Threat Intel Report on

MITRE ATT&CK® Framework macOS Platform

by

Team Data Wizards

(Shruti Shahu - 2040, Sanika Raul - 2041,

Riya Kadam – 2044, Vibhuti Naik - 2046)

Table of Contents

MITRE ATT&CK® Framework

Introduction

- MITRE ATT&CK is a community-driven knowledge base of tactics, techniques, and procedures that are used by malicious actors to compromise computers or mobile devices.
    - **Tactic:** goal or reason for a malicious actor to perform an action
    - **Technique:** action or method used to achieve a tactical goal
    - **Procedure:** real-world example of a technique in action
- Techniques can contain sub-techniques, a more specific variant or lower-level description of a malicious technique.
- For example, in the macOS Matrix, the Phishing technique contains the following sub-techniques:
    - Spearphishing Attachment
    - Spearphishing Link
    - Spearphishing via Service
    - Spearphishing Voice
- MITRE ATT&CK matrices are organized by platform, and separated between computers and mobile devices.
- Tactics, techniques, and procedures can be used to create detection analytics to monitor and respond to malicious actions.
- For security teams, the MITRE ATT&CK framework into both a reference and a playbook.
- The framework explains how its matrices are organized, which platforms and environments they cover, and how tactics and techniques connect from one phase of an attack to the next.

The 12 Enterprise (macOS) Tactics

1. Initial Access (TA0001)

Tactic Observation:

Initial Access is the first and most critical stage of the cyber attack lifecycle, as defined by the MITRE ATT&CK framework. This tactic represents the techniques used by adversaries to gain unauthorized entry into a target system, application, or network. Successful initial access allows attackers to establish a foothold in the environment, enabling them to execute further malicious actions such as privilege escalation, lateral movement, persistence, data exfiltration, or system disruption. Attackers often exploit technical vulnerabilities, misconfigurations, or human trust to bypass security controls during this stage.

Tactic Description:

The Initial Access tactic focuses on how an attacker enters a system for the first time. This access can be achieved through multiple vectors such as phishing, exploitation of public-facing applications, malicious file uploads, drive-by downloads, or content injection. In modern cyber environments, attackers increasingly rely on stealthy techniques that blend malicious activities with legitimate user behavior, making detection difficult.

Initial Access is crucial because it determines the attacker's ability to remain undetected and expand control within the system. Organizations with weak input validation, poor patch management, or inadequate monitoring are especially vulnerable at this stage. Once access is gained, attackers can deploy malware, steal credentials, or manipulate system configurations to maintain persistence.

Tactic ID: TA0001

Total Techniques: 10

ATT&CK Version: Created 17 October 2018

Technique1: Content Injection

Description:

Content Injection is a technique under the Initial Access tactic in which an attacker injects malicious content into trusted or legitimate platforms such as websites, applications, or data streams. This technique is particularly dangerous because it exploits the trust users place in legitimate systems.

Content Injection involves inserting unauthorized or malicious content into a legitimate digital resource. This content may include malicious scripts, altered web pages, injected advertisements, or hidden executable code. When users access the compromised content, the injected payload executes automatically or through user interaction.

The injected content often appears harmless, which allows attackers to bypass security mechanisms and trick users into unknowingly enabling malicious actions.

Procedure:

The typical procedure followed by attackers during content injection includes:

- Identifying a vulnerable website or application.
- Scanning for insecure input fields or outdated components.
- Injecting malicious content into the application or data stream.
- Ensuring the injected content blends with legitimate content.
- Waiting for users to access the compromised resource.
- Executing malicious payloads upon user interaction.
- Gaining initial system access or user credentials.

Mitigations Strategies:

To prevent content injection attacks, organizations should implement the following mitigations:

Mitigation	Description	Priority
Input Validation & Sanitization	Validate and sanitize all user inputs to prevent injection of malicious content into applications	High
Secure Coding Practices	Follow secure development standards to eliminate vulnerabilities such as XSS and improper input handling	High
Content Security Policy (CSP)	Implement CSP headers to restrict execution of untrusted scripts and content sources	High
Regular Patch Management	Keep web servers, frameworks, and CMS platforms updated with latest security patches.	High
Principle of Least Privilege (PoLP)	Restrict user and application privileges to minimum required access.	Medium
Web Application Firewall (WAF)	Deploy WAF to filter and block malicious web requests.	Medium
Third-Party Content Control	Restrict and monitor third-party scripts, ads, and plugins.	Medium
Security Awareness Training	Educate developers and users about secure coding and content risks.	Low
Regular Security Audits	Conduct periodic vulnerability scans and penetration testing	Low

File Integrity Monitoring	Monitor web files for unauthorized changes.	Low

Detection Methods:

Detection of content injection attacks requires continuous monitoring and analysis. Effective detection methods include:

- Web Application Firewalls (WAFs)
- File integrity monitoring systems
- Regular code and content audits
- Log analysis for abnormal behaviors
- Network traffic monitoring
- Behavioral analysis of user sessions
- Intrusion Detection Systems (IDS)

Technique2: Phishing (T1566).

Description:

Phishing is one of the most commonly used social engineering techniques under the Initial Access tactic in the MITRE ATT&CK framework. In phishing attacks, adversaries attempt to deceive users into revealing sensitive information such as login credentials, personal data, or financial details by impersonating trusted entities. These attacks are usually carried out through emails, messages, fake websites, or malicious attachments that appear legitimate.

Phishing exploits human psychology rather than technical vulnerabilities. Attackers rely on urgency, fear, curiosity, or trust to manipulate victims into taking actions that compromise security. Due to its low cost and high success rate, phishing remains a preferred entry point for attackers across industries.

Sub-Techniques:

Phishing is further divided into multiple sub-techniques based on the method of delivery:

1. **Spearphishing Attachment (T1566.001):**This involves sending emails with malicious attachments such as PDF files, Word documents, or ZIP files. When the victim opens the attachment, malware is executed or a malicious macro is triggered.
2. **Spearphishing Link (T1566.002):** Attackers include malicious links in emails or messages that redirect users to fake login pages or malware-hosting websites.
3. **Spearphishing via Service (T1566.003):** This sub-technique uses third-party services such as social media platforms, collaboration tools, or messaging applications to deliver phishing messages.

Procedure:

The general procedure followed in a phishing attack includes the following steps:

- The attacker identifies a target individual or organization.
- Information about the target is collected to make the message convincing.
- A fake email, message, or website is crafted to impersonate a trusted source.
- The phishing content is delivered through email, SMS, or online services.
- The victim interacts with the message by clicking a link, downloading an attachment, or entering credentials.
- The attacker captures the sensitive information or gains unauthorized access.
- The compromised access is used for further attacks such as privilege escalation or data theft.

Mitigations Strategy:

Priority	Descriptions	Mitigation Techniques
High	Conduct regular training programs to educate users on identifying phishing emails, malicious links, and suspicious attachments	Security Awareness Training
High	Adds an extra verification step to protect accounts from unauthorized access.	Multi-Factor Authentication (MFA)
High	Automatically detect and block suspicious or malicious emails.	Email Filtering & Anti-Phishing Tools
High	Prevents execution of harmful code from email attachments.	Disable or Restrict Macros
Medium	Users get only necessary access required for their work	Principle of Least Privilege (PoLP)
Medium	Scans and filters emails to stop threats before delivery.	Secure Email Gateway (SEG)
Medium	Authenticate email sources to prevent spoofing and phishing.	Domain & Email Authentication (SPF, DKIM, DMARC)
Low	Verifies sender identity to detect spoofed or fake emails..	Domain & Email Authentication (SPF, DKIM, DMARC)
Low	Enables users to report suspicious emails quickly for faster action.	Incident Reporting Mechanism

Low	Uses known threat data to identify phishing attempts early	Threat Intelligence Integration

Detection Methods:

Effective detection of phishing attacks involves multiple security layers, including:

- Email security solutions that detect malicious content.
- Monitoring for abnormal login attempts.
- Detection of credential misuse.
- Web proxy monitoring for access to known phishing sites.
- User-reported phishing emails.
- Behavioral analytics to identify compromised accounts.
- Security Information and Event Management (SIEM) systems.
- Threat intelligence feeds for known phishing indicators.

2. Execution (TA0002)

Tactic Observation:

The Execution tactic focuses on how adversaries run malicious code on a target system after gaining initial access. This stage enables attackers to activate payloads, maintain control, and continue malicious operations. Execution techniques are commonly used to establish persistence, escalate privileges, or enable further attack stages.

Tactic Description:

Execution refers to the methods adversaries use to run malicious programs, scripts, or commands on compromised systems. Attackers may leverage

legitimate system tools, scheduled jobs, or inter-process communication mechanisms to execute code while avoiding detection. Successful execution allows adversaries to maintain access, deploy malware, and perform unauthorized actions.

Tactic ID: TA0002

Total Techniques: 10

Att&ck and Created: 17 October 2018

Technique1: Scheduled Task/Job(T1053)

Description:

Scheduled Task/Job is a technique where attackers create or modify scheduled tasks to execute malicious code automatically at a specified time or system event. This technique is widely used to maintain persistence and ensure repeated execution of malware without user interaction.

Attackers exploit native scheduling utilities such as Task Scheduler (Windows) or jobs (Linux/Unix) to disguise malicious activities as legitimate system operations. Once established, the malicious task can execute scripts, payloads, or backdoors regularly.

Detection Methods:

Detection of malicious scheduled tasks requires continuous monitoring of system scheduling services. Security teams should identify abnormal task creation, unexpected execution times, and unauthorized command execution. Behavioral analysis and endpoint monitoring tools help detect suspicious task activity.

The Real World-Case:

In real-world cyberattacks, ransomware groups frequently use scheduled tasks to re-execute malware after system reboots. Advanced Persistent Threats (APTs) also rely on scheduled jobs to maintain stealthy persistence across enterprise environments.

Sub-Techniques:

- **T1053.002 – At (Windows):** Uses the at.exe command to schedule tasks.
    - Attackers use it for delayed or persistent execution.
- **T1053.005 – Scheduled Task (Windows):** Creates or modifies Windows scheduled tasks.
    - Used to maintain persistence or run malware regularly.
- **T1053.006 – Cron (Linux/macOS):** Schedules jobs using cron.
    - Adversaries add malicious commands for persistence.
- **T1053.007 – Container Orchestration Job:** Schedules tasks in container environments (e.g., Kubernetes).
    - Used to run malicious jobs inside clusters

Mitigation Strategies(T1053):

Id	Mitigation	Description
M1048	Application Isolation and Sandboxing	Ensure all COM alerts and Protected View are enabled.
M1040	Behavior Preventation on Endpoint	On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent DDE attacks and spawning of child processes from Office programs.

M1054	Software Configuration	Consider disabling embedded files in Office programs, such as OneNote, that do not work with Protected View.
M1026	Privileged Account Management	Modify Registry settings (directly or using Dcomcnfg.exe)
M1042	Disable or Remove Feature or Program	Registry keys specific to Microsoft Office feature control security can be set to disable automatic DDE/OLE execution.

Technique2: Inter-Process:

Description:

Inter-Process Communication (IPC) is a technique where adversaries use communication between processes to execute malicious actions. IPC mechanisms allow processes to exchange data and signals, which attackers abuse to inject code, control execution flow, or trigger malicious payloads.

Attackers may exploit shared memory, named pipes, sockets, or messaging systems to communicate between malicious and legitimate processes. This approach helps adversaries evade detection by hiding malicious execution within trusted system processes.

IPC-based execution is commonly used in advanced malware and fileless attacks. Since legitimate applications frequently use IPC, malicious activity can easily remain unnoticed without deep Communication(T1559)

Detections Methods:

Detection of malicious IPC activity involves monitoring unusual inter-process communication patterns. Security solutions should identify abnormal

data exchanges, unauthorized process interactions, and suspicious command execution flows

The Real World-Case:

In advanced attacks, malware uses IPC to communicate with injected processes, allowing attackers to control infected systems remotely. Fileless malware often relies on IPC to execute commands without writing files to disk.

Sub-Techniques:

- **T1559.001 – Named Pipe:** Processes communicate using named pipes to exchange data.
    - Attackers use it for stealthy command execution between processes.
- **T1559.002 – Apple Events**: macOS processes send events to control other applications.
    - Adversaries abuse it to execute commands without user interaction.
- **T1559.003 – D-Bus:** Linux IPC system used for communication between services.
    - Attackers exploit it to trigger actions in privileged processes.
- **T1559.004 – XPC Services**: macOS IPC mechanism for app-to- service communication.
    - Used by attackers to run malicious code via trusted services.

Mitigation Strategies:

Id	Mitigation	Description
M1032	Multi-factor Authentication	Use multi-factor authentication for user and privileged accounts.

M1030	Network Segmentation	Configure access controls and firewalls to limit access to domain controllers and systems used to create and manage accounts.
M1028	Operating System Configuration	Protect domain controllers by ensuring proper security configuration for critical servers.
M1026	Privileged Account Management	Limit the number of accounts with permissions to create other accounts. Do not allow domain administrator accounts to be used for day-to- day operations

3. Persistence (TA0003)

Tactic Observation:

Persistence is a critical phase in the cyber attack lifecycle where adversaries aim to maintain long-term access to a compromised system. Once initial access is achieved, attackers deploy persistence mechanisms to survive system reboots, credential changes, or security updates. This tactic allows attackers to continue their operations without repeatedly exploiting the system.

Persistence techniques are designed to be stealthy and often blend into legitimate system behavior. Adversaries rely on misconfigurations, weak identity controls, and trusted system components to remain undetected for extended periods. Without effective monitoring, persistent threats can remain active for months or even years.

Tactic Description:

The Persistence tactic focuses on methods used by adversaries to ensure their continued presence in a compromised environment. Persistence enables attackers to repeatedly access systems, execute malicious code, steal data, and move laterally across networks. These techniques are often executed immediately after initial access to secure a foothold.

Attackers commonly exploit operating system features such as user accounts, authentication mechanisms, scheduled tasks, startup scripts, and registry keys. By abusing legitimate services, adversaries avoid triggering security alerts and appear as authorized users. Persistence mechanisms may be simple, such as creating a hidden account, or advanced, such as modifying authentication workflows.

Tactical ID: TA0003

Total Techniques: 18

Att&ck and Created: 17 October 2018

Technique 1 Create Account (T1136):

Description:

Create Account is a persistence technique where attackers establish new user accounts within a compromised system or network. These accounts allow adversaries to regain access even if the original entry point is removed. Created accounts may be local, domain-based, cloud-based, or application-specific.

Attackers often disguise malicious accounts by using names similar to legitimate users or service accounts. In some cases, these accounts are assigned administrative privileges, allowing full system control. The technique is highly effective because user accounts are trusted entities within authentication systems.

Detection Methods:

Detection of unauthorized account creation requires continuous monitoring of authentication systems and identity management platforms. Security teams should analyze logs for abnormal account creation events, privilege assignments, and account usage patterns.

Behavioral analysis helps identify accounts that authenticate at unusual times, from unexpected locations, or perform unauthorized actions. Integration with SIEM solutions improves visibility across endpoints, servers, and cloud services.

The Real World-Case:

Numerous ransomware and APT campaigns use account creation to maintain persistence. In enterprise breaches, attackers create domain admin accounts to regain access after cleanup attempts. Cloud-based attacks frequently involve the creation of shadow administrator accounts in identity platforms.

In real incidents, organizations often discover malicious accounts months after compromise, highlighting the effectiveness of this technique.

Mitigation Strategy:

Id	Mitigation	Description
M1032	Multi-factor Authentication	Use multi-factors authentication for user and privileged accounts
M1030	Network Segmentation	Configure access controls and firewalls to limit access to domain controllers and systems used to create and manage accounts

M1028	Operating system Configurations	Protect domain controllers by ensuring proper security configuration for critical servers
M1026	Privileged Account Management	Limit the number of accounts with permission to create other accounts

Sub-Techniques:

- **T1136.001 – Local Account**: Attackers create a local user account on a compromised system to maintain persistent access.
    - These accounts may be hidden or given administrative privileges to avoid detection and regain access even after system reboots.
- **T1136.002 – Domain Account**: Adversaries create accounts within a domain environment such as Active Directory.
    - Domain accounts allow attackers to move laterally across multiple systems and maintain long-term access to enterprise networks.
- **T1136.003 – Cloud Account**: Attackers create new user accounts in cloud environments like Azure AD or AWS.
    - These accounts enable persistent access to cloud resources and services without relying on traditional malware

Technique 2 Modify Authentication Process (T1556):

Description:

Modify Authentication Process is an advanced persistence technique where attackers alter authentication mechanisms to maintain access. Instead of creating new accounts, adversaries manipulate existing authentication workflows to bypass security controls.

Attackers may modify password validation logic, inject malicious authentication modules, alter credential storage, or weaken authentication

policies. This technique allows persistent access without creating visible artifacts such as new accounts.

Adversaries often target systems like Active Directory, PAM modules, Single Sign-On (SSO) systems, and cloud identity providers. By controlling authentication, attackers can impersonate users, escalate privileges, and bypass multi-factor authentication.

Detection Methods:

Detection involves monitoring authentication infrastructure for unauthorized changes. Security teams should review configuration changes, authentication logs, and policy modifications.

Anomalies such as successful logins without MFA, altered password policies, or unexpected authentication modules indicate possible compromise. Endpoint detection tools and identity monitoring solutions are critical for detecting this technique

The Real World-Case:

Advanced Persistent Threat groups use authentication modification to maintain stealthy access. In high-profile breaches, attackers have modified authentication services to silently accept malicious credentials, allowing long-term infiltration.

Cloud identity attacks frequently involve disabling MFA or modifying conditional access policies to maintain persistence.

Mitigation Strategy:

Id	Mitigation	Description
M1 018	User Account Management	Integrating multi-factor authentication (MFA) as part of organizational policy can greatly reduce the risk of an adversary gaining control of valid credentials that may be used for additional tactics such as initial access, lateral movement, and collecting information. MFA can also be used to restrict access to cloud resources and APIs.
M1 024	Restrict Registry Permission	Restrict Registry permissions to disallow the modification of sensitive Registry keys such as HKEY_LOCAL_MACHINE\SYSTEM\CurrentControl Set\Control\NetworkProvider\Order.
M1 022	Restrict File and Director y Permission	Restrict write access to the /Library/Security/SecurityAgentPlugins directory.
M1 027	Password Policies	Ensure that AllowReversiblePasswordEncryption property is set to disabled unless there are application requirements.[
M1 032	Multi- Factor Authentication	Integrating multi-factor authentication (MFA) as part of organizational policy can greatly reduce the risk of an adversary gaining control of valid credentials that may be used for additional

Sub-Techniques:

- **T1556.001 – Domain Controller Authentication**: Attackers modify authentication processes on a domain controller to bypass normal security checks.
    - This allows adversaries to authenticate as legitimate users or

administrators without valid credentials.

- **T1556.002 – Password Filter DLL**: Adversaries install a malicious password filter DLL to intercept or manipulate user passwords.
    - This technique enables attackers to capture credentials or weaken password enforcement policies.
- **T1556.003 – Pluggable Authentication Modules (PAM):** Attackers alter or replace PAM modules in Linux or Unix systems.
    - This allows them to bypass authentication controls or grant unauthorized access during login.
- **T1556.004 – Network Device Authentication:** Adversaries modify authentication mechanisms on network devices such as routers or firewalls.
    - This enables persistent access to network infrastructure and monitoring of network traffic.
- **T1556.005 – Reversible Encryption**: Attackers configure systems to store passwords using reversible encryption.
    - This allows adversaries to recover plaintext credentials from stored password data.
- **T1556.006 – Multi-Factor Authentication**: Adversaries weaken, bypass, or disable MFA protections.
    - This allows them to authenticate using stolen credentials without triggering additional verification steps.
- **T1556.007 – Hybrid Identity:** Attackers exploit synchronization between on-premise and cloud identity systems.
    - Changes made locally are propagated to the cloud, giving attackers persistent access across environments.
- **T1556.008 – Network Provider DLL:** Adversaries replace or modify network provider DLLs to intercept authentication requests. This enables credential theft and unauthorized authentication.
- **T1556.009 – Conditional Access Policies**: Attackers modify conditional access rules to remove security restrictions.
    - This allows logins from unauthorized locations, devices, or users without detection.

4. Privilege Escalation (TA0004)

Tactic Observation:

It is for adversaries to gain higher-level permissions on a system or network, allowing them to access protected resources, modify configurations, install services, or perform actions normally reserved for administrators.

Tactic Description:

Privilege Escalation (TA0004) is a critical tactic in cyberattacks where adversaries seek to elevate their permissions on a system or network after gaining initial access, moving from a regular user account to one with administrative or root privileges; by doing so, attackers can access protected resources, modify system configurations, install unauthorized software, disable security controls, and maintain persistent access, which allows them to further their objectives such as data theft, lateral movement, or launching additional attacks; this tactic leverages vulnerabilities, misconfigurations, or weaknesses in security controls to exploit the system, making it a pivotal phase in the attack lifecycle and a major concern for cybersecurity defenses.

Tactic ID: TA0004

Total Techniques: 11

Typical Phase: Post-initial access

ATT&CK Version: Created October 2018

Technique1: Abuse Elevation Control Mechanism (T1548)

Description:

It is a technique where adversaries bypass or misuse built-in system mechanisms designed to control elevated privileges, such as User Account Control (UAC) on Windows or sudo on Linux, to gain higher-level permissions on a system. These controls are intended to limit which users can perform high-risk tasks,

requiring explicit authorization for elevated actions. Attackers exploit weaknesses, misconfigurations, or weak rules in these mechanisms to run commands or processes with elevated privileges, often without needing to exploit a vulnerability—just by misusing existing permissions. This allows them to access restricted resources, modify system settings,or perform actions normally reserved for administrators.

Sub-Techniques:

1. **T1548.001**: Abuse Elevation Control Mechanism: Adversaries misuse built-in elevation controls to gain higher privileges without exploiting vulnerabilities, often by abusing misconfigurations or weak rules.
2. **T1548.002**: Bypass User Account Control: Attackers circumvent Windows User Account Control (UAC) to run processes with elevated privileges, often using known bypass techniques.
3. **T1548.003**: Sudo and Sudo Caching: Adversaries may exploit sudo caching or misconfigurations in the sudoers file to execute commands with elevated privileges, often bypassing intended restrictions or authentication requirements.
4. **T1548.004**: Elevated Execution with Prompt: Adversaries use prompts or authorization dialogs to gain elevated privileges, sometimes tricking users into approving malicious actions.
5. **T1548.005**: Abuse Cloud Instance Metadata API: Attackers misuse cloud instance metadata APIs to escalate privileges in cloud environments, often by accessing sensitive credentials or permissions.
6. **T1548.006**: Abuse Sudo and Sudoers: Adversaries exploit sudo misconfigurations or weak sudoers rules to run commands with elevated privileges on Linux/Unix systems

Detection Methods:

- Monitor for suspicious registry modifications related to elevation controls, such as UAC bypass keys.
- Watch for unusual parent-child process relationships, like control.exe spawning cmd.
- Detect executions where the user ID differs from the effective user ID, indicating sudo or privilege escalation.

- Look for setuid/setgid bit changes and elevated binaries launched by unprivileged users.
- Review Windows audit logs for event codes related to process creation and logon session metadata.
- Use auditd and unified logs on Linux/macOS to identify unauthorized privilege escalation attempts and suspicious API calls.

The Real-World Case:

SolarWinds (APT29)

During the SolarWinds compromise, the attackers didn't just stop at entering the network; they needed to move laterally and escalate privileges to access sensitive cloud data. They frequently used T1548.002 (Bypass User Account Control).

Mitigation Strategies:

Mitigation	Implementation Details	Priority
Privileged Account Management	Enforce the Principle of Least Privilege (PoLP). Limit the number of users in local "Administrators" or "sudo" groups. Use Just-in-Time (JIT) access tools to grant elevated rights only when necessary.	Critical
User Account Control (UAC) Policy	On Windows, set UAC to "Always Notify". This prevents silent elevation by processes and ensures a prompt is required for any administrative task, making bypasses more difficult to execute unnoticed.	High
Audit & Logging	Monitor for unusual parent-child process relationships (e.g., a standard app spawning a high-integrity shell). Enable Command Line Logging (Event ID 4688) and monitor /var/log/auth.log for sudo abuse.	High
Execution Prevention	Use Application Control (like AppLocker or WDAC) to block known UAC-bypass binaries or unauthorized scripts. Block execution from common user-writable directories like Temp or Downloads.	Medium

Operating System Hardening	Regularly audit Linux systems for unnecessary SUID/SGID bits on binaries. Remove the SUID bit from tools that users don't need (e.g., nmap, vim, or find) to prevent privilege escalation.	Medium
Update & Patch Management	Keep OS kernels and system-level services (like sudo or Systemd) updated to the latest versions to patch known vulnerabilities that allow for privilege escalation.	Medium

Technique2: Account Manipulation (T1098)

Description:

Account Manipulation (T1098) is a technique where adversaries modify existing accounts or security settings to maintain persistence, escalate privileges, or evade detection. Unlike simply creating a new account (which is more likely to trigger alerts), account manipulation involves "living off the land" by altering legitimate system or cloud accounts.

Sub-Techniques:

1. **T1098.001:** Additional Cloud Credentials: Adversaries add their own credentials (like API keys, certificates, or app passwords) to an existing cloud account. This allows them to log in even if the original user's password is changed.
2. **T1098.002**: Additional Email Delegate Permissions: In environments like Outlook/Exchange, attackers grant themselves "delegate" access to another user's mailbox. This lets them read, send, or delete emails on behalf of the victim without needing their password.
3. **T1098.003**: Additional Cloud Roles: Attackers assign new, high-level roles (like "Global Admin" or "Owner") to an account they control. This upgrades a low-level foothold into full control over the cloud tenant.
4. **T1098.004**: SSH Authorized Keys: On Linux/macOS, attackers add their public SSH key to a user's (~/.ssh/authorized_keys) file. This grants them persistent, passwordless remote access to the server.
5. **T1098.005**: Device Registration: Adversaries register their own device (laptop/phone) to a victim's account in systems like Microsoft Entra ID (Azure AD). This can bypass Multi-Factor Authentication (MFA) or conditional access policies.

6. **T1098.006**: Additional Container Cluster Roles: Attackers modify permissions within container orchestration platforms (like Kubernetes) to grant a specific service account or user "Cluster Admin" rights.
7. **T1098.007**: Additional Local or Domain Groups: The "classic" method: adding a compromised user to a sensitive group like Domain Admins (Windows) or the sudo group (Linux) to gain administrative control.

Detection Methods:

- **Monitor Event ID 4738** to detect any modifications to sensitive user account attributes.
- **Track Event ID 4728 and 4732** for unauthorized additions to high-privileged local or domain groups.
- **Identify anomalous API calls** like CreateAccessKey (AWS) or Add member to role (Azure) for cloud credential abuse.
- **Audit File Integrity** for changes to /etc/sudoers or ~/.ssh/authorized_keys on Linux systems.
- **Analyze process logs** for the execution of native tools like net.exe, usermod, or dscl outside of admin windows.
- **Flag "Subject-Target Mismatch" events** where a non-administrative user modifies another account's permissions.
- **Detect dormant account reactivation** followed immediately by credential changes or privilege escalation.
- **Scan for new device registrations** in Entra ID (Azure AD) originating from previously unobserved IP addresses.
- **Monitor mailbox delegate changes** for unauthorized access granted to sensitive email accounts.
- **Watch for iterative password resets** used to bypass password history and duration policies.

The Real-World Case:

SolarWinds (APT29): The attackers didn't just steal passwords; they manipulated the identity infrastructure (Microsoft 365 and Azure AD) to create their own "permanent" access.

- **Cloud Credential Addition (T1098.001):** Once inside the cloud environment, APT29 added their own certificates and "secrets" to existing legitimate **Service Principals** (automated accounts used by applications). This allowed them to log in as those applications and read emails or data without needing a user's password.
- **Device Registration (T1098.005):** In some instances, the group identified **dormant (inactive) accounts** that lacked Multi-Factor Authentication (MFA). They "reactivated" these accounts by registering their own attacker-controlled devices to them. This made their rogue laptops appear as "trusted corporate devices," bypassing security checks.
- **Mailbox Permission Manipulation (T1098.002):** The attackers used administrative access to grant themselves **Full Access** or **Delegate permissions** on the mailboxes of high-ranking executives. This allowed them to sync and read entire email histories silently.

Mitigation Strategies:

Mitigation	Implementation Details	Priority
Privileged Account Management (PAM)	Use Just-in-Time (JIT) access to eliminate permanent memberships in high-privileged groups (e.g., Domain Admins). Ensure admins use separate, non-privileged accounts for daily tasks like email.	Critical
Multi-Factor Authentication (MFA)	Enforce phishing-resistant MFA (FIDO2/WebAuthn) for all administrative actions and cloud console access. Ensure MFA is required for adding new credentials or devices to an account.	Critical
User Account Management	Strictly adhere to the Principle of Least Privilege (PoLP). Ensure low-privileged users cannot modify other accounts, group memberships, or security policies. Regularly audit and disable dormant accounts.	**High**
Active Directory Tiering	Implement a Tiered Administration Model (Tier 0 for Domain Controllers, Tier 1 for Servers, Tier 2 for Workstations). This prevents credentials from a highly privileged Tier 0 account from being exposed on a lower-tier system.	High
Application & Feature Control	Disable unnecessary features that bypass standard auth, such as App Passwords in Entra ID or legacy authentication protocols (POP3/IMAP) that do not support modern MFA.	Medium

Network Segmentation	Isolate Domain Controllers and identity servers in a dedicated, highly restricted VLAN. Limit management access to these systems to authorized Secure Admin Workstations (SAW) only.	Medium

5. Defense Evasion (TA0005)

Tactic Observation:

Defense Evasion (TA0005) is the tactical goal where adversaries attempt to avoid detection throughout their compromise. Observing this tactic requires looking for "negatives" (things being disabled) or "masquerading" (things pretending to be something else).

Tactic Description:

Defense Evasion consists of techniques that adversaries use to avoid technical defenses, bypass security controls, and hide their presence throughout the entire attack lifecycle. Unlike other tactics that have a specific start and end point, evasion is continuous—it is the "how" behind almost every other action an attacker takes.

Tactic ID: TA0005

Total Techniques: 26

Typical Phase: Post-Compromise / Continuous Phase

ATT&CK Version: Created 17 October 2018

Technique1: Debugger Evasion(T1622)

Description:

Debugger Evasion (T1622) is a defense evasion technique where malware detects if it is being executed within a debugger or a dynamic analysis

environment. If the malware detects a debugger, it will alter its behavior—typically by crashing, entering an infinite loop, or executing "benign" code—to prevent security researchers and automated sandboxes from reverse-engineering it.

Detection Methods:

- **Monitor for Anti-Debug API Calls** such as IsDebuggerPresent, CheckRemoteDebuggerPresent, and NtQueryInformationProcess (specifically looking for ProcessDebugPort).
- **Track unusual process timing checks** by identifying the frequent use of the RDTSC instruction or GetTickCount in short bursts to measure execution delays.
- **Audit Registry Queries** that look for keys related to analysis tools, such as HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall searching for "Wireshark," "x64dbg," or "Process Hacker."
- **Identify Exception Handling Manipulation** by monitoring calls to SetUnhandledExceptionFilter or the use of AddVectoredExceptionHandler which can be used to trap "trap" signals.
- **Detect Environment Fingerprinting** where a process enumerates loaded DLLs looking for analysis-specific libraries like dir_watch.dll, sniff_m_u.dll, or dbghelp.dll.
- **Analyze Parent-Child Process Trees** for processes that attempt to spawn themselves as a child or use DEBUG_PROCESS flags during creation.
- **Flag High-Frequency Thread Context Queries** where a process uses GetThreadContext to inspect its own hardware breakpoints (DR0-DR7 registers).
- **Monitor for "Junk Code" or NOP Sleds** that are intentionally designed to cause disassembler misalignment or "sliding" in static analysis tools.

The Real-World Case:

FinFisher (FinSpy): FinFisher's evasion is not just a single check; it is a persistent effort to ensure that if a researcher tries to open it in a debugger (like x64dbg or IDA Pro), the malware either breaks the debugger or changes its own code to hide its true intent.

Mitigation Strategies:

Mitigation	Implementation Details	Priority
Analysis Environment Hardening	Use "cloaked" debuggers and sandboxes (e.g., ScyllaHide, Cuckoo with anti-anti-VM/debug plugins). These tools intercept anti-debug API calls and return "false" values to the malware.	Critical
Static Analysis Prioritization	Analyze suspicious binaries using static methods (disassemblers like IDA Pro or Ghidra) before execution. This allows researchers to identify and "patch out" evasion loops or jump instructions.	High
Endpoint Detection (EDR) Behavioral Rules	Configure EDR to flag processes that perform "Environment Fingerprinting," such as querying the BIOS, checking CPU debug registers, or calling IsDebuggerPresent.	High
Dynamic Binary Instrumentation (DBI)	Use frameworks like Intel PIN or Frida to instrument code at runtime. These can bypass evasion checks by modifying the program's register values or memory state on the fly.	Medium
Memory Scanning	Frequently scan process memory for unpacked code. Malware often uses evasion to delay unpacking; scanning memory directly can find the "payload" even if the evasion routine is active.	Medium

Technique2: Delay Execution (T1678)

Description:

Delay Execution (T1678) is a defense evasion technique where an adversary incorporates a deliberate pause into their malicious code to bypass security controls. The primary goal is to wait out automated analysis environments—such as sandboxes or emulators—which usually have a limited "timeout" window (often 2 to 5 minutes) to scan a file.

Detection Methods:

- **Monitor High-Value Sleep API Calls:** Monitor for processes calling Sleep, SleepEx, NtDelayExecution, or nanosleep with values exceeding common thresholds (e.g., > 180,000 milliseconds / 3 minutes).
- **Identify "API Hammering" Behavior:** Detect processes that call benign Windows API functions (like GetTickCount, GetProcessHeap, or GetSystemInfo) thousands of times in a tight loop. This "junk activity" is often used to create a natural delay without using a suspicious Sleep function.
- **Track Built-in Utility Abuse:** Alert on the use of ping.exe or timeout.exe to create artificial delays. A common pattern is cmd.exe /c ping 127.0.0.1 -n 100 > nul, which creates a ~100-second pause.
- **Identify Timing-Check Loops:** Look for processes that compare the system time before and after a delay. If the "actual" time elapsed is much shorter than the "requested" time (common in sandboxes that skip sleeps), the malware may terminate or change behavior.
- **Analyze Unusual Parent-Child Timings:** Flag instances where a parent process (like a Word document macro) spawns a child process that remains completely idle for several minutes before suddenly initiating network connections or file encryption.
- **Detect "Sleep Obfuscation" in Memory:** Monitor for threads that change their own memory permissions (from RWX to RW) and then call a wait function. This technique hides the malware's malicious code in non-executable memory during the delay to avoid memory scanners.
- **Monitor for Environmental Triggers:** Use behavioral analytics to flag malware that waits for specific user actions that a sandbox might not simulate, such as Document_Close events in Office files or reaching a specific number of mouse clicks.

The Real-World Case:

SolarWinds (SUNBURST Malware) APT29 (Nobelium):

The attackers injected a backdoor, known as **SUNBURST**, into the legitimate SolarWinds Orion software update. To ensure they weren't caught by the automated testing environments used by SolarWinds or their customers, they implemented a sophisticated "dormancy" period.

- **The 14-Day Sleep:** Once the malicious update was installed on a victim's server, the SUNBURST malware was programmed to stay completely dormant for a period of **12 to 14 days**.
- **Why it worked:** Most sandboxes only analyze a file for 2 to 10 minutes. By waiting two weeks, the malware ensured that the software had passed all initial "quarantine" periods and was running in a production environment before it even attempted to contact its Command and Control (C2) server.
- **Contextual Checks:** In addition to the time delay, the malware checked if it was running in a "small" environment (like a test lab) by looking at the hard drive size and the number of running processes. If the environment looked fake, it would never "wake up."

Mitigation Strategies:

Mitigation	Implementation Details	Priority
Sandbox Time-Patching	Configure automated analysis environments to intercept Sleep() and NtDelayExecution calls, forcing them to return immediately or "fast-forwarding" the system clock.	Critical
Environmental Spoofing	Hardening sandboxes to appear as "real" systems (large disk size, realistic file names, active processes) to ensure malware doesn't use delays to wait for a "better" target.	High
Human Interaction Simulation	Use analysis tools that simulate random mouse movements, button clicks, and window resizing to trigger malware that waits for human presence before executing.	**High**
EDR Behavioral Logic	Configure EDR rules to flag "Dormant-to-Active" transitions—where a long-quiet process suddenly initiates network connections or writes to sensitive directories.	High
Scripting Policy	Use GPO or ASR (Attack Surface Reduction) rules to prevent Office macros or standard users from calling timeout.exe or ping.exe for the purpose of creating pauses.	Medium
Memory Forensics	Use tools like Volatility to scan for "Sleeping" processes that have suspicious memory permissions (e.g., RWX) while in a wait state.	Medium

Post-Execution Monitoring	Extend the "Watch period" for new files beyond the initial sandbox run; maintain a log of "Recently Introduced Binaries" and monitor them for 14+ days.	Medium

6. Credential Access (TA0006):

Tactic Observation:

Credential Access (TA0006) is the tactical goal where adversaries attempt to steal account names and passwords, hashes, or tokens. In modern environments, this often shifts from "guessing passwords" to "stealing identities" already stored in memory or databases.

Tactic Description:

Adversaries use this tactic because having a valid set of credentials is far more effective than trying to exploit a software vulnerability. Once an attacker has legitimate credentials, their activity looks like normal user behavior, making it extremely difficult for security tools to distinguish between a "real user" and a "hacker."

Tactic ID: TA006

Total Techniques: 15

Typical Phase: Post-Compromise / Exploitation.

ATT&CK Version: Created 17 October 2018

Technique1: Brute Force (T1110)

Description:

Brute Force (T1110) is a technique under the Credential Access (TA0006) tactic. It involves an adversary systematically attempting to gain access to an account by guessing passwords, hashes, or PINs. Rather than exploiting a

software bug, the adversary exploits weak passwords, predictable patterns, or lack of account lockout policies.

Sub-Techniques:

1. ・**T1110.001: Password Guessing** The adversary tries many passwords against a single account. This is high-risk for the attacker because it often triggers account lockouts quickly.
2. **T1110.003: Password Spraying** This is the most common "stealth" method. The attacker tries a single, common password (like Summer2025!) against thousands of different usernames. This avoids locking out any single account and often stays below the detection threshold of security tools.
3. **T1110.004: Credential Stuffing** Adversaries take lists of usernames and passwords leaked from previous data breaches (from other companies) and "stuff" them into the login page of their current target, hoping the user reused their credentials.
4. **T1110.002: Password Cracking** An **offline** attack. The adversary steals a file containing password "hashes." They then use powerful hardware (GPUs) to guess millions of passwords per second locally until they find a match. This generates zero logs on the victim's network.

Detection Methods:

- **Monitor for a high frequency of failed authentication attempts** (Windows Event ID 4625) targeting a single account within a short timeframe, indicating password guessing.
- **Watch for "Password Spraying" patterns**, characterized by a single failed login attempt across many unique usernames from the same source IP or user-agent.
- **Detect "Success-after-Failure" sequences**, where a long string of failed attempts from a specific remote host is immediately followed by a successful logon (Event ID 4624).
- **Look for multiple account lockouts** (Windows Event ID 4740) occurring simultaneously across the network, which often signals an automated brute-force tool in operation.

- **Review application and cloud logs** (e.g., Entra ID Result 50126) for unusual sign-in activity, such as logins from "Impossible Travel" locations or unexpected ASNs/VPNs.
- **Audit Kerberos pre-authentication failures** (Event ID 4771), which can indicate an adversary attempting to guess passwords or perform "AS-REP Roasting" without triggering standard logon failure events.
- **Detect unauthorized use of administrative tools** or scripts (like Hydra, Medusa, or custom Python modules) that interface with common authentication ports like SSH (22), RDP (3389), or SMB (445).
- **Monitor for unusual User-Agent strings** in web and API logs, identifying automated bots that do not match the organization's standard browser or application profiles.
- **Watch for failed logins to non-existent or "decoy" accounts**, such as admin, guest, or root, which are frequently targeted by generic brute-force script

The Real-World Case:

Midnight Blizzard (APT29) breach of Microsoft:

In late 2023, the Russian state-sponsored actor Midnight Blizzard targeted Microsoft's internal corporate environment. Instead of using a complex zero-day exploit, they used a "low and slow" brute-force approach.

Mitigation Strategies:

Mitigation	Implementation Details	Priority
Phishing-Resistant MFA	Implement FIDO2 or WebAuthn tokens for all external access (VPN, SaaS, Email). Disable SMS-based MFA due to SIM-swapping risks.	Critical
Conditional Access Policies	Block authentications from non-compliant devices, unmanaged IPs, or "Impossible Travel" locations (e.g., login from two different continents in 1 hour).	Critical
Legacy Protocol Removal	Disable old protocols like POP3, IMAP, and SMTP AUTH that bypass modern security defaults and do not support MFA prompts.	High

Smart Account Lockouts	Configure "Smart Lockout" (Azure/Entra ID) to lock the attacker's specific IP while allowing the legitimate user to continue using their account with MFA.	High
Rate Limiting & Throttling	Set strict thresholds at the Application Gateway or WAF level to limit the number of auth requests per second from a single IP or session.	High
Password Complexity Policies	Enforce passphrases of $15+$ characters and use a "Banned Password List" to prevent common choices like Company2025! or Password123.	Medium
Environmental Deception	Deploy Decoy Accounts (Honeypots) with no real permissions. Any login attempt on these accounts should trigger an immediate "Severity 1" incident.	Medium
User Risk Self-Service	Enable automated "Self-Service Password Reset" (SSPR) that triggers only when the user passes an additional MFA challenge after a suspicious event.	Medium

Technique2: Network Sniffing (T1040)

Description:

Network Sniffing (T1040) is a technique used by adversaries to capture and monitor data packets as they traverse a network. By intercepting traffic, attackers can steal sensitive information such as cleartext credentials, session tokens, and configuration details about the network environment.

Detection Methods:

- **Monitor for network interface mode changes**, specifically alerting when a NIC enters "Promiscuous Mode" (e.g., Linux kernel logs showing device entered promiscuous mode or Windows Event ID 4624/4688 with related command-line arguments).
- **Watch for the execution of known packet capture utilities**, such as tcpdump, Wireshark, tshark, or Windump, especially when launched by non-administrative users or during non-maintenance windows.
- **Detect unauthorized "Traffic Mirroring" or "Packet Mirroring" configurations** in cloud environments (AWS VPC Mirroring, Google Cloud

Packet Mirroring, or Azure vTAP), which can be used to exfiltrate raw traffic to an attacker's instance.

- **Identify "Adversary-in-the-Middle" (AiTM) indicators**, such as an unusual volume of ARP traffic or "Gratuitous ARP" messages where a single MAC address attempts to associate itself with multiple IP addresses (ARP Spoofing).
- **Look for "MAC Flapping" on network switches**, where the same MAC address is seen appearing on different physical ports in a short period, often indicating a MAC flooding attack intended to force the switch into hub-like broadcast mode.
- **Review Windows audit logs for built-in diagnostic tool abuse**, such as the use of netsh trace start or the pktmon utility to capture local network traffic for malicious analysis.
- **Monitor for unauthorized LLMNR, NBT-NS, or MDNS traffic**, which is often used by tools like Responder to sniff and intercept credential hashes via name service resolution poisoning.
- **Track the creation of raw sockets** (e.g., AF_PACKET on Linux) by suspicious processes, which allows an application to bypass the standard TCP/IP stack and read every packet on the wire.
- **Analyze network latency and jitter anomalies**, as traffic redirection through an intermediary "sniffing" node (AiTM) often causes a measurable increase in Round Trip Time (RTT) and packet processing delays.

The Real Case:

APT28 (Fancy Bear/Forest Blizzard): The Responder Attacks

The Russian threat group **APT28** is the primary example cited by MITRE for this technique. They frequently use the open-source tool **Responder** to perform a combination of sniffing and poisoning.

- **The Scenario:** Once inside a network, APT28 deploys Responder to listen for broadcast name resolution requests (LLMNR, NBT-NS, and mDNS).
- **The Sniffing Action:** When a user's computer tries to find a network resource that doesn't exist (like a mistyped file share), it broadcasts a request. APT28's sniffing tool "hears" this and responds by pretending to be that resource.
- **The Outcome:** The victim's machine automatically sends the user's **NTLMv2 password hashes** to the attacker. APT28 then cracks these hashes offline to gain legitimate cleartext credentials.

Mitigation Strategies:

Mitigation	Implementation Details	Priority
Encrypt Sensitive Information (M1041)	Enforce TLS/SSL (HTTPS) and SSH for all internal and external communications. Use Kerberos for authentication and ensure legacy unencrypted protocols (like HTTP, Telnet, FTP, and LDAP) are disabled or tunneled through IPsec.	Critical
Multi-Factor Authentication (M1032)	Implement MFA across all services. Even if an attacker sniffs a cleartext password, they cannot reuse it for lateral movement or persistence without the second factor.	Critical
User Account Management (M1018)	Restrict Traffic Mirroring Permissions. In cloud environments (AWS, Azure, GCP), ensure users do not have permissions to create or modify traffic/packet mirrors (e.g., VPC Traffic Mirroring or vTAP) unless explicitly required for their role.	High
Network Segmentation (M1030)	Limit Broadcast Domains. Use VLANs and micro-segmentation to deny direct access to broadcast/multicast traffic. This prevents attacks like LLMNR/NBT-NS poisoning and SMB Relay by keeping sniffing contained to a small segment.	High
Physical/Wireless Security	Secure Network Ports & Wi-Fi. Disable unused physical ethernet ports in open areas and enforce WPA3 or Enterprise-grade (802.1X) authentication for wireless networks to prevent unauthorized "on-wire" access.	Medium
VPN Usage	Encrypt Untrusted Traffic. Require the use of a VPN for any traffic traversing untrusted or public networks to ensure end-to-end encryption and prevent "coffee-shop" style packet sniffing.	Medium

7. Discovery (TA0007)

Tactic Objective:

The primary objective of Discovery is to gather detailed information about the internal environment of the compromised system, such as system configurations, network resources, user accounts, and services running.

Tactic Description:

The Discovery tactic focuses on how adversaries gather information about a system or network environment after gaining initial access. The goal is to map the environment, identify assets, and locate vulnerabilities for further exploitation. These techniques help adversaries observe the environment and orient themselves before deciding how to act. They also allow adversaries to explore what they can control and what's around their entry point in order to discover how it could benefit their current goal. This tactic is crucial as it provides attackers with the knowledge, they need to further their operations, whether that's moving laterally, or extracting valuable information.

Tactic ID: TA0007

Total Techniques: 26

Typical Phase: Post-compromise

ATT&CK Version: Created on 17 October 2018

Technique1: File and Directory Discovery (T1083)

Description:

File and Directory Discovery is a technique where adversaries attempt to enumerate the files and directory structures on a compromised system. Unlike others, this is an information-gathering step. Its purpose is to help the attacker find sensitive data, configuration files, or credentials that will allow them to move laterally or complete their mission. Using this information, it also helps to frame follow-on behaviors, including whether or not the adversary fully infects

the target and/or attempts specific actions. The adversaries using built-in commands (like ls, find, tree) or native APIs to map file systems, locate sensitive data (documents, keys, config files), and understand the system structure to plan further actions

Common Commands: On macOS, File and Directory Discovery are primarily performed through the Bash shell.

- **ls -la:** Lists all files, including hidden files, in long format with permissions and ownership.
- **pwd:** "Print Working Directory"—shows the absolute path of the current folder.
- **ls -R:** Recursively lists every file in every subdirectory from the current location.
- **find /Users -name "*.plist"**: Searches the Users directory for .plist files, which often contain application configurations and preferences.
- **locate:** Uses a pre-built database to find files quickly.

Real world Example:

XCSSET (2020)

- XCSSET is unique because its primary infection vector is Xcode projects. It doesn't just infect an app; it infects the "source code" environment so that any app a developer builds becomes a carrier for the malware.
- It works as follows:
    - The malware hides within the configuration files (project.pbxproj) of a local Xcode project. Developers inadvertently infect their systems when they open and build an infected project.
    - The malicious payload executes as a seemingly benign build process rides on legitimate developer actions.
    - Once a developer's system is compromised, XCSSET attempts to find and infect other local Xcode projects, allowing it to spread to new projects and potentially downstream to the end-users of the apps created from them.

Detection Methods:

- Monitor for commands that use grep, find, or mdfind combined with sensitive keywords
- Monitor for file system-related syscalls
  like getdirents, stat, fstat, read, open on directories.
- Alert on processes reading many file metadata (e.g., stat calls) across broad, non-standard directories
- Watch for large numbers of files being copied or archived into single staging directories before exfiltration.
- Configure rules to alert on access to sensitive files/directories
- Establish baseline file/directory access for normal user/system processes and alert on significant deviations
- Endpoint Detection & Response can provide deep visibility into command-line arguments and parent-child process relationships, highlighting suspicious parent processes.

Mitigation Strategies:

Mitigation	Implementation	Priority
Standard User Accounts	Revert daily-use accounts to "Standard" status. Admin credentials should only be used via sudo or for configuration changes.	Critical
Tighten TCC Permissions	Ensure Full Disk Access (FDA) is granted only to essential system tools.	Critical
Encrypted SSH Keys	Always use passphrases for SSH keys. This ensures that even if id_rsa is discovered, the raw key cannot be used.	High
System Integrity Protection	Ensure SIP is enabled. This prevents even root users from listing/reading certain system-protected paths.	High
Script Execution Controls	Restrict script execution capabilities	High
Canary Files	Place "Honeytoken" files (e.g., Passwords.txt) in ~/Documents and set an EDR alert for any READ action on that specific path.	Medium

Technique2: Process Discovery (T1057)

Description:

Process Discovery describes how an adversary gets a list of running processes on a system. Information obtained could be used to gain an understanding of common software/applications running on systems within the network. Such information can also be used to map network activity, find security tools, or identify targets for later actions like privilege escalation. Administrator or otherwise elevated access may provide better process details. On macOS, this is a critical step for attackers to identify security software, find browsers to hijack, or determine if they are being analyzed in a virtual machine (VM). Adversaries may use the information from Process Discovery during automated discovery to shape follow-on behaviors, including whether or not the adversary fully infects the target and/or attempts specific actions.

Common Commands:

- **ps aux:** Provides a detailed, comprehensive list of all running processes owned by all users, displayed in a user-oriented format.
- **pstree**: Shows processes in a tree hierarchy.
- **top**: Provides a real-time, dynamic view of running processes and system resource usage (CPU, memory)
- **top -l 1:** Displays a snapshot of CPU/Memory usage.
- **pgrep**: Used to look up processes based on name or other attributes, often used by malware to identify specific processes, such as security software, that they might want to avoid or disable.

Real world Example:

DazzleSpy (Hong Kong in late 2021 and early 2022)

- DazzleSpy is a "full-featured" backdoor, meaning it gives the attacker almost total control over the Mac once it's installed.
- It works as follows:

- The malware was distributed through "watering hole" attacks, where attackers either compromised legitimate websites (like a pro-democracy radio station's news site) or created fake websites to lure specific targets.
- The infection chain typically exploited two vulnerabilities in macOS (which have since been patched by Apple):
  - **CVE-2021-1789:** A vulnerability in the WebKit engine (used by the Safari browser) that allowed for remote code execution when a user visited a malicious webpage.
  - **CVE-2021-30869**: A local privilege escalation flaw in the macOS kernel that allowed the malware to gain root privileges and escape the browser's sandbox.

Detection Methods

- Monitor for the execution of command-line utilities like ps.
- Analyze the full command line and the parent process.
- Use security software that is capable of detecting anomalous behavior in processes.
- Utilize the macOS Unified Log to monitor for an unusual volume or sequence of process-related API calls
- Alert on anomalous frequency or timing of process discovery activities
- Implement allow- or deny-lists for known benign versus malicious command patterns
- Look for the *behavioral chain* rather than just a single command. For example, an application that lists processes and immediately attempts to connect to a command-and-control server or access sensitive files is highly suspicious.

Mitigation Strategies

Mitigation	Implementation	Priority
App Sandboxing	Ensure all internal and third-party apps use the App Sandbox.	Critical

Standard User Roles	Remove administrative privileges from end-user accounts.	High
System Integrity Protection	Ensure SIP is enabled. This prevents even root users from listing/reading certain system-protected paths.	High
Principle of Least Privilege	Restrict user and process privileges to the minimum necessary for their function	High
Behavioral Analysis	Monitor for anomalous activity that might indicate an adversary is attempting to enumerate processes.	HIgh
Audit Logging	Enable macOS AuditD or Unified Logging to track the use of system binaries by non-admin users.	Medium
Hardened Runtime	Enable "Hardened Runtime" for all locally developed binaries.	Medium

8. Lateral Movement (TA0008)

Tactic Objective:

The objective of Lateral Movement is to discover and access high-value systems, sensitive accounts, or critical data, allowing them to pivot across multiple systems and accounts to achieve their ultimate goals, such as data theft, financial gain, or expanding control.

Tactic Description:

Lateral Movement focuses on how an adversary moves from one compromised Mac to another within a network. MacOS has its own unique set of native protocols and services that attackers abuse. It describes how adversaries navigate through a network once they have gained an initial access. Lateral movement allows adversaries to search for key targets, gather information, and gain access to additional systems in the network. On macOS, this often involves abusing built-in remote management tools, exploiting shared sessions, or using legitimate credentials to access other systems.

Tactic ID: TA0008

Total Techniques: 7

Typical Phase: Post-compromise

ATT&CK Version: Created on 17 October 2018

Technique1: Lateral Tool Transfer(T1570)

Description:

Lateral Tool Transfer describes the process of an adversary copying malware, scripts, or utility tools from one compromised Mac to another within the same network. To do this it often uses built-in file sharing, protocols (ftp), or legitimate remote access software (like AnyDesk) to stage malware, spread tools, and support further attacks. Adversaries may copy files between internal victim systems to support lateral movement using inherent file sharing protocols. The Lateral Tool Transfer technique involves manually or automatically moving the necessary malware, scripts, or legitimate admin tools (like remote access tools) to other internal systems.

Real world Example:

OceanLotus(APT32)

- OceanLotus, also known as APT32, is a sophisticated, state-aligned cyber espionage group widely assessed to be linked to the government of Vietnam
- This group focuses on intelligence collection that supports Vietnam's strategic political and economic interests.
- In March 2021, it was reported that the group's operations were impacted by a fire at an OVHcloud data centre in France.
- How it works:
    - The group uses sophisticated methods to gain an initial foothold in a victim's system.
    - Once initial access is achieved (often requiring user execution of a malicious file or macro), the malware is installed.

- The malware establishes communication with C2 servers using standard protocols like HTTP/HTTPS to avoid detection.
- Data is typically compressed and encrypted before being exfiltrated back to the attackers' infrastructure.
- After gaining a foothold, the attackers perform network reconnaissance to identify other valuable systems, move laterally across the network using stolen credentials.

Detection Methods:

- Look for the execution of common macOS utilities that can be used for file transfer.
- Correlate file creation and modification events with transfer activity.
- Monitor for new files appearing in directories that are commonly used for staging malware because they don't require high permissions to write to.
- Monitor for files created in hidden folders (e.g., ~/.local/, ~/Library/.hidden/) immediately following an inbound SSH connection.
- Use network intrusion detection systems (NIDS) to identify unusual data transfer over standard protocols.
- Use the built-in log command to hunt for evidence of transfers that have already occurred.
- Look for spikes in internal traffic on port 22 (SSH) or port 445 (SMB) between workstations that don't usually communicate.

Mitigation Strategies

Mitigation	Implementation	Priority
Disable Unnecessary Services	Turn off "Remote Login" (SSH) and "Remote Management" (ARD) in System Settings for non-admin users.	Critical
Network Segmentation	Implement "Client Isolation" or VLANs to prevent MacBook-to-MacBook communication	Critical

Multi-Factor Authentication	Enforce MFA on all remote access	Critical
Admin Share Hardening	Disable ADMIN$ share where possible	High
SSH Hardening	Set PasswordAuthentication no in /etc/ssh/sshd_config and enforce the use of SSH keys.	High
MFA for SSH	Integrate a PAM module (like Duo or Google Authenticator) for local SSH sessions to block stolen key usage.	Medium
SSH Key Protection	Transition users to storing SSH keys in the macOS Secure Enclave (via tools like *Secretive*) so keys cannot be exported.	Low

Technique2: Remote Service Session Hijacking (T1563)

Description:

Remote Service Session Hijacking is a technique where adversaries take control of an already existing, legitimate user's session with a remote service (like SSH) to move laterally within a network. Instead of creating a new session using stolen credentials, an attacker commandeers an active or disconnected session, often without the legitimate user being notified. This allows them to bypass authentication mechanisms and inherit the permissions of the hijacked user. Users may use valid credentials to log into a service specifically designed to accept remote connections, such as telnet, SSH, and RDP. When a user logs into a service, a session will be established that will allow them to maintain a continuous interaction with that service.

Sub Techniques:

- **T1563.001 - SSH Hijacking:** SSH Hijacking is a sub-technique within the MITRE ATT&CK framework where an adversary takes control of an already

existing, legitimate Secure Shell (SSH) session to move laterally within a network. This differs from simply using SSH with stolen credentials because it involves commandeering an active session rather than initiating a new one

- **T1563.002 - RDP Hijacking:** Adversaries use this technique to hijack an already active or a disconnected RDP session that a legitimate user has established with a remote system. Instead of creating a new session (which might require credentials and trigger new authentication logs), the attacker effectively "steals" the existing session, often without the legitimate user being prompted or even noticing the takeover.

Real world Example:

CircleCI Breach (2022) - SSO Session Hijacking

- The 2022 CircleCI breach involved attackers using malware on an engineer's laptop to steal a valid, 2FA-backed SSO session cookie, enabling them to impersonate the employee and access production systems, leading to the exfiltration of customer secrets
- The Scenario: A developer’s macOS laptop was infected with malware that was not detected by traditional antivirus software.
- The Hijack: The malware specifically targeted browser session cookies. By stealing a valid, 2FA-backed SSO (Single Sign-On) session cookie from the developer's browser, the attacker was able to "impersonate" the employee.
- Lateral Movement: Because the hijacked session was already authenticated with MFA, the attacker accessed CircleCI's internal GitHub and production systems without triggering any new login alerts.
- Impact: This resulted in the theft of customer secrets and environmental variables stored on the platform.

Detection Methods

- Alert on multiple active sessions for a single user account occurring simultaneously across different systems or IP addresses.

- Monitor system and application logs for unusual activity.
- Implement tools to detect sudden changes in session properties that should remain static.
- Use endpoint detection and response (EDR) or host-based intrusion detection systems (HIDS) to monitor ongoing activity.
- Monitor for suspicious child processes spawned from a legitimate remote service session
- Monitor for unusual data flows or network communication initiated by processes that don't typically use the network.
- Employ behavioral analytics to establish a baseline of "normal" user activity.

Mitigation Strategies

Mitigation	Implementation	Priority
Disable Unused Services	Turn off unnecessary remote services (e.g., SSH, Screen Sharing, Remote Management) to reduce the attack surface.	High
Strong User & Privileged Account Management	Limit remote access to only necessary users and restrict privileged accounts (like root or admin) from direct remote login.	Medium
Enforce Secure Password Policies	Mandate strong, unique passwords and ensure secure key pairs for services like SSH to prevent credential-guessing or cracking.	Medium
Operating System Configuration	Configure shorter session timeouts and automatic logouts after inactivity to minimize the window of opportunity for a hijacking event.	Medium
User Education	Train users to avoid public Wi-Fi for sensitive transactions, use VPNs, and log out of sessions when done to prevent client-side vulnerabilities.	Low

9. Collection (TA0009)

Tactic Objective:

The objective of the Collection tactic is to locate and gather information of interest from a target environment to fulfill the adversary's ultimate goal. In the context of macOS, this is the phase where the attacker shifts from simply "being in the system" to "stealing the value" within it.

Tactic Description:

The Collection describes the techniques adversaries use to identify, gather, and stage data of interest within a compromised macOS environment. Once an adversary has gained access to an environment, they may search for and collect specific types of valuable information, which can include sensitive documents, credentials, configuration files, business data, and system information. this tactic is particularly focused on accessing sensitive user data stored in the keychain, browser history, or local files, as well as capturing live user activity through the OS's multimedia and input capabilities.

Tactic ID: TA0009

Total Techniques: 14

Typical Phase: Post-compromise

ATT&CK Version: Created on 17 October 2018

Technique1: Clipboard Data (T1115)

Description:

Clipboard Data describes adversaries stealing sensitive info (passwords, tokens, data) by monitoring or replacing user clipboard contents on macOS (using pbpaste), exploiting the cross-application access needed for functionality, often via malware harvesting data after user copy/paste actions or by replacing copied data to insert malicious info like fake cryptocurrency wallet addresses. On macOS, this is a highly effective way to steal sensitive data like

passwords from password managers, 2FA recovery codes, or private messages that never touch the hard drive as a file.

Real world Example

MacSpy (2017)

- MacSpy is a type of spyware that specifically targets macOS systems and is known to steal data from the clipboard.
- It works as follows:
  - MacSpy runs in the background, continuously monitoring the clipboard for newly copied content.
  - The collected clipboard data is gathered into temporary files and sent periodically to a remote C&C server managed by the attackers.
  - The malware attempts to operate without leaving a digital trace for the operator and deletes the temporary files containing the data after exfiltration.
  - MacSpy establishes persistence on the infected Mac by creating a Launch Agent, ensuring it runs every time the system starts up or a user logs in.

Anatomy of the Attack (Technical View)

A typical "Living off the Land" attack on the macOS clipboard looks like this simple script often found in malware payloads:

```
# A simple bash loop that steals clipboard data every 10 seconds

while true; do

# Get the current clipboard content

DATA=$(pbpaste)

# If data isn't empty, append it to a hidden log file

if [ ! -z "$DATA" ]; then

echo "$(date): $DATA" >> ~/.local/.clipboard_history
```

fi

sleep 10

done

Detection Methods

- visually inspect the current clipboard content by opening a Finder window, selecting Edit from the top menu bar, and then choosing Show Clipboard.
- The pbpaste command in the Terminal can be used to output the current text contents of the clipboard.
- If you suspect your Mac is being monitored, you can run this command in Terminal to see if any background process is currently trying to "squat" on your clipboard tools:
- # This looks for any running processes that have 'pbpaste' in their command string
- ps -ef | grep pbpaste
- Track any instance of the /usr/bin/pbpaste binary being called
- Investigate whenever pbpaste is launched by suspicious parents, such as: osascript (AppleScript execution).
- Scan for hidden files in /tmp/, /var/tmp/, or ~/Library/ that contain text patterns matching copied data or common regex for sensitive info

Mitigation Strategies

Mitigation	Implementation	Priority
Use a Password Manager with Auto-Clear	Utilize password manager applications (e.g., 1Password) that automatically clear sensitive data from the clipboard after a short, configurable timeout	High
Enable/Maintain System Integrity Protection (SIP)	Ensure macOS's built-in SIP feature is enabled to protect core system processes and prevent malicious software from easily tampering with the OS, including potential clipboard snoopers	High

Disable Universal Clipboard	Uncheck "Allow Handoff between this Mac and your iCloud devices" to prevent data from syncing to potentially compromised mobile devices.	High
Use "Secure Paste"	Utilize the upcoming "Secure Paste" functionality in macOS 16, which allows pasting content into an active app without triggering a privacy alert, reducing the risk of surreptitious access.	Medium
Automated Clipboard Clearing	Configure password managers and sensitive apps to clear the clipboard automatically after a short duration (e.g., 30–60 seconds).	Medium
Regularly Clear Clipboard Manually	Adopt the habit of manually clearing your clipboard after copying sensitive information by copying a non-sensitive item (e.g., a single space).	Low

Technique2: Screen Capture (T1113)

Description:

Adversaries use screen capture technique to collect additional information about a target by taking screenshots or recording a user's screen. Adversaries may use various methods on macOS to capture information displayed on a compromised system's screen. This can reveal sensitive data such as: Applications running in the foreground; User data, credentials, or other sensitive information; User intent by capturing screenshots around mouse clicks.

Real world Example:

FruitFly

- FruitFly (S0277) is one of the most persistent and enigmatic examples of macOS malware.
- First publicly detailed in 2017, it is believed to have operated undetected for over a decade (possibly since 2003).

- It is primarily a Remote Access Trojan (RAT) designed for long-term, high-granularity surveillance.
- It works as follows:
    - Capture screens and record keystrokes.
    - Access webcams and microphones to record audio and video.
    - Control the mouse and simulate user interactions.
    - Steal sensitive data, including financial and medical records.

Detection Methods

- Use Finder's search feature (Command + F) and the query string kMDItemIsScreenCapture:1 to find all recent screenshots, regardless of where they are saved.
- Open the Activity Monitor and observe the running processes. When a standard macOS screenshot is taken (e.g., using Shift + Command + 3), a process named screencapture briefly appears in the process list.
- Monitor for any execution of /usr/sbin/screencapture.
- Regularly review System Settings > Privacy & Security > Screen & System Audio Recording
- Since macOS Monterey, a purple dot in the Menu Bar indicates the screen is being recorded, while an orange dot indicates the microphone is active.
- If the screen is being captured, a message stating "Your screen is being observed" will appear on the Mac’s lock screen

Mitigation Strategies

Mitigation	Implementation	Priority
Technical Restriction	Disable native macOS screenshot/recording functionality via MDM or configuration profiles	High
Restrict "Screen Recording" Permissions	Audit Privacy & Security > Screen & System Audio Recording. Remove any apps that do not have a legitimate, business-justified reason to see your desktop.	High

Monitoring & Auditing	Monitor and log screen recording activities and file transfers to detect potential insider threats.	Medium
Disable "Universal Control" & Screen Sharing	Turn off Screen Sharing and Remote Management unless actively required for IT support. Disable "Handoff" to prevent visual data leaks across Apple devices.	Medium
App-Level Protection (Sensitive Data)	Apps (like Banking or Password Managers) should use the NSWindow.sharingType = .none API to prevent their specific window from appearing in screenshots/recordings.	Medium
"Purple Dot" Awareness Training	Train users to look for the purple icon/dot in the Menu Bar. If it appears and they aren't in a meeting, it indicates an active (potentially rogue) recording.	Low

10. Command and Control (TA0011):

Tactic Objective:

To establish and maintain communication between compromised systems and the attacker in order to send commands, receive instructions, and exfiltrate data.

Tactic Description:

Command and Control consists of techniques that adversaries may use to communicate with systems under their control within a victim network. Adversaries commonly attempt to mimic normal, expected traffic to avoid detection. There are many ways an adversary can establish command and control with various levels of stealth depending on the victim’s network structure and defenses .

Tactic ID: TA0011

Total Techniques: 18

ATT&CK Version: Created 17 October 2018

Last modified: 25 April 2025

Technique 1: Data Encoding (T1132)

Description:

Adversaries may encode data to make the content of command and control traffic more difficult to detect. Command and control (C2) information can be encoded using a standard data encoding system. Use of data encoding may adhere to existing protocol specifications and includes use of ASCII, Unicode, Base64, MIME, or other binary-to-text and character encoding systems.

Created: 31 May 2017

Last modified: 24 oct 2025

Version: 1.3

Platform: Linux, Windows, macOS

Tactics: Command and Control

Sub-techniques:

- **T1132.001 :** Standard Encoding **–** use of **common encoding methods (such as Base64 or URL encoding)** to obscure data or commands and evade detection.

- **T1132.002 :** Non-Standard Encoding **–** involves using **custom, uncommon, or proprietary encoding methods** to conceal data or commands, making detection by standard security tools more difficult.

Last modified – 24 Oct 2025
Created : 14 March 2020
Version – 1.1
Platform - Linux, macOS

Real world Example:

- **FIN7 cybercrime group** encoded stolen payment-card data before exfiltration to evade DLP systems.
- **Emotet malware** encoded C2 commands and stolen credentials using Base64 to avoid signature-based detection.

Detection Methods:

- Monitor unusual encoded strings (e.g., long Base64-like text) in network traffic
- Detect abnormal outbound data volume or timing patterns
- Use IDS/IPS and DLP tools to flag encoded data transfers
- Analyze command-and-control traffic for non-standard encoding
- Perform log and packet inspection for suspicious payload formats

Mitigation Strategies:

Mitigation	Implementation	Priority
Data Loss Prevention (DLP)	Inspect outbound traffic for encoded sensitive data (Base64, URL encoding)	**Critical**
Network Traffic Analysis	Detect abnormal encoded payload patterns	**High**
Egress Filtering	Restrict unauthorized outbound data transfers	**High**
Protocol Enforcement	Allow encoding only in approved applications/services	**Medium**
Endpoint Monitoring	Detect processes performing	**Medium**

(EDR)	suspicious encoding before transmission

Technique 2: Encrypted Channel (T1573)

Description:

Adversaries may employ an encryption algorithm to conceal command and control traffic rather than relying on any inherent protections provided by a communication protocol. Despite the use of a secure algorithm, these implementations may be vulnerable to reverse engineering if secret keys are encoded and/or generated within malware samples/configuration files.

Created: 16 March 2020

Last modified: 24 oct 2025

Version: 1.2

Platform: Linux, Windows, macOS

Tactics: Command and Control

Sub-techniques:

- **T1573.001 :** Symmetric Cryptography **–** use of a **shared secret key** to encrypt and decrypt data for secure attacker communication or data exfiltration.
- **T1573.002 :** Asymmetric Cryptography **–** is the use of **public–private key encryption** to securely exchange data or commands without sharing a secret key.

Last modified – 24 Oct 2025
Created : 16 March 2020
Version – 1.2

Platform - Linux, macOS

Real World Example:

- **Cobalt Strike beacons** use HTTPS/TLS encryption to hide attacker communications inside normal web traffic.
- **DarkHotel APT** encrypted C2 traffic to evade network monitoring in hotel Wi-Fi networks.

Detection Method :

- Monitor unusual encrypted outbound traffic to unknown or rare domains/IPs
- Detect anomalous TLS/HTTPS behavior (self-signed certs, uncommon cipher suites)
- Use SSL/TLS inspection where permitted to analyze encrypted traffic patterns
- Identify beaconing patterns (regular timing, small packet sizes)
- Correlate endpoint logs showing processes initiating unexpected encrypted connections

Mitigation Strategies:

Mitigation	Implementation	Priority
SSL/TLS Inspection	Decrypt and inspect encrypted HTTPS/TLS traffic	**Critical**
DNS Filtering	Block malicious, DGA, and suspicious domains	**Critical**
Network Traffic Analysis	Detect beaconing and anomalous encrypted traffic patterns	**Critical**
Egress Filtering	Restrict outbound connections to approved destinations	**High**
Domain Reputation	Block newly registered or low-reputation domains	**High**

11. Exfiltration (TA0010):

Tactic Objective:

The adversary aims to **collect, transfer, and steal sensitive information** such as personal data, credentials, intellectual property, or financial records from a compromised system or network.

Tactic Description:

Exfiltration consists of techniques that adversaries may use to steal data from your network. Once they've collected data, adversaries often package it to avoid detection while removing it. This can include compression and encryption. Techniques for getting data out of a target network typically include transferring it over their command and control channel or an alternate channel and may also include putting size limits on the transmission.

Tactic ID: TA0010

Total Techniques: 8

ATT&CK Version: Created 17 October 2018

Last modified: 25 April 2025

Technique 1: Data Transfer Size Limits (T1030)

Description:

An adversary may exfiltrate data in fixed size chunks instead of whole files or limit packet sizes below certain thresholds. This approach may be used to avoid triggering network data transfer threshold alerts.

Created: 31 May 2017

Last modified: 24 oct 2025

Version: 1.1

Platform: Linux, Windows, macOS

Tactics: Exfiltration

Real World Example:

- **Magecart attacks** slowly leaked credit-card data in small packets from compromised e-commerce websites.
- **Insider data theft cases** where employees upload small files repeatedly to cloud services to avoid alerts.

Detection method:

- Monitor frequent small outbound data transfers instead of normal large uploads
- Detect beaconing patterns with consistent packet sizes
- Use DLP and network traffic analysis to flag abnormal transfer behavior
- Correlate endpoint processes sending repeated low-volume data
- Analyze timing and frequency anomalies in data exfiltration traffic

Mitigation Strategies:

Mitigation	Implementation	Priority
Network Traffic Analysis	Detect abnormal patterns in small, frequent data transfers	**Critical**
Egress Filtering	Restrict outbound connections to approved destinations	**High**
Data Loss Prevention (DLP)	Monitor for unusual volumes or frequency of sensitive data transfers	**High**
Endpoint Monitoring (EDR)	Track processes generating repeated small data uploads	**Medium**
User Awareness & Access Control	Limit user/application permissions for large or frequent transfers	**Medium**

Technique 2 : Exfiltration Over C2 Channel (T1041)

Description:

Adversaries may steal data by exfiltrating it over an existing command and control channel. Stolen data is encoded into the normal communications channel using the same protocol as command and control communications.

Created : 31 May 2017

Last modified : 24 oct 2025

Version : 2.3

Platform: Linux, Windows, macOS

Tactics: Exfiltration

Real World Example:

- **SolarWinds (SUNBURST) attack** exfiltrated sensitive data through the same C2 channel used for command traffic.
- **APT10 (Cloud Hopper)** used C2 connections to steal intellectual property from managed service providers.

Detection method:

- Detect encrypted C2 sessions with abnormal data upload ratios
- Use network behavior analytics to identify covert data transfer patterns
- Correlate endpoint activity with outbound C2 communications
- Apply DLP tools to flag sensitive data leaving via non-standard channels

Mitigation Strategies:

Mitigation	Implementation	Priority
Network Traffic Analysis	Detect anomalous or high-frequency C2 communication patterns	**Critical**
SSL/TLS Inspection	Decrypt and inspect encrypted C2 traffic	**Critical**
Egress Filtering	Restrict outbound connections to approved domains/IPs	**High**
Endpoint Detection & Response (EDR)	Monitor processes communicating with known or suspicious C2 servers	**High**
Threat Intelligence / Domain Reputation	Block known C2 infrastructure and malicious domains	**High**

12. Impact (TA0040):

Tactic Objective:

The adversary seeks to **manipulate, disrupt, interrupt, or destroy systems and data** to achieve their goals. This may include **data destruction, data manipulation, service disruption, ransomware deployment, or resource hijacking,** leading to **operational downtime, financial loss, loss of data integrity, reputational damage, and reduced trust** in affected systems or organizations**.**

Tactic Description:

Impact consists of techniques that adversaries use to disrupt availability or compromise integrity by manipulating business and operational processes. Techniques used for impact can include destroying or tampering with data. In some cases, business processes can look fine, but may have been altered to benefit the adversaries' goals. These techniques might be used by adversaries to follow through on their end goal or to provide cover for a confidentiality breach.

Tactic ID: TA0040

Total Techniques: 15

ATT&CK Version: Created 14 March 2019

Last modified: 25 April 2025

Technique 1: Data Manipulation (T1565)

Description:

Adversaries may insert, delete, or manipulate data in order to influence external outcomes or hide activity, thus threatening the integrity of the data. By manipulating data, adversaries may attempt to affect a business process, organizational understanding, or decision making.

Created: 02 March 2020

Last modified: 24 oct 2025

Version: 1.1

Platform: Linux, Windows, macOS

Tactics: Impact

Sub-techniques:

- **T1565.001 (Stored Data Manipulation) –** is the unauthorized **modification or tampering of data at rest** (files, databases, logs) to mislead, disrupt operations, or gain advantage.

- **T1565.002 (Transmitted Data Manipulation) –** is the unauthorized **alteration of data while it is being transmitted** over a network to deceive or disrupt the recipient.

- **T1565.003 (Runtime Data Manipulation) –** is the unauthorized **modification of data in memory during execution** to alter program behavior or outputs.

Real World Example:

- **NotPetya malware** corrupted file systems and altered system data, causing massive operational disruption.
- **Financial fraud attacks** where attackers manipulate transaction records to hide stolen funds.

Detection method:

- Monitor unauthorized changes in files, databases, or logs
- Use file integrity monitoring (FIM) to detect unexpected modifications
- Analyze audit logs for abnormal user or admin activity
- Detect sudden data inconsistencies or anomalies in systems

Mitigation Strategies:

Mitigation	Implementation	Priority
File Integrity Monitoring (FIM)	Track and alert on unauthorized changes to files, databases, and logs	**Critical**
Access Control	Enforce least privilege and strong authentication for sensitive data	**Critical**
Audit Logging	Maintain detailed logs of all data modifications for traceability	**High**
Regular Backups	Keep secure, immutable backups to restore altered or corrupted data	**High**

Endpoint Detection & Response (EDR)	Monitor applications and processes for suspicious data modification behavior	**Medium**

Technique 2: Email Bombing (T1667)

Description:

Adversaries may flood targeted email addresses with an overwhelming volume of messages. This may bury legitimate emails in a flood of spam and disrupt business operations. An adversary may accomplish email bombing by leveraging an automated bot to register a targeted address for e-mail lists that do not validate new signups, such as online newsletters. The result can be a wave of thousands of e-mails that effectively overloads the victim's inbox.

Created: 31 January 2025

Last modified: 15 April 2025

Version: 1.0

Platform: Linux, Windows, macOS

Tactics: Impact

Real World Example:

- **Fraud and OTP bypass attacks** where thousands of emails are sent to distract users while attackers reset passwords.
- **Ransomware campaigns** that email-bomb IT teams to delay detection of active system compromise.

Detection method:

- Monitor high-frequency messages from the same sender or IP range
- Use email gateway rate-limiting and spam filters
- Analyze mail server logs for abnormal delivery patterns
- Alert on mailbox storage exhaustion or service slowdown

Mitigation Strategies:

Mitigation	Implementation	Priority
Email Rate Limiting	Limit the number of email received per sender/IP	**Critical**
Spam & Email Filtering	Use advanced spam filters and content analysis	**Critical**
Mail Server Monitoring	Detect sudden spikes in email volume or delivery failures	**High**
IP & Sender Blocking	Block abusive IP addresses and domains	**High**
Mailbox Quotas & Alerts	Set storage limits and alerts for rapid mailbox growth	**Medium**
User Awareness	Train users to report unusual email floods quickly	**Medium**

Conclusion

The MITRE ATT&CK® Framework for macOS provides a comprehensive lens through which organizations can understand, anticipate, and defend against adversarial behavior. By mapping tactics such as Initial Access, Execution, Persistence, Privilege Escalation, Defense Evasion, Credential Access, Discovery, Lateral Movement, Collection, Command and Control, Exfiltration, and Impact, the framework highlights the full lifecycle of a cyberattack and the interconnected nature of malicious techniques.

Key takeaways include:

- **Holistic visibility**: ATT&CK emphasizes that adversaries exploit both technical vulnerabilities and human factors, requiring defenses that span technology, processes, and awareness.
- **Defense in depth**: Mitigation strategies—ranging from secure coding and patch management to multi-factor authentication and network segmentation—must be layered to reduce exposure.
- **Detection and monitoring**: Continuous monitoring, behavioral analytics, and proactive threat intelligence integration are essential to identify subtle attack patterns.
- **Practical utility**: For security teams, ATT&CK serves not only as a reference but also as a playbook, guiding incident response, red-team exercises, and long-term resilience planning.

Ultimately, the framework underscores that cybersecurity is not static; adversaries continuously evolve, and defenders must adapt with equal agility. By embedding ATT&CK into organizational practices, teams can transform threat knowledge into actionable defense, strengthening resilience against the ever-changing macOS threat landscape.

References

- https://attack.mitre.org/
- https://attack.mitre.org/matrices/enterprise/macos/
- https://www.crowdstrike.com/en-us/cybersecurity-101/cyberattacks/mitre-attack-framework/
- https://www.ruianding.com/blog/attck-for-enterprise-discovery/#:~:text=%E2%80%9CDiscovery%E2%80%9D%20in%20the%20context%20of,privilege%20escalation%2C%20or%20data%20exfiltration.
- https://www.zdnet.com/article/new-analysis-fruitfly-mac-malwware-almost-undetectable-backdoor/
- https://en.wikipedia.org/wiki/OceanLotus#:~:text=OceanLotus%2C%20also%20named%20APT32%2C%20BISMUTH,businesses%20with%20ties%20to%20Vietnam.
- https://www.infosectrain.com/news/in-a-watering-hole-attack-apples-macos-was-infected-with-new-dazzlespy-backdoor-exploits
- https://www.intego.com/mac-security-blog/dazzlespy-mac-malware-used-in-targeted-attacks/#:~:text=Posted%20on%20January%2025th%2C%202022,sponsored%2C%20cyber%2Despionage%20campaign.
- https://www.sentinelone.com/labs/20-common-tools-techniques-used-by-macos-threat-actors-malware/
- https://xsoar.pan.dev/docs/reference/playbooks/mitre-attck-co-a---t1083---file-and-directory-discovery