# **Table of Contents**

1. **Introduction**
   1. Purpose of the Guide
   2. Red Teaming in Large Enterprises
   3. Hybrid Cloud Considerations
   4. Scope: Azure, AAD, Windows, and Integrated Platforms
2. **Red Team Methodology**
   1. Reconnaissance
   2. Initial Access
   3. C2 and Persistence
   4. Privilege Escalation & Lateral Movement
   5. Actions on Objectives
3. **Goals and Attack Paths**
   1. Key Objectives of an Adversary
   2. Mapping Realistic Attack Paths
   3. Prioritizing Exploitable Weaknesses
4. **Crown Jewels – High-Value Targets**
   1. Active Directory Domain Controllers and Kerberos Keys
   2. Azure AD Tenant and Global Administrator Accounts
   3. Privileged Access Management (PAM) Vaults
   4. IT Service Management Systems (e.g., ServiceNow)
   5. Intellectual Property & Sensitive Data Repositories
   6. Certificate Authorities and Trust Infrastructure
   7. Global Infrastructure Control Points
5. **Attack Tactics and Techniques**
   1. Azure & AAD Identity Exploitation
      1. Token Theft (PRT, OAuth, SAML, Refresh Tokens)
      2. Conditional Access Evasion
      3. Cloud Persistence via Service Principals & App Secrets
      4. Azure AD Connect & Hybrid Identity Weaknesses
   2. LAPS Exploitation – Local Administrator Password Abuse
   3. IT Management Platform Attacks
      1. Microsoft Intune & ServiceNow Admin Compromise
      2. Lateral Movement via ITSM Workflows
   4. Windows Attacks in Hybrid Environments
      1. Golden SAML & Federation Trust Exploitation
      2. AD Certificate Services (AD CS) Misconfigurations
      3. Azure AD Kerberos Ticket Forgery
      4. Compromising Azure-Connected Machines
   5. Coordinated Hybrid Attacks – Simultaneous On-Prem & Cloud Takeover
6. **Real-World Attack Scenarios**
   1. Scenario 1: Cloud Account Takeover via OAuth Token Theft
   2. Scenario 2: Bypassing MFA and Establishing Cloud Persistence
7. **Defensive Strategies and Mitigations**
   1. Strengthening Identity Security in Azure & Windows
   2. Hardening Cloud Configurations & Privileged Roles
   3. Network Segmentation and Host Monitoring
   4. Detecting Illicit OAuth Consent & Token Abuse
   5. Secure Management of Secrets & Certificate Authorities
   6. Cloud and On-Prem Incident Detection & Response
   7. Continuous Security Testing and Red Team / Blue Team Collaboration
8. **Conclusion**
   1. Key Takeaways
   2. Red Team Insights for Hardening Enterprise Security
   3. The Role of Continuous Testing and Threat Intelligence
9. **References**

# Introduction

**Red Teaming is a simulated cyberattack conducted by authorized security professionals to test an organization’s defenses. Unlike traditional penetration testing that might focus on isolated technical vulnerabilities, Red Team operations take a holistic adversarial approach – emulating real threat actors to challenge the entire security posture. In large enterprises, this means not only probing on-premises networks and Windows domains but also cloud services, identity platforms, and management tools that modern businesses rely on. The goal is to identify gaps in people, processes, and technology by staging covert, full-scope attacks that mirror advanced persistent threats. This proactive simulation helps uncover critical weaknesses before malicious actors do, strengthening defenses in an era of increasingly sophisticated breaches.**

**Modern enterprise infrastructure is often a hybrid of on-prem and cloud systems, so Red Team assessments must adapt accordingly. Attackers today are as interested in cloud credentials and access tokens as they are in on-premises server passwords. A comprehensive Red Team engagement in an Azure-centric enterprise will typically span both Azure Active Directory (AAD) and on-prem Active Directory, leverage cloud services like Microsoft Intune or ServiceNow, and target high-value secrets stored in enterprise vaults. Red Teamers need to think like an adversary who can pivot between cloud and on-premises, exploiting trust relationships and misconfigurations along the way. This guide provides a deep dive into Red Team strategies for large enterprises, focusing on Azure infrastructure (AAD, Intune, and integrated SaaS applications like ServiceNow) and Windows environments tied to Azure. We outline the assessment methodology, key attack tactics, important objectives (ultimate “attack paths” or goals), and the crown-jewel targets adversaries seek. Real-world inspired scenarios (leveraging frameworks like Cobalt Strike for command-and-control) illustrate how these attacks unfold in practice, followed by a discussion of defensive mitigations. By understanding these tactics and techniques, security professionals can better prepare and fortify enterprise systems against determined attackers. A blend of industry knowledge, case studies, and trusted research is used to ensure accuracy and credibility in the guidance that follows.**

## Methodology

**A Red Team assessment in a large enterprise follows a structured methodology that mimics the full kill chain of a targeted attack. This typically involves several phases: reconnaissance, initial access, establishing persistence, privilege escalation, lateral movement, collection/exfiltration, and actions on objectives. In practice these phases aren’t strictly linear – skilled adversaries pivot fluidly as opportunities arise – but a methodical plan ensures coverage of all potential weaknesses. Below is an overview of the general approach a Red Team might use when assessing an enterprise heavy on Azure and related technologies:**

**Reconnaissance: Quietly gather open-source intelligence (OSINT) and internal insight about the target’s footprint. This includes mapping external-facing assets, enumerating domains and subdomains, identifying employee emails or credentials leaked online, and profiling technologies in use. For an Azure-focused enterprise, recon might reveal public-facing Azure services (web apps, storage accounts), metadata about the AAD tenant, or exposed endpoints for cloud services like an ITSM portal. Social engineering intelligence is also gathered – e.g. names of administrators or patterns in helpdesk processes – to craft credible phishing pretexts. Effective recon provides initial attack surface indicators and potential weak entry points without alerting the target.**

**Initial Access: Using the recon data, operators attempt to gain a foothold. Common tactics include spear-phishing emails (or smishing texts) that trick users into revealing credentials or executing malware, exploiting vulnerable internet-facing applications, or leveraging stolen credentials found in breach dumps. In cloud-heavy environments, stealing a single set of valid user credentials can be as impactful as malware, especially if multi-factor authentication (MFA) can be bypassed. For instance, obtaining an employee’s Office 365/Azure AD credentials via a convincing phishing page and getting around MFA can grant direct access to cloud resources. If malware deployment is used for initial access, the Red Team may implant a covert beacon (e.g. a Cobalt Strike agent) on a compromised machine, which calls back to a command-and-control (C2) server. This provides remote access to issue further commands stealthily on that host.**

**C2 and Persistence: Once inside, maintaining access is crucial. The team establishes persistence mechanisms to survive restarts or credential changes. If a Cobalt Strike Beacon was deployed, operators might install it as a service, create scheduled tasks, or abuse startup registry keys on Windows to ensure it re-launches. In Azure/AAD contexts, persistence can also mean backdooring cloud credentials – for example, adding a new Azure AD user or application client secret that the team controls. Throughout this phase, stealth is emphasized: using “living off the land” techniques (abusing legitimate admin tools and scripts) and carefully shaping network traffic to evade detection. C2 channels are encrypted (HTTPS, DNS tunnels, etc.) and operate with low-and-slow patterns to blend in. Simultaneously, the Red Team performs internal reconnaissance from their foothold – enumerating directory structures, group memberships, roles, and network topology. In an Azure-integrated environment, this might include querying Azure AD for user and role information, checking Intune-managed device lists, or identifying privileged accounts in systems like ServiceNow or secret vaults.**

**Privilege Escalation and Lateral Movement: After establishing an initial beachhead (often a low-level user or single machine), the team seeks higher privileges and broader access. In a Windows domain, this involves techniques like credential dumping (using tools such as Mimikatz to scrape passwords or hashes from memory), exploiting misconfigurations (e.g. weak ACLs on AD objects), and leveraging enterprise management tools against themselves. For example, if the compromised user’s workstation is Hybrid Azure AD-joined, the team might extract that user’s Primary Refresh Token (PRT) for Azure AD from memory. A PRT is an authentication token for cloud SSO; with it, an attacker can impersonate the user to access Azure cloud services without needing their password, effectively bridging an on-prem compromise to cloud access. On the cloud side, privilege escalation may involve exploiting misconfigured Azure AD roles (for instance, a user who is an owner of a highly privileged application or service principal) or abusing API permissions to elevate access. The Red Team may also target Azure AD Connect servers if present – this directory synchronization service often holds a credential with high privileges in Azure AD. By accessing Azure AD Connect’s database or memory, the team can extract the plaintext password of the sync account, potentially gaining a route to Azure tenant admin rights if that account has directory write permissions. On-premise, escalating to Domain Admin remains a classic objective; in Azure, the counterpart goal is Global Administrator of the tenant (or equivalent high-privilege cloud roles). Lateral movement entails hopping from one compromised system or account to others. This could be done by techniques like pass-the-hash or pass-the-ticket in a Windows domain (reusing captured credentials to move between machines), or by using a compromised cloud admin account (for example, an Intune administrator) to push malicious software to many endpoints at once. All of these moves are executed with caution to avoid detection, often using Cobalt Strike’s built-in capabilities or custom scripts delivered through the existing beacons.**

**Actions on Objectives: Finally, the Red Team works toward the engagement’s explicit goals – the “crown jewels” defined for the scenario. This could mean obtaining certain sensitive data (exfiltrating database records or confidential files), gaining full control of core infrastructure (like domain controllers or Azure subscriptions), or simulating destructive actions (e.g. deploying ransomware or wiping data) depending on scope. Upon reaching this phase, the team will document how they achieved each objective. They often demonstrate access rather than cause damage – for example, proving they can retrieve all secrets from a password vault or download emails from the CEO’s mailbox as evidence of compromise, rather than actually exfiltrating everything or disrupting services. In stealthy Red Team operations, it’s common to avoid any noticeable impact; the team might take screenshots, obtain password hashes, or perform read-only actions as proof of access. Throughout, operational security is maintained: covering tracks by clearing or modifying logs when possible, using compromised accounts at times and in ways that blend in with normal behavior, and adapting on the fly when defenses are encountered. (For instance, if an action triggers an alert on an endpoint detection system, the team may change tactics to evade further detection.) The exercise concludes with the Red Team achieving as many objectives as possible without detection, thereby revealing the paths real attackers could take. These findings then inform remediation efforts.**

## Attack Tactics

**The following sections highlight several key attack tactics and techniques that Red Teams (and real adversaries) use, particularly in Azure-cloud and hybrid enterprise environments. Each tactic corresponds to specific techniques in frameworks like MITRE ATT&CK, and together they cover the avenues an attacker might exploit from cloud identity systems to on-premises infrastructure.**

**Azure and AAD Attacks – Abusing Cloud Identities and Integrations**

**Token Stealing (Session Tokens & Refresh Tokens): Rather than cracking passwords or breaching servers, attackers often target the tokens that grant access to cloud resources. For example, on a compromised Windows machine, an adversary can use tools like Mimikatz to extract a user’s Azure AD Primary Refresh Token (PRT) from memory. With a valid PRT (and its session key), the attacker can request access tokens for various Azure services as that user, bypassing the normal login process. This effectively allows the attacker to impersonate the user in the cloud without needing their password or MFA approval. Similarly, if OAuth refresh tokens for SaaS applications are stored on disk or in browser memory, an attacker who compromises an endpoint may steal those tokens. With a stolen OAuth token, the adversary can continue to access cloud services even if the user changes their password, since the token remains valid until revoked or expired. Red Teams will attempt to steal such session tokens or cookies from memory, browser caches, or token stores on disk, then replay them to cloud services. This technique, sometimes called “pass-the-token,” enables attackers to immediately leverage an on-premises foothold to access cloud applications as the victim user.**

**Azure AD Connect & Hybrid Identity Exploitation: In enterprises with on-prem Active Directory linked to Azure AD (a hybrid identity setup), Azure AD Connect is a critical component. It synchronizes user accounts and sometimes passwords or password hashes between on-prem and cloud. Azure AD Connect often uses an account (e.g. an MSOL\* account) with directory synchronization privileges. Attackers target Azure AD Connect servers to extract the stored credentials for this sync account. For example, by accessing Azure AD Connect’s SQL database and using the tool’s binaries or known scripts, a Red Team can decrypt and retrieve the plaintext password of the Azure AD sync account.**

**In older or misconfigured setups, this account might even have Global Administrator rights in Azure AD or other high privileges. By compromising Azure AD Connect, the Red Team gains another path to cloud dominance from on-prem. Additionally, trust relationships like federation can be abused – if the organization uses ADFS (Active Directory Federation Services) for Azure authentication, and the team steals the ADFS signing certificate or token-signing key, they could perform a Golden SAML attack (forging SAML tokens to impersonate any user in Azure AD). Thus, weaknesses in the hybrid identity bridge (synchronization, federation, etc.) are prime targets. The general principle is that the intersection of on-prem AD and Azure AD is fertile ground for attacks: any feature that connects the two (password hash sync, pass-through authentication agents, ADFS trusts) can be leveraged to move from one realm to the other if not securely configured. For instance, an on-prem Domain Admin who compromises ADFS or certificate services could forge cloud authentication tokens, or a Global Admin who misconfigures password writeback could inadvertently give on-prem attackers a path to Azure.**

**Cloud Admin Misconfigurations: Red Teams also look for mistakes in Azure AD tenant configuration and cloud security settings. Common examples include unused or stale Global Administrator accounts (e.g. a “break-glass” emergency account with no MFA that could be quietly compromised), lack of MFA enforcement on admin roles, overly permissive application consent settings, or legacy authentication protocols left enabled. Attackers will perform password spray attacks against Azure AD accounts (attempting common passwords at scale) and try legacy endpoints (like IMAP/POP3 or SMTP Auth for Exchange Online) that may allow bypassing MFA if those protocols are not disabled.**

**Well-known attacks such as Golden SAML (forging authentication tokens by exploiting ADFS, as noted) have their cloud analogues – for instance, if an attacker somehow obtains the signing keys used by the Azure AD tenant (or an app’s JSON Web Token signing certificate), they could forge tokens for that environment. While Azure AD’s core keys are highly secure and not tenant-accessible, similar concepts apply in misconfigured federations or if third-party IdPs are integrated. Another focus is Conditional Access policies in Azure AD: if these are misconfigured, a Red Team might find that certain networks, device states, or user groups are excluded from MFA requirements, providing an easier route in. For example, an organization might exclude its on-premises network from MFA prompts – an attacker who gains a foothold on that network (or spoofs their IP address) could then authenticate as a privileged user without MFA. In summary, Azure AD and cloud attacks often revolve around abusing identity and access management gaps. Because Azure AD’s role-based access model and application consent framework can be complex, attackers frequently find subtle misconfigurations that yield privilege escalation or persistence (such as an application with excessive permissions, or an admin account with no MFA). The Red Team will methodically probe these cloud settings to escalate their access—from a low-privileged user to an Application Administrator to a Global Admin—and to maintain that access without detection.**

**In essence, attacking Azure AD means abusing the identity system itself. A clever adversary can turn features against the organization: e.g. using a legitimate admin’s token or session to do malicious acts, or using an allowed legacy protocol to bypass a modern control. Identity is the new perimeter, so many Azure-focused attacks exploit identity tokens, trust relationships, and configuration weaknesses rather than traditional OS vulnerabilities.**

**LAPS Exploitation (Local Administrator Password Solution) – Attacking Password Management on Endpoints**

**Microsoft’s Local Administrator Password Solution (LAPS) is a tool designed to improve security by regularly rotating the local Administrator password on every domain-joined machine and storing those passwords securely in Active Directory. This ensures each machine has a unique admin password, mitigating lateral movement with shared credentials. However, if not implemented with strict access controls, LAPS itself can become an avenue for attack. A Red Team will examine how LAPS is deployed and attempt to exploit any weaknesses. Key tactics include:**

**Abusing Privileged Access to LAPS Passwords: In a proper LAPS deployment, only specific security principals (e.g. a dedicated AD group for helpdesk or system administrators) have permission to read the confidential ms-MCS-AdmPwd attribute on computer objects (which stores the machine’s local admin password). The Red Team will attempt to compromise any user or group that has these rights. For example, if a helpdesk user account that can read LAPS passwords is phished, that account gives the team the ability to immediately retrieve the local administrator password for potentially hundreds of machines. There have been real cases where built-in groups like “Account Operators” or other unintended users had read access to LAPS passwords across many computers. An attacker who gains such an account can escalate from a single compromised user to local admin on many endpoints by pulling those passwords from AD. This turns what would have been a contained foothold into broad lateral movement, since with local admin rights the attacker can move between systems (RDP, run remote commands, etc.) and further dump credentials on each one.**

**LDAP Relay to Dump LAPS Passwords: A more advanced technique leverages network-level attacks to exploit LAPS. If the Red Team can coerce a privileged user (say, a helpdesk admin with LAPS read rights) to authenticate to a server they control (using tricks like LLMNR/NBNS poisoning or malicious UNC paths in a phishing email), they can capture an NTLM authentication hash. Using an NTLM relay tool, the team can relay that authentication to the domain controller’s LDAP service and perform allowed actions as that user. Researchers have demonstrated modules that, when relayed, issue an LDAP query to read LAPS passwords from AD on behalf of the user. In essence, even without knowing the cleartext of the privileged user’s password, the Red Team could use the one-time network authentication to retrieve any machine password that user is authorized to view. This is a stealthy way to extract a trove of local admin credentials across many systems in one go. After dumping those passwords, the operators gain administrative access on those machines, greatly facilitating lateral movement.**

**Targeting LAPS Client-Side: Another angle is to attack LAPS on the endpoint. LAPS periodically runs a process on each domain-joined computer to update its local Administrator password in AD. By default that process runs as SYSTEM. If an attacker already has code execution as SYSTEM on a machine (for example, by exploiting a vulnerability to become NT Authority\SYSTEM), they could attempt to intercept the LAPS process or its data. For instance, they might hook the function that generates the new password or intercept the communication that sends the updated password to AD, thereby capturing the credential before it’s securely stored in AD. This is a complex approach and less commonly needed—after all, if an attacker is already SYSTEM on one box, they have many options to move further. But it’s illustrative of the depth of exploitation: even a tool meant to improve security can be subverted under certain conditions.**

**In summary, LAPS is a great defensive tool, but its security hinges on proper access control and monitoring. A Red Team will search for any missteps (like too many users with read access or lack of monitoring on LAPS attribute queries) to turn LAPS into an attack vector. Once they have LAPS passwords, they effectively obtain keys to many kingdom doors at once, enabling stealthy expansion of their control over the network.**

## Intune and ServiceNow Admin Compromise – Leveraging IT Management Platforms

**Large enterprises often rely on cloud-based management platforms for both devices and IT services. Two common examples are Microsoft Intune (part of Endpoint Manager) for device/mobile management, and ServiceNow for IT service management (ITSM). Individuals with administrative access to these platforms hold significant power: an Intune admin can control thousands of computers and deploy software to them, while a ServiceNow admin can access sensitive incident data and integrate with other systems. A Red Team will seek to compromise such admin accounts because they offer potent opportunities for privilege escalation, lateral movement, and data access that might bypass traditional security controls.**

**Abusing Microsoft Intune (Endpoint Manager): Intune is Microsoft’s cloud-based solution for mobile device and PC management via Azure AD integration. An Intune Administrator (or any Global Admin, since Global Admins by default have Intune admin rights) can push configurations, software, and scripts to managed devices. This makes Intune a powerful attack vector once an attacker has admin access to it. A common tactic is to deploy a malicious PowerShell script or other payload to devices through Intune. Intune allows administrators to upload PowerShell scripts that execute as SYSTEM on enrolled Windows machines. In practice, the Red Team could upload a Cobalt Strike Beacon loader or other malware as an Intune management script and assign it to a set of devices (for example, all workstations in a certain group). Researchers have documented that when such a script policy is added, it will run automatically on target machines (typically during their next regular Intune check-in). This means an attacker can move from a cloud compromise to an on-premises foothold by using Intune to remotely execute code on internal devices. In one real scenario, an attacker with Azure Global Admin rights leveraged Intune to push a Cobalt Strike beacon to a hybrid-joined workstation; when the machine checked in and ran the script, the attacker got a system-level shell on that on-premises host– effectively bridging cloud to ground. Even without Global Admin, a user with the Intune Admin role alone could do the same. Beyond malware deployment, Intune can be abused for reconnaissance as well: the portal contains device inventory information and user assignments, which the Red Team can use to identify high-value targets (e.g. finding a device tagged as belonging to a Domain Admin). Intune might also store integration tokens or API keys (for example, if Intune is integrated with other services via Microsoft Graph API); a compromised Intune admin could steal these for additional persistence or access.**

**Persistence via Intune: In addition to immediate code execution, a compromised Intune environment can facilitate long-term persistence. An attacker could create a new Intune configuration policy or continuously push a malicious script, ensuring that even if one backdoor on a device is removed, it will be redeployed on the next device check-in. They could also add a new admin account to Intune (or abuse Intune’s role delegation) to ensure they maintain access even if the originally compromised account is detected. Essentially, Intune becomes the attacker’s own internal software deployment system. This is particularly stealthy because it leverages the organization’s trusted management channel – endpoints will view the malicious actions as coming from the legitimate Intune service. From a defender’s perspective, this looks like normal administrative activity. By exploiting Intune, the Red Team can rapidly scale control over many systems that were not directly accessible, turning a single cloud account compromise into domain-wide impact.**

**ServiceNow Admin Account Abuse: ServiceNow is a widely used ITSM platform for handling incidents, requests, and various IT workflows. An administrator on ServiceNow has access to very sensitive data: user information, support tickets (which often include passwords, configuration details, or even incident responses), configuration management databases, and sometimes integration credentials to other systems. If a Red Team compromises a ServiceNow admin account (via phishing or credential reuse), they can quietly harvest a wealth of information. First, they can exfiltrate data by reading tickets and records – for example, incident reports might contain recent security issues or even credentials pasted by IT staff. (Notably, in one real incident an attacker accessed an enterprise’s ServiceNow and obtained extensive employee data and internal communication.)**

**Such data can be used for further social engineering or to understand the network’s structure and weak points. Second, ServiceNow often integrates with other systems: it might have stored credentials for an AD integration account (used to pull user data or reset passwords), or API keys for monitoring tools. A ServiceNow admin could write a malicious script or workflow to extract these credentials. For instance, if there’s an integration that allows ServiceNow to create accounts in Active Directory, the Red Team might attempt to abuse that feature to create a new admin user. Additionally, ServiceNow’s email integration could be used to send phishing emails from a trusted internal address, or its orchestration capabilities could potentially execute code on connected systems (though this depends on what modules the organization has and how they’re configured). In short, a ServiceNow admin account can be a goldmine for both intelligence and pivoting opportunities. Real threat actors have shown interest in such platforms: simply logging in with stolen admin credentials and querying the database can yield passwords, architectural diagrams, and other valuable intel– all through a single cloud interface without touching an endpoint.**

**Privilege Escalation via ITSM Workflows: In some cases, IT service platforms like ServiceNow can even be leveraged to gain higher access in the environment. Consider that ServiceNow is often used by IT support to perform actions like password resets for users via automated workflows. If an attacker has admin control of ServiceNow, they could attempt to trigger a password reset for a privileged account (for example, impersonating a helpdesk process to reset a Domain Admin’s AD password to one of their choosing). This depends on how ServiceNow is integrated – perhaps it has an automation account in AD with rights to reset certain passwords or create accounts. If multi-factor authentication or additional approvals are not in place for such actions, the attacker could effectively turn the ITSM into an illicit admin tool. For example, the Red Team might identify a ServiceNow workflow that allows an admin to unlock user accounts or change passwords; they could misuse this to reset a service account that is a member of Domain Admins, thereby granting them high privileges on-prem. This is a less direct route and requires understanding the ITSM workflows, but it underscores that any system with administrative ties to others can be weaponized. The Red Team will evaluate how far they could go with ServiceNow admin rights alone.**

**In summary, administrative access to management platforms like Intune and ServiceNow can be as devastating as access to a domain controller, albeit in different ways. Intune gives direct control over computers (and by extension the ability to indirectly obtain domain credentials by deploying tools like credential dumpers), while ServiceNow grants expansive visibility into IT data and even indirect control over accounts through integrations. These are attractive targets during Red Team ops because they often have weaker monitoring – an attacker abusing legitimate functionality in these platforms may not trip traditional security alarms. A successful compromise of an IT management platform allows the attacker to live off the land using sanctioned tools, making detection much harder.**

**(Note: Attacks on enterprise secret management systems (password vaults, certificate management) are another critical area, but those will be addressed in the context of “crown jewels” later in this guide. In essence, if an attacker can compromise a privileged access vault or machine identity system, the impact is immense – as it often yields the keys to many other systems.)**

## Windows Attacks in Azure-Integrated Environments – Bridging On-Prem and Cloud

**Even when focusing on Azure, a Red Team must still consider attacks against on-premises Windows infrastructure, especially where it intersects with cloud identity. Many advanced techniques involve leveraging on-prem compromises to access cloud resources or vice versa. A few notable tactics include:**

**Forging Tickets and Tokens (Golden SAML): In hybrid setups using Active Directory Federation Services (ADFS) or other federated identity, on-prem compromise can directly translate to cloud access. If the Red Team steals the token-signing certificate from an ADFS server (for example, by obtaining the server’s system privileges and exporting the cert), they can forge authentication tokens for Azure AD (this is known as Golden SAML, analogous to the Golden Ticket attack in AD Kerberos). This allows them to impersonate any federated user, including Global Administrators, in the Azure tenant. Similarly, older “silver ticket” techniques (forging Kerberos service tickets) could be used against ADFS services if the ADFS service account’s credentials are compromised. The key point is that weaknesses in federated identity can completely undermine cloud authentication.**

**Compromising AD CS (Certificate Services) to Forge Trust: Many organizations deploy Active Directory Certificate Services (AD CS) on-prem for enterprise PKI. If the Red Team compromises the AD CS infrastructure, they can issue themselves certificates that might be trusted for authentication. For instance, Azure AD can be configured to trust certificate-based auth for hybrid join or VPN – if the attacker can produce a valid cert for a user or device, they may bypass other controls. Recent research has shown AD CS is often misconfigured, allowing low-privilege domain users to request certificates that map to high-privilege accounts (known as ESC8/ESC9 privilege escalation via AD CS). By exploiting this, an attacker can escalate to Domain Admin on-prem, then use that to further attack Azure. For example, once Domain Admin, the attacker could register a new device in Azure AD (a rogue Azure AD join) to create a foothold in the cloud tenant. If Azure AD Connect or ADFS use certificate-based authentication, a compromised on-prem CA could issue a certificate that Azure trusts for a given user, effectively letting the attacker log in as that user without needing their password. Red Teams will include AD CS in their target list when Azure AD is in scope, because abusing trust in certificates can seamlessly bridge on-prem and cloud identity systems.**

**Abusing Azure AD Kerberos (Ticket Constraints): Microsoft has introduced an Azure AD Kerberos feature that allows cloud-issued Kerberos tickets for Azure AD-joined devices to access on-prem file shares without a direct line of sight to a domain controller. This involves a trust where a Kerberos Ticket-Granting-Ticket (TGT) for the device is stored in Azure AD and certain keys are shared between on-prem AD and Azure AD. While this is cutting-edge and not widely used yet, a Red Team aware of it would note that if they could compromise the keys or trust in this setup, they might forge Kerberos tickets that Azure AD will accept or vice versa. This is an emerging attack surface that blurs cloud and on-prem further. (At present, such attacks are largely theoretical or beyond the scope of most engagements, but they highlight how deep integration creates new risks.)**

**Lateral Movement via Azure-Connected Machines: On-prem servers and workstations that have connections to Azure can themselves become stepping stones. For example, suppose the enterprise uses Azure Arc to manage on-prem servers, or has the Azure Virtual Machine Agent on Azure VMs that connect to Azure management planes. These agents often run with high privileges and use local certificates or tokens to authenticate to Azure. If an attacker compromises a server that has an Azure Arc agent, they could extract the agent’s service principal credentials or certificate from that machine. Those credentials might allow them to impersonate that server in Azure or access the Azure management backend for that server. Likewise, credentials used by automation tools (Azure Automation runbook credentials, Logic App connectors) might be stored on-prem or in cloud configurations accessible from on-prem. A Red Team that finds such credentials on a compromised system can use them to pivot into the cloud environment. This effectively uses the on-prem machine’s identity to perform actions in Azure that the machine is authorized for. It’s another way an on-prem compromise can lead to cloud compromise beyond just user credentials.**

**Coordinated Destructive Actions: Once holding both on-prem and cloud control, a Red Team may demonstrate the “worst-case” potential of an attack by orchestrating simultaneous impacts. For instance, they might illustrate that they could deploy ransomware on on-prem servers (Domain Controllers, file shares) while at the same time using their Azure Admin access to delete cloud resources or turn off VMs and cloud backups. In a real attack, this combination could cripple a hybrid environment: encryption of on-prem data coupled with removal of cloud-based backups or services. Red Teams typically won’t actually cause damage, but showing the ability to do so drives home the point that full cross-environment compromise can be catastrophic. It emphasizes the need for unified security monitoring and incident response across both domains.**

**Overall, attacking a hybrid Azure/on-prem environment means the Red Team leverages any foothold to traverse the trust boundary between cloud and ground. By compromising shared components (like sync services, federation, or management agents), they turn an on-prem breach into cloud takeover (and vice versa). This “worst of both worlds” scenario is exactly what defenders must prepare for in modern enterprises.**

## Goals of Attacks and Key Objectives

**Every Red Team operation ultimately aims to achieve specific high-impact objectives that mirror real adversary missions. Depending on the threat actor’s intent – whether espionage-driven, sabotage-driven, or financially motivated – the end goals could be stealing sensitive data, disrupting critical services, or establishing persistent control for future exploitation. For example, an espionage-motivated test might prioritize obtaining confidential R&D data or emails from executives, whereas a ransomware simulation would focus on demonstrating the ability to corrupt or encrypt core systems. Achieving these goals in a controlled exercise demonstrates how an organization’s defenses could fail against a real attack and provides valuable lessons for remediation. By documenting how each “mission success” was accomplished (and how long it went undetected), the Red Team highlights where improvements are needed so the real adversaries can be stopped before reaching the same objectives.**

## Crown Jewels: Highest-Value Targets

**“Crown jewels” are the assets an attacker would covet most – the systems, accounts, or data whose compromise would be most devastating to the organization. In a large enterprise with Azure and hybrid infrastructure, these jewels span both cloud and on-prem realms. Identifying and securing them is paramount, and a Red Team exercise often uses them as the finish line for success. Based on the attacks we’ve outlined, the crown jewels typically include:**

**Active Directory Domain Controllers and Kerberos Keys: The on-prem AD Domain Controllers (DCs) are perennial crown jewels. They contain authentication data (password hashes, Kerberos keys) for the entire organization. With control of a DC, an attacker can impersonate any user (by issuing themselves Kerberos tickets or extracting password hashes), decrypt network authentication traffic, and essentially “own” the enterprise. Domain Admin groups and critical accounts like the AD KRBTGT account (which signs Kerberos tickets) are part of this category. If a Red Team fully compromises a domain controller, they’ve hit a crown jewel – remediation often requires a complete Active Directory restoration or reset of trust.**

**Azure AD Tenant and Global Administrator Accounts: In the cloud arena, the Azure AD tenant (identity directory) itself is a crown jewel. A Global Administrator account – or similarly highly privileged Azure roles like Privileged Role Administrator or Subscription Owner – is the cloud equivalent of Domain Admin. Compromising the Azure AD tenant means the attacker can access virtually any resource in that cloud environment: they could read or send emails from any user’s mailbox, download all files from SharePoint/OneDrive, create or delete users, and modify cloud configurations at will. For instance, with Global Admin access, an attacker could impersonate the CEO’s account via email to conduct fraud, or spin up dozens of expensive cloud resources for cryptomining as a distraction. Within Azure, other jewels include things like Azure Key Vaults that store application secrets and encryption keys – if those are compromised, an attacker gains keys to decrypt databases or access APIs (unless those keys are protected by separate HSMs). Service principals and managed identities with high privileges (e.g. an application identity with Owner rights over a subscription) are also crown jewels, as compromising them can equate to full cloud control.**

**Privileged Access Management Vaults: Enterprise password vaults or privileged access management (PAM) systems (e.g. CyberArk PAM, Dell/Quest TPAM, HashiCorp Vault) concentrate the most sensitive credentials in the organization. The vault server and its datastore are crown jewels because they literally hold the “keys to the kingdom.” For example, CyberArk’s Digital Vault often stores Domain Admin passwords, service account passwords, SSH keys, and more. If an attacker compromises the vault (or even a single administrator account on the vault), they can potentially obtain a trove of credentials that unlock dozens of other systems. One could liken it to a treasure chest: the container (vault server) and the contents (credentials) are both extremely valuable. In a Red Team context, demonstrating the ability to get into the PAM vault is usually game over, as it implies immediate or eventual compromise of many other crown jewels (domain controllers, databases, etc.). Thus, PAM systems themselves (and their admin credentials) are uber-targets.**

**IT Service Management Systems (ServiceNow, etc.): Platforms like ServiceNow might not traditionally be seen as “crown jewels,” but if they house sensitive records or capabilities, they are critical. For instance, ServiceNow often contains incident tickets, which could include detailed internal discussions of security issues or even passwords shared in troubleshooting. It may also hold customer PII in support requests or financial data in IT change requests. If ServiceNow has integration credentials (say, an account to reset AD passwords or query HR databases), those become stepping stones to other high-value systems. So, while ServiceNow itself may not be as fundamentally critical as a domain controller, the data and integrations it holds can be crown jewels. A breach of ServiceNow could expose a wealth of information (as seen in the Microsoft ServiceNow incident of 2024) and even provide indirect access to other systems. Similarly, other IT management systems (Jira, Confluence, etc., if they contain secrets or sensitive project data) could be considered crown jewels in certain contexts.**

**Intellectual Property & Sensitive Data Repositories: Many enterprises have core data stores that represent the heart of their business – source code repositories (GitHub, Azure DevOps, etc.), product designs, formulae, proprietary research data, customer databases, etc. These might be hosted on-prem (e.g. a file server or on-prem Git server) or in cloud services. If the crown jewel is the data itself, the Red Team’s ultimate goal might be to exfiltrate this data. Achieving Domain Admin or Global Admin is often an intermediate goal en route to the true objective: for example, obtaining Domain Admin might allow access to the SharePoint server that houses all the company’s R&D documents. In short, any system that houses vital intellectual property or regulated data can be a crown jewel. For a pharmaceutical company, it might be the research database; for a software company, the source code repository. The Red Team will treat those with the highest priority once high-level access is gained.**

**Certificate Authorities and Trust Infrastructure: The security of the entire enterprise often hinges on a few root trusts – for example, the root certificate authorities that issue certificates, or services like Venafi that manage machine identities. If an attacker compromises the enterprise CA (or a machine identity management system like Venafi that interfaces with the CA), they can forge identities at will. For instance, they could issue a code-signing certificate in the name of the company and use it to sign malware (bypassing antivirus), or issue a client auth certificate that the VPN will accept as valid. Venafi itself, if used, becomes a crown jewel because it often has connectivity to HSMs and CAs and might allow issuing certificates for any internal or external service. The keys that secure SAML tokens or OAuth JWTs are also crown jewels (recall the Storm-0558 incident where a stolen signing key allowed forged Azure AD tokens) – though in Azure AD’s case, those keys are managed by Microsoft and not accessible to the tenant. In a hybrid environment, the keys on ADFS or other identity providers under the enterprise’s control are in scope and are high-value.**

**Global Infrastructure Controls: This category includes things that control large swaths of IT infrastructure. For example, the vCenter server that manages all VMware virtual machines, or the SCADA system that controls industrial processes, or the root account for an AWS/Azure subscription hosting production services. In the Azure context, a Subscription Owner or User Access Administrator role that grants control over critical Azure resources is a crown jewel (often this overlaps with Global Admin in Azure AD, but in some orgs they split duties). If an attacker gets the Azure Subscription Owner credentials for the production environment, they can delete resources, exfiltrate databases, or open backdoors for future use. In multi-cloud environments, the equivalent in each cloud (like an AWS root IAM user) is similarly critical. Essentially, any meta-control system – be it cloud orchestration, virtualization hosts, container orchestration admins, etc. – that can be used to manipulate a large number of systems or services is considered a crown jewel. These are often overlooked in traditional IT security (focus tends to be on AD and databases), but Red Teams will mark them as high-value targets. For example, compromising the Jenkins server that deploys code to production could be as deadly as compromising the code repository itself.**

**These crown jewels are what sophisticated adversaries will ultimately target, because they either grant expansive access (domain controllers, global admin) or directly contain the organization’s most prized assets (sensitive data, crypto keys). A successful Red Team exercise often means demonstrating control or access to one or more of these crown jewels, thereby showing the business the realistic impact of a breach.**

## Example Scenarios (Real-World Case Studies)

**To illustrate how these tactics come together, let’s walk through two example attack scenarios that reflect modern, Azure-focused adversary techniques. Each scenario is derived from real-world attack patterns observed in recent years, combining multiple techniques to achieve the Red Team’s objectives.**

### Scenario 1: Cloud Account Takeover via OAuth Token Theft

**Initial Access: A targeted finance employee receives a phishing email that appears to come from Microsoft, inviting them to review a document. Upon clicking the link, instead of asking for credentials directly, it prompts the user to grant access to an application (OAuth consent). The app is malicious – created by the Red Team – but it is presented as if a known business application is requesting permissions. Believing it to be legitimate, the user consents, granting the fake app permissions such as “Read and send mail” on their Office 365 account. The Red Team’s rogue application receives an authorization code and exchanges it for tokens, including an access token and refresh token for the user’s Office 365 data. This means the Red Team now has an API token that can act on behalf of the user without needing the user’s password or interactive login. No malware is involved, and the user’s account wasn’t “hacked” in the traditional sense – yet the Red Team has obtained a foothold in the cloud as that user.**

**Abusing OAuth Access: Using the token, the Red Team’s application connects to Microsoft Graph API and accesses the user’s mailbox. They quietly read through emails and Teams messages the finance employee has. They discover an ongoing email thread with the company CFO regarding quarterly financial results, which includes a spreadsheet of revenue projections. This data is highly sensitive. The Red Team exfiltrates the spreadsheet from the CFO’s emails via the compromised employee’s access (since the employee had a copy forwarded to them). They also find, in the user’s Teams chats, that an IT support technician had sent the user a temporary password last week for a shared drive – a credential the Red Team gladly notes for later use. At this point, the Red Team has achieved a primary objective (access to confidential financial data) just by abusing an OAuth token. Importantly, this access persists even if the user changes their password, because the refresh token can be used to obtain new access tokens until it is manually revoked by an admin. In fact, the attackers maintain access to the mailbox for several days, silently harvesting any new emails. (Many real threat actors use exactly this technique – illicit OAuth apps – to maintain long-term access to email accounts even after the initial point of compromise, since it often bypasses MFA and doesn’t trigger obvious login alerts.)**

**Pivoting Internally (Cloud-to-Cloud): With the finance user’s account under control, the Red Team moves to broaden their access. They leverage the trust inherent in internal communications: from the finance user’s email, they send the CFO (and a few other executives) a new phishing email, this time with a link to what looks like a secure file share. When the CFO clicks it, they are also prompted to grant access to an “Office 365 – Finance Documents” app. Given that the email came from a colleague and references real financial data (the attackers craft the email using tidbits from what they read in the CFO’s mailbox), the CFO proceeds and consents. Now the Red Team obtains the CFO’s token as well, with even higher access—the CFO’s account has access to financial systems and other confidential projects. Through the CFO’s account, the team finds emails about a potential merger and valuable intellectual property. They exfiltrate several files marked “Confidential – Merger Plans.” At this stage, the Red Team has compromised two O365 accounts and accessed critical data, all through OAuth abuse and internal phish, without ever needing to touch the endpoint or break a password.**

**Outcome: This scenario demonstrates how a cloud-focused attack can unfold completely within Azure/O365: from an OAuth consent phishing to persistent mailbox access and internal spear-phishing to expand access. The Red Team achieved significant objectives (financial data theft) while flying under the radar – there were no malware binaries for antivirus to detect, and the logins all appeared as API access by “trusted” applications. The token-based access bypassed MFA in place. In a real incident, threat actors have used exactly these methods to maintain Business Email Compromise (BEC) campaigns, automating phishing and even injecting themselves into email threads for fraud. One notable nation-state incident in 2023 involved attackers forging Azure AD tokens to access emails of dozens of organizations. Our Red Team scenario, while using a consent phish instead of token forgery, underscores the same point: cloud accounts can be hijacked and abused in ways that don’t involve malware or on-prem network traffic at all. By the end of this scenario, the Red Team had covert access to sensitive emails and files and could have proceeded to conduct financial fraud (e.g. by sending false payment instructions as the CFO). The organization’s cloud-centric defenses were put to the test against purely identity- and token-based threats.**

### Scenario 2: Bypassing MFA and Establishing Cloud Persistence

**Initial Access: The Red Team targets an IT administrator who has high privileges in Azure AD. This admin’s account is protected by MFA and Conditional Access policies (for example, only compliant domain-joined devices from certain IP ranges can log in). Instead of using technical exploits, the Red Team employs a social engineering attack known as MFA fatigue: they repeatedly attempt to log in with the admin’s stolen password (obtained via an earlier phishing email), causing the admin’s authenticator app to prompt over and over late at night. After dozens of persistent prompts, the groggy admin accidentally approves one. Immediately, the Red Team’s login succeeds on a Virtual Machine they had set up in an Azure region that wasn’t geoblocked. Now they have an interactive session as a Global Administrator on the Azure AD tenant. They are conscious that Conditional Access policies might still restrict some actions (for instance, the policy might enforce MFA again for certain sensitive operations or block unfamiliar locations), but with one successful MFA approval, they have a foothold. They quickly register a new device in Azure AD and mark it as compliant (using admin privileges to do so), essentially simulating that their VM is now a trusted device for that admin account. This allows them to satisfy Conditional Access requirements going forward – a technique akin to the one real attackers use by registering their own MFA method or device for a compromised account.**

**Cloud Privilege Escalation & Persistence: Now operating as a Global Admin, the Red Team has the keys to the kingdom in Azure. Their first move is to create persistence mechanisms that will survive if the compromised admin account is discovered. They create a new user account, “azure-admin2”, and assign it the Global Administrator role. To avoid immediate suspicion, they set this account’s MFA status to enabled but register their own device as its MFA (essentially pre-enrolling an attacker-controlled authenticator app). They also add this account to a special group that was excluded from Conditional Access policies (many organizations have a “breakglass” emergency account or group that bypasses CA – in this case, the Red Team simply uses their privileges to put their new user into that group). This means azure-admin2 can log in from anywhere without MFA challenges, providing a backdoor that the Red Team controls entirely. Next, they move to create an application in Azure AD – they register a new app, “Monitoring Tool,” and give it high privileges by granting admin consent to required permissions (for example, Directory.ReadWrite.All in Graph API). They add a client secret for this app and record it. This gives them a non-user identity to use; even if all users have MFA, the app uses certificate/secret auth which doesn’t require MFA. With this application’s credentials, the Red Team can script interactions with Azure AD and other services (reading/writing data) at any time, without needing a user login. They effectively have a service principal backdoor in the tenant.**

**Now that multiple persistence options are in place, the Red Team explores what they can do with full Azure AD access. They enumerate all user accounts and notice an on-prem AD synchronization account (from Azure AD Connect) that has the “password hash synchronization” permission – meaning if they compromise on-prem AD, they could potentially use that to regain cloud access. They also verify they have User Access Administrator rights (Global Admin can grant themselves this), which allows them to assign Azure RBAC roles at the subscription level. Using this, they elevate their new backdoor account to an Owner role on the primary Azure subscription. Now, even outside of Azure AD, they control the actual cloud resources. They proceed to demonstrate access to a few assets: for instance, they use the admin account to read contents of an Azure Key Vault (which stored some application connection strings and secrets) and to dump several executives’ Exchange Online mailboxes via eDiscovery (Compliance Center) – all completely legitimate actions from an admin perspective.**

**Abusing Conditional Access and Evasion: During this scenario, the Red Team made a point to evade and manipulate Conditional Access, which is a key defense. By adding their accounts to the exclusion group for CA, and by registering their own MFA devices, they ensure no further MFA prompts will block them. They even temporarily modify a Conditional Access policy to allow legacy authentication protocols, then perform a login using an older IMAP method to one of their backdoor accounts, and revert the change. This tests whether legacy protocol use is being monitored. It was not. Essentially, they have shown that once an attacker attains a Global Admin role, they can disable or bypass the very cloud security policies intended to stop them. The Red Team also carefully covers their tracks: they delete some Azure AD audit logs (or if unable, they create an “alert fatigue” situation by generating numerous benign audit events to bury their malicious ones). They turn off any user risk policies in Azure AD Identity Protection to prevent Microsoft’s systems from flagging the unusual login patterns.**

**Outcome: In this second scenario, the Red Team demonstrated how an attacker can move from an initial credential compromise to full cloud tenancy takeover, despite MFA and Conditional Access protections. They employed modern cloud-focused techniques: MFA fatigue to get in, Conditional Access evasion by manipulating policy and device registration, and cloud persistence by creating new accounts and app credentials. At the end of the engagement, the Red Team had multiple independent backdoors into the Azure environment – even if one was found, others could still provide access. They also had the theoretical ability to inflict massive damage (they could delete all cloud VMs, or download all sensitive data, or create new accounts at will). Perhaps most importantly, they operated largely within the bounds of normal admin behavior, making detection very difficult. This scenario mirrors real-world breaches where attackers, upon gaining admin access, have been seen to add their own credentials or apps to maintain persistence. It underscores the importance of securing admin accounts against social engineering, monitoring policy changes and new credential additions, and employing defense-in-depth even when attackers gain high-level access.**

## Defensive Strategies and Mitigations

**After examining the many ways an enterprise can be compromised, it’s crucial to consider how to defend against these scenarios. A layered defense strategy is required – there is no single silver bullet. Below is a brief outline of defensive measures corresponding to the tactics and weaknesses discussed, which an organization can implement to improve resilience:**

**Strengthen Identity Security: Enforce phishing-resistant MFA for all users, especially privileged roles. This means using methods that are harder to fatigue or phish, such as FIDO2 security keys or number-matching push notifications, rather than simple approve/deny pushes. Educate users about MFA fatigue attacks so they don’t approve unexpected prompts. Implement Azure AD Conditional Access policies to block logins from unusual locations or legacy authentication protocols, cutting off common credential abuse pathways. For on-prem AD, disable legacy protocols like NTLM where possible and require signing/sealing for LDAP to prevent relay attacks (mitigating things like the LAPS relay scenario). Regularly review membership of privileged groups (Domain Admins, Global Admins, etc.) to minimize who has standing high access, and remove any unnecessary accounts that can read sensitive data like LAPS passwords. Use Azure AD Privileged Identity Management (PIM) to make admin access just-in-time and temporary, so even if an account is compromised, it may not have active privileges at that moment.**

**Hardening Cloud Configuration: Apply the principle of least privilege in Azure AD and Azure resources. Ensure no low-level user is an owner of a highly privileged service principal or app (audit app permissions and consented apps). Monitor for creation of new application credentials or any OAuth consent to high-privilige apps – these can signal abuse of the kind observed in real attacks. Limit and log who can grant OAuth consent for the tenant (consider disabling user consent for most apps and requiring admin approval). Use Azure AD access reviews to catch accounts in excluded groups (e.g. users excluded from MFA policies) and ensure those exceptions are still needed. Limit the number of Global Administrators – use roles like Global Reader for day-to-day and only elevate to GA when necessary via PIM. In Intune, require a second layer of approval or notifications for any new software deployment or script push. Enable alerts for when an Intune policy that runs code is created or modified. Similarly, integrate ServiceNow and other SaaS into Azure AD SSO and enforce MFA for their admin logins. Monitor ServiceNow admin activities: if an admin account suddenly queries thousands of records or extracts credentials, that should generate an alert.**

**Network Segmentation and Host Monitoring: On the on-prem side, continue to defend traditional perimeters. Segregate critical servers (Domain Controllers, tier-0 systems, vault servers, CA servers) on network segments with strict access controls. For example, ensure that the password vault is only reachable from a small set of jump-hosts or admin subnets, not from every client machine. Implement LAPS correctly but also monitor its usage – enable auditing so that any read of a ms-MCS-AdmPwd attribute is logged, and alert if an unusual account is reading multiple passwords. Deploy endpoint detection and response (EDR) tools on servers and high-value workstations to catch behavior like credential dumping (Mimikatz’s telltale activity in LSASS) or suspicious token misuse. Modern EDR solutions can often detect patterns of memory injection or anomalous use of Windows API calls that tools like Cobalt Strike exhibit. Ensure Windows Defender Credential Guard or similar is enabled on administrative workstations to make stealing tokens and hashes more difficult.**

**Monitoring for Illicit Consent and Token Abuse: Because many cloud attacks bypass traditional network monitoring, invest in identity-based auditing. Enable Azure AD sign-in logs and log all OAuth consent grants and application registrations. Feed these logs to a SIEM and create alerts for anomalies – e.g. an OAuth consent to an unknown app that requests high permissions, especially if granted by a non-admin user (could indicate consent phishing). Also monitor for an influx of failed MFA attempts (which could indicate MFA fatigue attacks in progress), or an account that successfully registers a new MFA device shortly after a series of failures. Utilize Azure AD Identity Protection to detect impossible travel (a user logging in from New York and then 10 minutes later from Russia) and token anomalies. Consider employing continuous access evaluation so that if an account is disabled or marked high risk, its existing tokens are revoked in near-real-time.**

**Protect Secret Management Systems: Given their criticality, PAM vaults and similar systems demand extra hardening. Keep vault software fully patched so known exploits (like the CyberArk web UI RCE from 2018) won’t work. Use strong master passwords/encryption keys and consider hardware security modules (HSMs) or secure enclaves to protect vault encryption keys. Strictly limit which accounts can retrieve secrets – implement role-based access control in the vault and enforce MFA on vault logins. Disable or remove any default admin accounts provided by the vault software and use a breakglass procedure for emergency access that isn’t just another static password. Enable the vault’s built-in anomaly detection features if available – for example, CyberArk has analytics that can flag unusual credential retrieval patterns. For HashiCorp Vault, require MFA for any admin operations and use namespaces/policies to compartmentalize access so one token can’t read everything. Store Vault’s unseal keys in a secure manner (split among trusted parties or use auto-unseal with an HSM-backed key in Azure Key Vault), to prevent an attacker from simply stealing unseal keys and opening the vault. Monitor all vault activity: alert on mass secret accesses or on use of root/admin tokens. For certificate management (like Venafi), protect the connected CAs with HSMs for their private keys and require multi-party approval for any highly sensitive certificate issuance (e.g. code signing or VPN auth certs). Enforce MFA for any administrators of the machine identity management system. Essentially, treat your secret management and trust infrastructure with the same level of security as domain controllers – or higher, since they often hold the crown jewels indirectly.**

**Detect and Respond to C2 Activity: Since tools like Cobalt Strike are prevalent in Red Team and adversary toolkits, organizations should invest in detecting their telltale behaviors. Monitor egress network traffic for patterns consistent with beaconing – for instance, a host making regular, small HTTPS POST requests to an unknown domain every few seconds or minutes. Even if beacons use TLS and mimic normal traffic, an unusual frequency or timing can be a giveaway. Use network detection and response (NDR) systems or well-tuned firewall analytics to flag these anomalies. On endpoints, watch for suspicious parent-child process relationships (e.g. Word spawning PowerShell, which is often seen in phishing payloads), or the use of “living off the land” binaries (LOLbins) in odd ways (e.g. rundll32.exe running code from a temp directory). EDR solutions can also detect memory injection attempts – Cobalt Strike beacons often inject into legitimate processes like explorer.exe or svchost.exe; an EDR can catch unusual memory protections or thread injections in those processes. Deploy honeypot credentials and resources as well: for instance, create a fake admin account that should never be used in normal operations, and have alerts if it ever logs in (indicating an attacker tried to use a captured hash). Similarly, plant a dummy secret in the vault or a fake AWS key in a file share; if someone tries to use it, you immediately know the vault or share was compromised. These canaries can drastically reduce detection time for certain breaches.**

**Incident Response Preparedness: Despite best efforts, assume a breach will occur and prepare for it. Have an up-to-date incident response (IR) plan that specifically covers cloud incidents (e.g. procedures for handling OAuth app abuse or Azure AD tenant compromises). Conduct regular Red Team vs Blue Team exercises (purple teams) to practice detection and response in scenarios like those described. Ensure logging is robust and retained – many Azure audit logs are not kept forever by default, so export them to a SIEM or storage for forensic analysis. Train the response team on cloud-specific remediation: e.g. how to evict an OAuth app or how to purge malicious service principal credentials, which are not steps traditional IT playbooks cover. Lastly, implement a post-Red Team remediation process: every finding from a Red Team should result in concrete security improvements (patching a system, changing a configuration, providing additional training, etc.). The goal is continuous improvement so that each assessment raises the bar for the attackers.**

## Conclusion

**In conclusion, the assessments and techniques discussed above underscore a key principle: security is only as strong as the weakest link in a deeply interconnected chain. Azure AD and cloud services offer incredible power and agility, but missteps in their configuration or in their integration with on-prem systems can open serious holes. Likewise, secret management tools greatly reduce day-to-day risk, yet if compromised they become a jackpot for attackers – a single vault breach can hand over a dozen other crown jewels. This PhD-level analysis of Red Team operations affirms that a holistic approach is needed. One must consider the attacker’s perspective across the entire enterprise terrain – from cloud tokens to domain controllers, from human factors to hardware. By rigorously testing and reinforcing each layer of defense, enterprises can raise the bar so high that even advanced adversaries will be detected and foiled before they reach the crown jewels. Ultimately, the value of Red Team exercises lies in revealing these insights proactively, so that real incidents can be prevented. Every weakness uncovered is an opportunity to fortify the enterprise and protect the integrity, confidentiality, and availability of its critical systems.**

**References:**

**IBM – Anderson, E. What is red teaming? (IBM, 2024).**

**Jumpsec Labs – Red Teaming the Cloud: A Shift in Perspective (2022).**

**SpecterOps – Hausknecht, R. Attacking Azure & Azure AD, Part II (2020).**

**SpecterOps – Robbins, A. Death from Above: Lateral Movement from Azure to On-Prem AD (2020).**

**Praetorian – Crosser, A. Obtaining LAPS Passwords Through LDAP Relaying Attacks (2020).**

**Savvy Security – Caraballo, J. Microsoft ServiceNow Incident: Lesson in Credential Hygiene (2024).**

**Tenable – Van de Wiele, T. Dangerous Permissions (Intune) in Entra ID (2023).**

**Microsoft – Threat actors misuse OAuth applications to automate financially driven attacks (Microsoft Security, 2023).**

**Microsoft – Analysis of Storm-0558 techniques for unauthorized email access (Microsoft Security, 2023).**

**MITRE – T1556.006: Modify Authentication Process – Multi-Factor Authentication (MITRE ATT&CK®, 2024).**

**CISA – CSA: Red Team Shares Key Findings (Alert AA23-059A, 2023).**