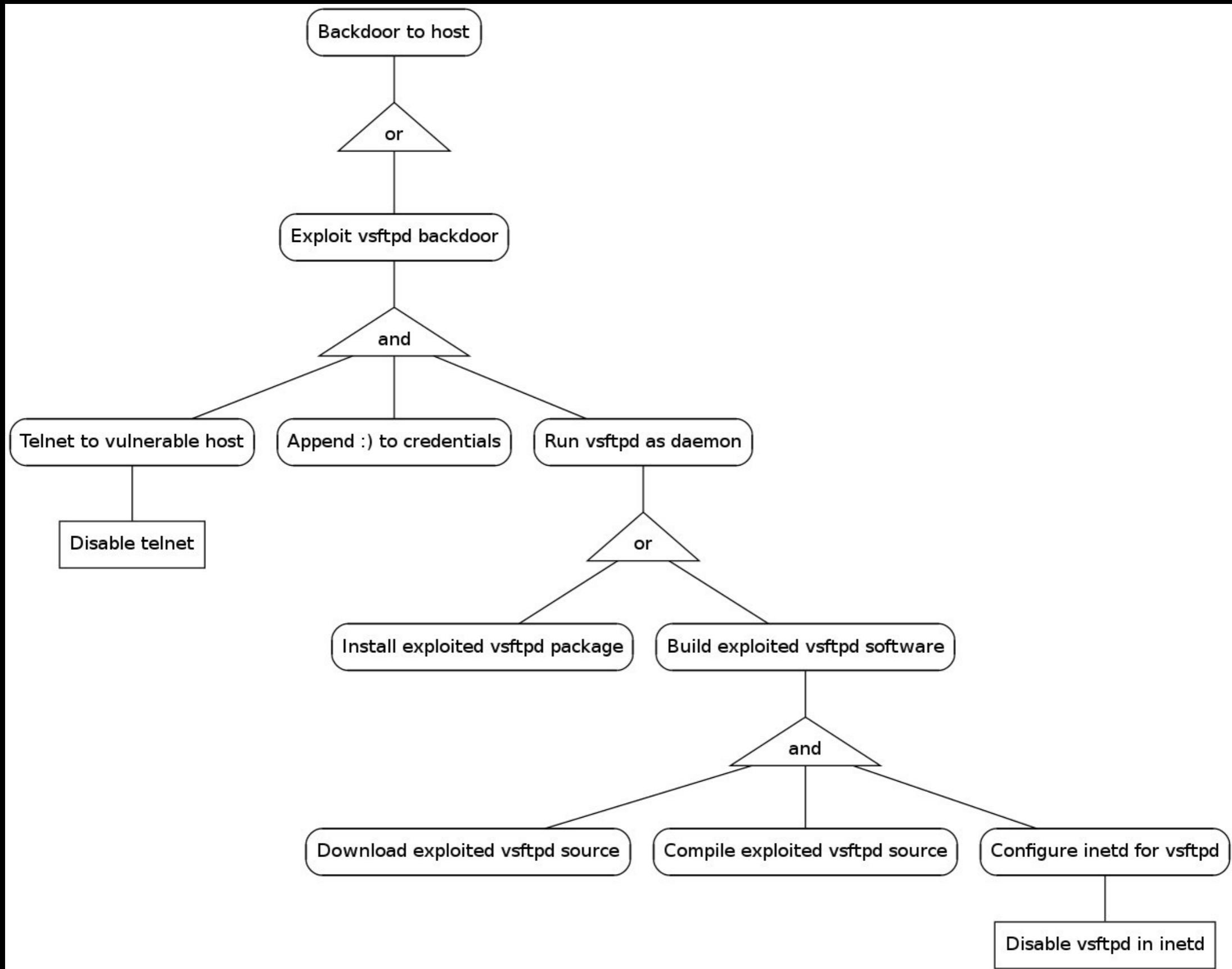
```
root@kali:~# nmap -sV -p 21 --script ftp-vsftpd-backdoor 10.0.0.8
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-16 21:10 EDT
Nmap scan report for 10.0.0.8
Host is up (0.00025s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.3.4
| ftp-vsftpd-backdoor:
|_ VULNERABLE:
    vsFTPD version 2.3.4 backdoor
        State: VULNERABLE (Exploitable)
        IDs: BID:48539  CVE:CVE-2011-2523
            vsFTPD version 2.3.4 backdoor, this was reported on 2011-07-04.
        Disclosure date: 2011-07-03
        Exploit results:
            Shell command: id
            Results: uid=0(root) gid=0(root)
        References:
            https://www.securityfocus.com/bid/48539
            http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
            https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
            https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/unix/ftp/vsftpd_234_backdoor.rb
MAC Address: 08:00:27:DF:24:5A (Oracle VirtualBox virtual NIC)
Service Info: OS: Unix
```

```
root@kali:~# nmap -sV -p 21 --script ftp-vsftpd-backdoor 10.0.0.8
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-16 21:10 EDT
Nmap scan report for 10.0.0.8
Host is up (0.00025s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.3.4
| ftp-vsftpd-backdoor:
|_ VULNERABLE:
    vsFTPD version 2.3.4 backdoor
        State: VULNERABLE (Exploitable)
        IDs: BID:48539  CVE:CVE-2011-2523
            vsFTPD version 2.3.4 backdoor, this was reported on 2011-07-04.
        Disclosure date: 2011-07-03
        Exploit results:
            Shell command: id
            Results: uid=0(root) gid=0(root) !
        References:
            https://www.securityfocus.com/bid/48539
            http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
            https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
            https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/unix/ftp/vsftpd_234_backdoor.rb
MAC Address: 08:00:27:DF:24:5A (Oracle VirtualBox virtual NIC)
Service Info: OS: Unix
```

Attack Tree: Graphviz



Issues with Attack Trees

- Great for **conveying information**, but hard to create.
- Without the right **root nodes**, you can **miss entire attack groupings**.
- Many attack trees will **yield out of scope issues**.
- No **consistency** around attack tree **semantics**, e.g. AND/OR nodes, sequences.

Attack libraries

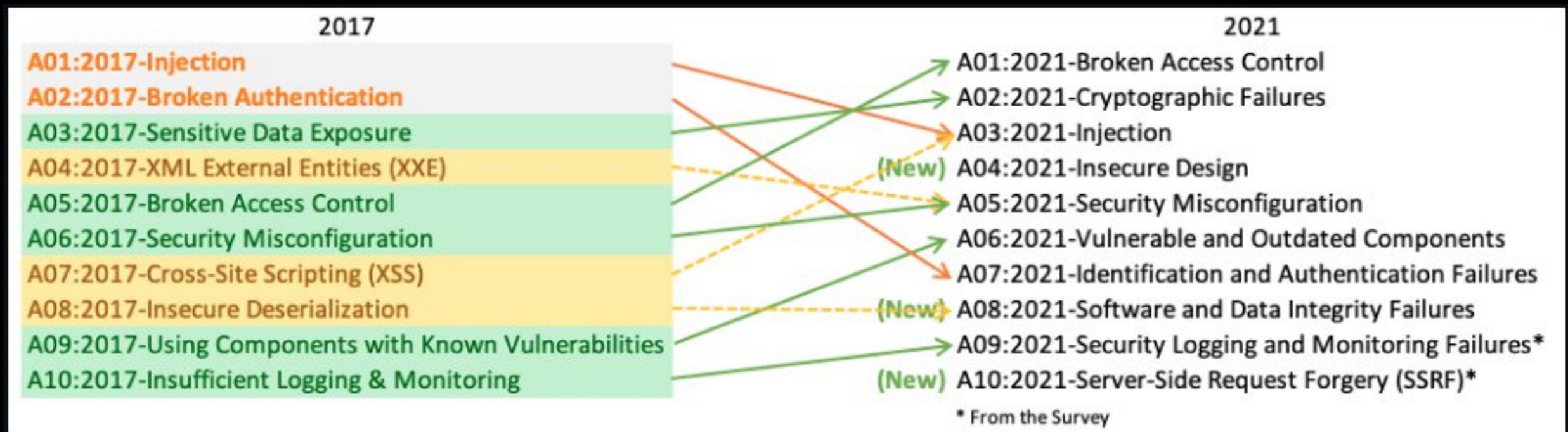
- Abstract libraries that **encapsulate knowledge** about attacks
- Commencing threat modelling from attack libraries alone **can be overwhelming.**
- Useful to **supplementing other threat modelling techniques**
 - Input and/or inspiration for attack trees
 - Implicit assets

OWASP Top 10

- An awareness tool for the most critical web security risks
- Could be used in the same way you might use STRIDE

T10 OWASP Top 10 Application Security Risks – 2017	
A1:2017-Injection	Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
A2:2017-Broken Authentication	Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
A3:2017-Sensitive Data Exposure	Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.
A4:2017-XML External Entities (XXE)	Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.
A5:2017-Broken Access Control	Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.
A6:2017-Security Misconfiguration	Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched and upgraded in a timely fashion.
A7:2017-Cross-Site Scripting (XSS)	XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
A8:2017-Insecure Deserialization	Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.
A9:2017-Using Components with Known Vulnerabilities	Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
A10:2017-Insufficient Logging & Monitoring	Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

OWASP Top 10



CAPEC

- Common Attack Pattern Enumeration and Classification
 - Over 500 different patterns across different categories
 - Classified by type of attack

The screenshot shows the CAPEC website's "About CAPEC" page. At the top, there's a red header bar with the CAPEC logo and the text "Common Attack Pattern Enumeration and Classification" and "A Community Resource for Identifying and Understanding Attacks". Below the header, a navigation bar includes links for Home, About, CAPEC List, Community, News, and Search. A search bar with placeholder text "ID Lookup: Go" is also present. The main content area has a title "About CAPEC" and a sub-section "Objective". It describes CAPEC as a publicly available catalog of common attack patterns that helps users understand how adversaries exploit weaknesses in applications and other cyber-enabled capabilities. It explains that "Attack Patterns" are descriptions of common attributes and approaches used by adversaries to exploit known weaknesses. The page also lists "Some Well-Known Attack Patterns" such as HTTP Response Splitting, Session Fixation, Cross Site Request Forgery, SQL Injection, Cross-Site Scripting, Buffer Overflow, Clickjacking, Relative Path Traversal, and XML Attribute Blowup. Further down, sections for "History" and "Benefits" are shown, along with a note about the tool's role in building secure cyber-enabled capabilities. A "Feedback" section at the bottom right includes a "BACK TO TOP" link.

CAPEC Common Attack Pattern Enumeration and Classification
A Community Resource for Identifying and Understanding Attacks

Home > About CAPEC

Home | About | CAPEC List | Community | News | Search

ID Lookup: Go

About CAPEC

Overview | Documents | Use Cases | Resources | Glossary | FAQs

Objective

The Common Attack Pattern Enumeration and Classification (CAPEC™) effort provides a publicly available catalog of common attack patterns that helps users understand how adversaries exploit weaknesses in applications and other cyber-enabled capabilities.

"Attack Patterns" are descriptions of the common attributes and approaches employed by adversaries to exploit known weaknesses in cyber-enabled capabilities. Attack patterns define the challenges that an adversary may face and how they go about solving it. They derive from the concept of design patterns applied in a destructive rather than constructive context and are generated from in-depth analysis of specific real-world exploit examples.

Each attack pattern captures knowledge about how specific parts of an attack are designed and executed, and gives guidance on ways to mitigate the attack's effectiveness. Attack patterns help those developing applications, or administering cyber-enabled capabilities to better understand the specific elements of an attack and how to stop them from succeeding.

Some Well-Known Attack Patterns:

- HTTP Response Splitting ([CAPEC-34](#))
- Session Fixation ([CAPEC-61](#))
- Cross Site Request Forgery ([CAPEC-62](#))
- SQL Injection ([CAPEC-66](#))
- Cross-Site Scripting ([CAPEC-63](#))
- Buffer Overflow ([CAPEC-100](#))
- Clickjacking ([CAPEC-103](#))
- Relative Path Traversal ([CAPEC-139](#))
- XML Attribute Blowup ([CAPEC-229](#))

[BACK TO TOP](#)

History

CAPEC was established by the [U.S. Department of Homeland Security](#) as part of the Software Assurance (SwA) strategic initiative of the [Office of Cybersecurity and Communications \(CS&C\)](#). Initially released in 2007, the [CAPEC List](#) continues to evolve with public participation and contributions to form a standard mechanism for identifying, collecting, refining, and sharing attack patterns among the cybersecurity community.

[BACK TO TOP](#)

Benefits

Attack patterns captured in such a formalized way can bring considerable value to the development and maintenance of cyber-enabled capabilities, including:

- **Training** – Educate software developers, testers, buyers, and managers.
- **Requirements** – Define potential threats.
- **Design** – Provide context for architectural risk analysis.
- **Implementation** – Prioritize review activities.
- **Verification** – Guide appropriate penetration testing.
- **Release** – Understand trends and attacks to monitor.
- **Response** – Leverage lessons learned into preventative guidance.

Of course, attack patterns are not the only useful tool for building secure cyber-enabled capabilities. Many other tools, such as misuse/abuse cases, security requirements, threat models, knowledge of common weaknesses and vulnerabilities, and attack trees, can help. Attack patterns play a unique role amid this larger architecture of security knowledge and techniques.

[BACK TO TOP](#)

Feedback

CAPEC-61: Session Fixation

Attack Pattern ID: 61

Status: Draft

Abstraction: Detailed

Presentation Filter: Basic ▾

Description

The attacker induces a client to establish a session with the target software using a session identifier provided by the attacker. Once the user successfully authenticates to the target software, the attacker uses the (now privileged) session identifier in their own transactions. This attack leverages the fact that the target software either relies on client-generated session identifiers or maintains the same session identifiers after privilege elevation.

Relationships

The table(s) below shows the other attack patterns and high level categories that are related to this attack pattern. These relationships are defined as ChildOf, ParentOf, MemberOf and give insight to similar items that may exist at higher and lower levels of abstraction. In addition, relationships such as CanFollow, PeerOf, and CanAlsoBe are defined to show similar attack patterns that the user may want to explore.

Nature	Type	ID	Name
MemberOf	●	370	WASC-37 - Session Fixation
ChildOf	■	593	Session Hijacking
CanFollow	■	196	Session Credential Falsification through Forging

Execution Flow

Explore

- Setup the Attack:** Setup a session: The attacker has to setup a trap session that provides a valid session identifier, or select an arbitrary identifier, depending on the mechanism employed by the application. A trap session is a dummy session established with the application by the attacker and is used solely for the purpose of obtaining valid session identifiers. The attacker may also be required to periodically refresh the trap session in order to obtain valid session identifiers. The attacker chooses a predefined identifier that he knows. The attacker creates a trap session for the victim.

Techniques

The attacker chooses a predefined identifier that he knows.

The attacker creates a trap session for the victim.

Experiment

- Attract a Victim:** Fixate the session: The attacker now needs to transfer the session identifier from the trap session to the victim by introducing the session identifier into the victim's browser. This is known as fixating the session. The session identifier can be introduced into the victim's browser by leveraging cross site scripting vulnerability, using META tags or setting HTTP response headers in a variety of ways. Attackers can put links on web sites (such as forums, blogs, or comment forms). Attackers can establish rogue proxy servers for network protocols that give out the session ID and then redirect the connection to the legitimate service. Attackers can email attack URLs to potential victims through spam and phishing techniques.

Techniques

Attackers can put links on web sites (such as forums, blogs, or comment forms).

Attackers can establish rogue proxy servers for network protocols that give out the session ID and then redirect the connection to the legitimate service.

Attackers can email attack URLs to potential victims through spam and phishing techniques.

Exploit

- Abuse the Victim's Session:** Takeover the fixated session: Once the victim has achieved a higher level of privilege, possibly by logging into the application, the attacker can now take over the session using the fixated session identifier. The attacker loads the predefined session ID into his browser and browses to protected data or functionality. The attacker loads the predefined session ID into his software and utilizes functionality with the rights of the victim.

Techniques

The attacker loads the predefined session ID into his browser and browses to protected data or functionality.

The attacker loads the predefined session ID into his software and utilizes functionality with the rights of the victim.

Prerequisites

Session identifiers that remain unchanged when the privilege levels change.

Permissive session management mechanism that accepts random user-generated session identifiers

Predictable session identifiers

Mitigations

Use a strict session management mechanism that only accepts locally generated session identifiers: This prevents attackers from fixating session identifiers of their own choice.

Regenerate and destroy session identifiers when there is a change in the level of privilege: This ensures that even though a potential victim may have followed a link with a fixated identifier, a new one is issued when the level of privilege changes.

Use session identifiers that are difficult to guess or brute-force: One way for the attackers to obtain valid session identifiers is by brute-forcing or guessing them. By choosing session identifiers that are sufficiently random, brute-forcing or guessing becomes very difficult.

More information is available — Please select a different filter.

CAPEC VIEW: Mechanisms of Attack

View ID: 1000
Structure: Graph

Downloads: Booklet | CSV | XML

Objective

This view organizes attack patterns hierarchically based on mechanisms that are frequently employed when exploiting a vulnerability. The categories that are members of this view represent the different techniques used to attack a system. They do not, however, represent the consequences or goals of the attacks. There exists the potential for some attack patterns to align with more than one category depending on one's perspective. To counter this, emphasis was placed such that attack patterns as presented within each category use a technique not sometimes, but without exception.

Relationships

The following graph shows the tree-like relationships between attack patterns that exist at different levels of abstraction. At the highest level, categories exist to group patterns that share a common characteristic. Within categories, meta level attack patterns are used to present a decidedly abstract characterization of a methodology or technique. Below these are standard and detailed level patterns that are focused on a specific methodology or technique used.

Show Details:

Expand All | Collapse All | Filter View

1000 - Mechanisms of Attack



CAPEC-697: DHCP Spoofing

Attack Pattern ID: 697

Abstraction: Standard

Presentation Filter: Basic

Description

An adversary masquerades as a legitimate Dynamic Host Configuration Protocol (DHCP) server by spoofing DHCP traffic, with the goal of redirecting network traffic or denying service to DHCP.

Extended Description

DHCP is broadcast to the entire Local Area Network (LAN) and does not have any form of authentication by default. Therefore, it is susceptible to spoofing.

An adversary with access to the target LAN can receive DHCP messages; obtaining the topology information required to potentially manipulate other hosts' network configurations.

To improve the likelihood of the DHCP request being serviced by the Rogue server, an adversary can first starve the DHCP pool.

Relationships

Nature	Type	ID	Name
ChildOf	S	194	Fake the Source of Data
CanPrecede	M	94	Adversary in the Middle (AiTM)
CanPrecede	D	158	Sniffing Network Traffic

View Name

Domains of Attack
Mechanisms of Attack

Top Level Categories

Software, Hardware, Social Engineering
Engage in Deceptive Interactions

Execution Flow

Explore

Determine Existing DHCP lease: An adversary observes network traffic and waits for an existing DHCP lease to expire on a target machine in the LAN.

Techniques

Adversary observes LAN traffic for DHCP solicitations

Experiment

Capture the DHCP DISCOVER message: The adversary captures "DISCOVER" messages and crafts "OFFER" responses for the identified target MAC address. The success of this attack centers on the capturing of and responding to these "DISCOVER" messages.

Techniques

Adversary captures and responds to DHCP "DISCOVER" messages tailored to the target subnet.

Exploit

Compromise Network Access and Collect Network Activity: An adversary successfully acts as a rogue DHCP server by redirecting legitimate DHCP requests to itself.

Techniques

Adversary sends repeated DHCP "REQUEST" messages to quickly lease all the addresses within network's DHCP pool and forcing new DHCP requests to be handled by the rogue DHCP server.

ATT&CK

- Adversarial Tactics, Techniques, and Common Knowledge
- A knowledge base of adversarial techniques based on
 - *tactics*: why an attacker carries out an attack.
 - *techniques*: what an attack does to achieve
- Tactics based on *kill chain* stage

MITRE ATT&CK Matrices Tactics ▾ Techniques ▾ Mitigations ▾ Groups Software Resources ▾ Blog ☰ Contribute Search site

ATT&CK Matrix for Enterprise

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Drive-by Compromise	AppleScript	.bash_profile and .basrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery	AppleScript	Audio Capture	Commonly Used Port	Automated Exfiltration	Data Destruction
Exploit Public-Facing Application	CMSTP	Accessibility Features	Accessibility Features	Binary Padding	Bash History	Application Window Discovery	Application Deployment Software	Automated Collection	Communication Through Removable Media	Data Compressed	Data Encrypted for Impact
External Remote Services	Command-Line Interface	Account Manipulation	AppCert DLLs	BITS Jobs	Brute Force	Browser Bookmark Discovery	Distributed Component Object Model	Clipboard Data	Connection Proxy	Data Encrypted	Defacement
Hardware Additions	Compiled HTML File	AppCert DLLs	Applinit DLLs	Bypass User Account Control	Credential Dumping	Domain Trust Discovery	Exploitation of Remote Services	Data from Information Repositories	Custom Command and Control Protocol	Data Transfer Size Limits	Disk Content Wipe
Replication Through Removable Media	Control Panel Items	Applinit DLLs	Application Shimming	Clear Command History	Credentials in Files	File and Directory Discovery	Logon Scripts	Data from Local System	Custom Cryptographic Protocol	Exfiltration Over Alternative Protocol	Disk Structure Wipe
Spearphishing Attachment	Dynamic Data Exchange	Application Shimming	Bypass User Account Control	CMSTP	Credentials in Registry	Network Service Scanning	Pass the Hash	Data from Network Shared Drive	Data Encoding	Exfiltration Over Command and Control Channel	Endpoint Denial of Service
Spearphishing Link	Execution through API	Authentication Package	DLL Search Order Hijacking	Code Signing	Exploitation for Credential Access	Network Share Discovery	Pass the Ticket	Data from Removable Media	Data Obfuscation	Exfiltration Over Other Network Medium	Firmware Corruption
Spearphishing via Service	Execution through Module Load	BITS Jobs	Dylib Hijacking	Compile After Delivery	Forced Authentication	Network Sniffing	Remote Desktop Protocol	Data Staged	Domain Fronting	Exfiltration Over Physical Medium	Inhibit System Recovery
Supply Chain Compromise	Exploitation for Client Execution	Bootkit	Exploitation for Privilege Escalation	Compiled HTML File	Hooking	Password Policy Discovery	Remote File Copy	Email Collection	Domain Generation Algorithms	Scheduled Transfer	Network Denial of Service
Trusted Relationship	Graphical User Interface	Browser Extensions	Extra Window Memory Injection	Component Firmware	Input Capture	Peripheral Device Discovery	Remote Services	Input Capture	Fallback Channels	Resource Hijacking	
Valid Accounts	InstallUtil	Change Default File Association	File System Permissions Weakness	Component Object Model Hijacking	Input Prompt	Permission Group Discovery	Replication Through Removable Media	Man in the Browser	Multi-hop Proxy		Runtime Data Manipulation
	Launchctl	Component Firmware	Hooking	Control Panel Items	Kerberoasting	Process Discovery	Shared Webroot	Screen Capture	Multi-Stage Channels		Service Stop
Local Job Scheduling	Component Object Model Hijacking	Image File Execution Options Injection	DCShadow	Keychain	Query Registry	SSH Hijacking	Video Capture		Multiband Communication		Stored Data Manipulation
	LSASS Driver	Create Account	Launch Daemon	Deobfuscate/Decode LLMNR/NBT-NS Files or Information	Poisoning and Relay	Remote System Discovery	Taint Shared Content		Multilayer Encryption		Transmitted Data Manipulation
PowerShell	Dylib Hijacking	New Service	Disabling Security Tools	Network Sniffing	Security Software Discovery	Third-party Software			Port Knocking		
		Path Interception	DLL Search Order Hijacking	Password Filter DLL	System Information Discovery	Windows Admin Shares			Remote Access Tools		
Regsvcs/Regasm	External Remote Services	Plist Modification	DLL Side-Loading	Private Keys	System Network Configuration Discovery	Windows Remote Management			Remote File Copy		
	Regsvr32	File System Permissions Weakness	Port Monitors	Execution Guardrails	System Network Connections Discovery				Standard Application Layer Protocol		
Rundll32	Hidden Files and Directories	Process Injection	Exploitation for Defense Evasion	Two-Factor Authentication Intermediate	System Owner/User Discovery				Standard Cryptographic Protocol		

TECHNIQUES

- Enterprise
 - Reconnaissance
 - Resource Development
 - Initial Access
 - Execution
 - Persistence
 - Privilege Escalation
 - Abuse Elevation Control Mechanism**
 - Setuid and Setgid
 - Bypass User Account Control
 - Sudo and Sudo Caching**
 - Elevated Execution with Prompt
 - Access Token Manipulation
 - Boot or Logon Autostart Execution
 - Boot or Logon Initialization Scripts
 - Create or Modify System Process
 - Domain Policy Modification
 - Escape to Host
 - Event Triggered Execution

Abuse Elevation Control Mechanism: Sudo and Sudo Caching

Other sub-techniques of Abuse Elevation Control Mechanism (4)

Adversaries may perform sudo caching and/or use the sudoers file to elevate privileges. Adversaries may do this to execute commands as other users or spawn processes with higher privileges.

Within Linux and MacOS systems, sudo (sometimes referred to as "superuser do") allows users to perform commands from terminals with elevated privileges and to control who can perform these commands on the system. The `sudo` command "allows a system administrator to delegate authority to give certain users (or groups of users) the ability to run some (or all) commands as root or another user while providing an audit trail of the commands and their arguments."^[1] Since sudo was made for the system administrator, it has some useful configuration features such as a `timestamp_timeout`, which is the amount of time in minutes between instances of `sudo` before it will re-prompt for a password. This is because `sudo` has the ability to cache credentials for a period of time. Sudo creates (or touches) a file at `/var/db/sudo` with a timestamp of when sudo was last run to determine this timeout. Additionally, there is a `tty_tickets` variable that treats each new tty (terminal session) in isolation. This means that, for example, the sudo timeout of one tty will not affect another tty (you will have to type the password again).

The sudoers file, `/etc/sudoers`, describes which users can run which commands and from which terminals. This also describes which commands users can run as other users or groups. This provides the principle of least privilege such that users are running in their lowest possible permissions for most of the time and only elevate to other users or permissions as needed, typically by prompting for a password. However, the sudoers file can also specify when to not prompt users for passwords with a line like `user1 ALL=(ALL) NOPASSWD: ALL`.^[2] Elevated privileges are required to edit this file though.

ID: T1548.003

Sub-technique of: T1548

- ① **Tactics:** Privilege Escalation, Defense Evasion
- ① **Platforms:** Linux, macOS
- ① **Permissions Required:** User
- ① **Effective Permissions:** root

Version: 1.0

Created: 30 January 2020

Last Modified: 14 March 2022

[Version Permalink](#)

TECHNIQUES

- Enterprise
 - Reconnaissance
 - Resource Development
 - Initial Access
 - Execution
 - Persistence
 - Privilege Escalation
 - Abuse Elevation Control Mechanism**
 - Setuid and Setgid
 - Bypass User Account Control
 - Sudo and Sudo Caching**
 - Elevated Execution with Prompt
 - Access Token Manipulation
 - Boot or Logon Autostart Execution
 - Boot or Logon Initialization Scripts
 - Create or Modify System Process
 - Domain Policy Modification
 - Escape to Host
 - Event Triggered Execution
 - Exploitation for Privilege Escalation
 - Hijack Execution Flow
 - Process Injection
 - Scheduled Task/Job
 - Valid Accounts
 - Defense Evasion
 - Credential Access

Procedure Examples

ID	Name	Description
S0154	Cobalt Strike	Cobalt Strike can use <code>sudo</code> to run a command. ^[4]
S0281	Dok	Dok adds <code>admin ALL=(ALL) NOPASSWD: ALL</code> to the <code>/etc/sudoers</code> file. ^[5]
S0279	Proton	Proton modifies the <code>tty_tickets</code> line in the sudoers file. ^[6]

Mitigations

ID	Mitigation	Description
M1028	Operating System Configuration	Ensuring that the <code>tty_tickets</code> setting is enabled will prevent this leakage across tty sessions.
M1026	Privileged Account Management	By requiring a password, even if an adversary can get terminal access, they must know the password to run anything in the sudoers file. Setting the <code>timestamp_timeout</code> to 0 will require the user to input their password every time <code>sudo</code> is executed.
M1022	Restrict File and Directory Permissions	The sudoers file should be strictly edited such that passwords are always required and that users can't spawn risky processes as users with higher privilege.

Detection

ID	Data Source	Data Component	Detects
DS0017	Command	Command Execution	Monitor executed commands and arguments that may perform sudo caching and/or use the sudoers file to elevate privileges, such as the <code>sudo</code> command.
DS0022	File	File Modification	On Linux, auditd can alert every time a user's actual ID and effective ID are different (this is what happens when you sudo). This technique is abusing normal functionality in macOS and Linux systems, but sudo has the ability to log all input and output based on the <code>LOG_INPUT</code> and <code>LOG_OUTPUT</code> directives in the <code>/etc/sudoers</code> file.
DS0009	Process	Process Creation	Monitor newly executed processes that may perform sudo caching and/or use the sudoers file to elevate privileges.
		Process Metadata	Monitor contextual data about a running process, which may include information such as environment variables, image name, user/owner that may perform sudo caching and/or use the sudoers file to elevate privileges.

GPT Agents (Pro rqd)

Awesome GPTs (Agents) for Cybersecurity

stars 572 forks 47

The "Awesome GPTs (Agents) Repo" represents an initial effort to compile a comprehensive list of GPT agents focused on cybersecurity (offensive and defensive), created by the community.

- [HackTricksGPT](#): A knowledgeable cybersecurity professional.
- [Cyber Security Tutor](#): Quality Cyber Security Advice, Tricks, & Tips
- [MITREGPT](#): Feed me any input and I'll match it with the relevant MITRE ATT&CK techniques and tactics.
- [ATT&CK Mate](#): Ask me anything about the ATT&CK Knowledge Base.
- [Hacker News GPT](#): Summarizes top Hacker News stories and comments daily or weekly, with precise search parameters.
- [Hacker Art](#): Generate badass hacker art and profile pics.
- [Threat Modelling](#): A GPT expert in conducting thorough threat modelling for system design and review.
- [Threat Modeling Companion](#): I am a threat modeling expert that can help you identify threats for any system that you provide.
- [Squidshing](#): Analyzes emails for phishing risks.
- [GetPaths](#): This GPT takes in content related to an application, such as HTTP traffic, JavaScript files, source code, etc., and outputs lists of URLs that can be used for further testing.

New GPT
• Draft

Create Configure

[+](#)

Name
Name your GPT

Description
Add a short description about what this GPT does

Instructions
What does this GPT do? How does it behave? What should it avoid doing?

Conversation starters

Knowledge
Upload files

Capabilities
 Web Browsing
 DALL-E Image Generation
 Code Interpreter

Actions
Add actions

Threat Modelling GPT

Threat Modelling ▾



You

Describe the system for threat analysis.

Threat Modelling

To provide a comprehensive threat analysis, I need a detailed description of the system you want to analyze. Please include the following information:

- System Overview:** Describe the system's main purpose, components, and functionality. This includes software, hardware, and network elements.
- Data Flow:** Explain how data moves through your system. This might include user inputs, database interactions, and any external APIs or services your system communicates with.
- User Interaction:** Detail how users interact with the system. This includes both legitimate users and potential unauthorized users.
- Security Measures:** List any existing security measures, such as firewalls, encryption, authentication mechanisms, etc.
- External Interfaces:** Identify any external systems or services your system interfaces with, such as third-party APIs, cloud services, or other external data sources.
- Regulatory Requirements:** If applicable, specify any regulatory compliance requirements your system must meet, like GDPR, HIPAA, PCI-DSS, etc.
- Physical Environment:** If relevant, describe the physical environment where the system is deployed. This could include data centers, office environments, or mobile settings.
- Operational Environment:** Provide details about the operational environment, such as the network infrastructure, operating systems used, and the types of devices that access the system.

ATT&CK Mate GPT

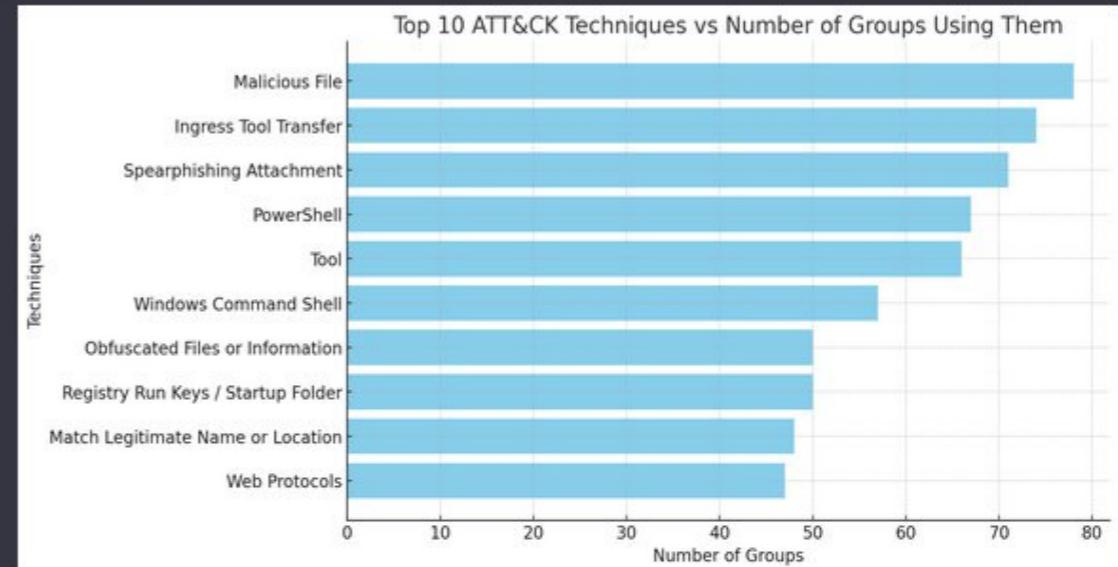
ATT&CK Mate ▾



You

Create a bar chart of top 10 techniques vs # of groups

ATT&CK Mate



Here is the bar chart displaying the top 10 techniques from the ATT&CK framework and the number of groups that utilize each technique. The chart provides a clear visual representation, showing which techniques are most commonly used by various groups. [Link]



Sources

- [1] Schoenfield, B. S. E. Securing Systems: Applied Security Architecture and Threat Models. CRC Press, 2015.
- [2] Shostack, A. Threat Modeling: Designing for Security. John Wiley & Sons, 2014.
- [3] https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
- [4] <https://capec.mitre.org>
- [5] <https://attack.mitre.org>
- [6] <https://attack.mitre.org/techniques/T1548/003/>