**Introduction**

Modern financial institutions rely on tightly integrated identity systems – both cloud-based (e.g., Okta for single sign-on) and on-premises (Active Directory for domain identity). Advanced Persistent Threat (APT) groups and sophisticated adversaries increasingly target these identity platforms as “keys to the kingdom.” This paper provides an offensive red-team perspective on exploiting identity systems in a financial sector context. We delve into step-by-step attack simulations against Okta and Active Directory (AD) environments, referencing real-world breaches and APT campaigns. We expand on multi-factor authentication (MFA) bypass techniques (from push fatigue to device enrollment exploits), Okta API abuse, federation hijacking, and Active Directory Federation Services (AD FS) attacks like Golden SAML. The goal is to map how adversaries can chain these techniques to compromise “crown jewel” financial systems – such as SWIFT payment networks, core banking platforms, and sensitive finance databases – often by first undermining identity controls. All discussion is focused on offensive tactics only (no defenses), providing red teams with a thorough playbook for simulating advanced attacks in the financial sector.

**Attack Simulation – Step by Step**

To illustrate the attack paths, we walk through a representative simulation of a determined adversary targeting a bank’s hybrid identity environment. This scenario involves a cloud IdP (Okta) federated with the bank’s on-prem AD. The adversary’s objectives are to gain administrative control of the identity systems and use that access to reach high-value financial applications.

1. **Initial Access via Spear Phishing and AiTM Proxy** – The attack often begins with phishing a privileged user (e.g., an IT admin or finance executive). Rather than a simple credential phish, advanced threat actors use **Adversary-in-the-Middle (AiTM) phishing kits** like *Evilginx2* or *Modlishka*. These kits proxy the real Okta (or Office 365) login page, intercepting both the credentials and the MFA token or session cookie . The phishing site passes the user’s credentials to the real service, triggers the MFA prompt, and captures the session token after the victim completes MFA. This yields an authenticated session cookie or token that the attacker can import into their browser, effectively bypassing MFA completely . Once the cookie is imported (using a tool like EditThisCookie), the attacker is logged in as the user without needing their password or MFA again. The *diagram below* illustrates this workflow, where the attacker’s phishing server sits between the victim and the legitimate service to hijack the authentication tokens:

*Workflow of an MFA bypass via Evilginx2 adversary-in-the-middle phishing. The attacker’s phishing server intercepts the MFA token and session cookie, allowing them to log in as the victim without further MFA prompts.*

1. **Bypassing MFA through Push Fatigue and Social Engineering** – Not all intrusions require stealing tokens; some attackers exploit human behavior. **MFA push bombing (fatigue attacks)** bombard a user with repeated push notifications until the user accidentally or out of fatigue approves one. The September 2022 *Uber* breach is a prime example: the LAPSUS$ group spammed an Uber contractor’s MFA app with endless prompts, eventually inducing an approval that let the attacker in. Many users, especially when harassed by late-night alerts, will eventually click “Approve,” thinking it’s a glitch. Sophisticated attackers combine this with phone-based social engineering – calling the target purporting to be IT support and urging them to accept the MFA prompt. Once the user accepts, the attacker’s device is authenticated. In one real case, adversaries even convinced helpdesk staff to **reset MFA factors** for an admin user – essentially removing the MFA barrier entirely. Okta observed such tactics in 2023 where threat actors called a service desk to reset all MFA factors on a Super Administrator’s account, then logged in using stolen credentials. This underscores that MFA, while a vital defense, can be circumvented by exploiting the human element or weak recovery workflows.
2. **Device Enrollment Manipulation** – Another attack path exploits how identity platforms handle new device registrations for MFA. Many MFA systems (Okta, Duo, etc.) allow a user who knows the primary credentials to enroll a new second-factor device . If an attacker has obtained a user’s username and password (e.g., via phishing or credential stuffing), they can often register their own phone or token as an MFA device on the account. This is particularly prevalent when organizations allow self-service MFA enrollment – for instance, if no factor was previously set, or when enrolling a first device only requires primary credentials. The **MITRE ATT&CK** framework documents this as *T1098.005: Device Registration manipulation*, noting that an adversary with compromised credentials may enroll a new device to bypass MFA and maintain persistent access. A notable real-world example occurred in the 2020 SolarWinds hack: FireEye discovered the attackers registered a new phone in their MFA system, appearing as if an employee added a device, which actually allowed the attacker to satisfy 2FA requirements. Kevin Mandia of FireEye noted this silent new device enrollment was a major red flag indicating advanced tradecraft – the attacker **bypassed two-factor by simply adding their own device** after obtaining the user’s primary credentials. From a red team perspective, if you gain a user’s AD credentials and that user uses Okta with MFA, attempt to register a new authenticator (like a new Okta Verify instance or a hardware token) for that account. Often, organizations have a gap where the first MFA device or a new device can be added with only a password, giving full access until the intrusion is detected.
3. **Exploiting Okta Session Tokens and API** – Once an attacker has a foothold (e.g., a valid user session via stolen cookie or an MFA-approved login), the next step is to escalate privileges inside the identity platform. Okta is a rich target; as an identity provider, it controls access to myriad applications. Attackers will often search for ways to become an **Okta administrator**. In an October 2023 incident, a threat actor who hijacked a support engineer’s Okta session was able to **create a new “all-powerful” admin user** within the customer’s Okta tenant. According to a KrebsOnSecurity investigation, an Okta customer (BeyondTrust) shared a troubleshooting HAR file with Okta support – unfortunately, that HAR contained an active session token. The attacker had already compromised Okta’s support system and stole the HAR, then replayed the session token to impersonate the support engineer. Within 30 minutes, the threat actor used the session to create a new global admin account in the customer’s Okta org. This illustrates how **session hijacking** and API misuse can lead to full Okta tenant takeover. In practice, once a red team has an authenticated Okta session (especially with admin or helpdesk privileges), key API calls include: creating new users, assigning admin roles, resetting MFA or passwords for admin accounts, and adding API tokens. Okta’s own API could be leveraged with an API token to script persistence (e.g., create a backdoor user). In the 2023 support system breach, the attackers likely leveraged Okta’s identity APIs while impersonating the support agent’s session to perform high-impact actions. Red teams can mimic this by using Okta’s API with a stolen session cookie or token to enumerate users (/api/v1/users), elevate privileges, or insert new factors on accounts. **Session token replay** essentially allows the attacker to skip authentication entirely and perform authorized actions as the victim user. Attackers have also stolen **API tokens or OAuth tokens** from source code or endpoints (e.g., Okta GitHub breach via stolen token), which can similarly grant admin API access.
4. **Privileged Access via Helpdesk & Federation Abuse** – A dangerous escalation observed in real breaches is **helpdesk impersonation combined with identity federation abuse**. A cluster of attacks in 2023 (attributed to threat actor *Scattered Spider*) targeted several US companies (including MGM Resorts) by social engineering IT service desks to reset MFA for an admin and then exploiting Okta’s **Org2Org federations** . In these cases, once the attacker gained Super Administrator access in the Okta tenant (through stolen creds and helpdesk-induced MFA resets), they used Okta’s legitimate **Inbound Federation** feature to create a trust relationship from an attacker-controlled IdP. In simpler terms, they added a new SAML identity provider in the victim’s Okta config – which only a Super Admin can do – and abused it to impersonate any user in the organization. The attacker’s IdP would assert the identity of targeted users (say a finance manager or database admin), and because the federation trust was configured, Okta would accept those identities as authenticated via SSO. This trick effectively bypassed the need to phish each user account; with one Super Admin compromise, the attacker impersonated many accounts by forging SAML responses from their own IdP. Okta’s security advisory described how the adversary **manipulated the SAML username parameter to match real users in the victim org**, thereby gaining access to any application as those users. A red team exercise could simulate this by, for example, setting up a malicious SAML IdP (if allowed in a test scenario) or simply describing the process in a white-team tabletop: once you have Okta admin, add a new inbound SAML IdP, then use SAML tokens to assume identities of high-value users.
5. **Active Directory Compromise and Lateral Movement** – In parallel to cloud IdP attacks, most financial orgs have Active Directory as the backbone of on-prem identity. After initial intrusion (perhaps via an Okta route or directly via phishing malware dropper on a workstation), the attacker will target AD to escalate privileges on-prem. A classic route is **dumping credentials and impersonating accounts in AD**. Tools like *Mimikatz* are used once the attacker has a foothold on a Windows system with sufficient privileges; Mimikatz can extract password hashes or Kerberos tickets from memory. If the attacker reaches a Domain Controller (DC) or steals a Domain Admin’s credentials, they can perform the **“Golden Ticket”** attack – forging Kerberos tickets that grant domain-wide access. A golden ticket is analogous to the Golden SAML (discussed later) but for Kerberos: by extracting the KRBTGT account hash from AD, the attacker can generate valid Kerberos TGTs for any user, gaining unlimited access as any user in AD. In our simulation, assume the attacker phished a user who is a local admin, ran an exploit to get a reverse shell, then used privilege escalation to become domain admin in AD (common techniques include exploiting unpatched vulnerabilities or abusing weak service accounts). Once Domain Admin is obtained, they will likely **dump the AD FS token-signing certificate** if the organization uses AD FS for federating to Okta or other cloud apps – this certificate enables the Golden SAML attack (discussed shortly). Additionally, with Domain Admin, the attacker can create new AD accounts or backdoor existing ones (for persistence), and also directly VPN into the network or connect to any system. Gaining AD access also often grants access to enterprise apps that rely on AD authentication, such as core banking databases or file shares with financial data.
6. **Combined Cloud-OnPrem Attack Paths** – Many real incidents see attackers pivot between cloud and on-premises. For example, the SolarWinds attackers (UNC2452, associated with APT29) first compromised on-prem AD (via the Sunburst backdoor on SolarWinds Orion), then **stole the AD FS token-signing cert to forge SAML tokens** for cloud services . They effectively bridged the gap – using on-prem domain access to produce cloud authentication tokens. Conversely, an attacker who starts in the cloud (via Okta) might leverage features like LDAP interface or AD agent to move on-prem. In our scenario, suppose the attacker who set up inbound federation in Okta can now single-sign-on into an internal HR or finance app that is integrated with Okta. If that app is internally hosted, the SSO token might give them a foothold on a server inside the firewall. They might then exploit the server (if vulnerable) to get OS-level access, and from there target AD domain controllers. Thus, an Okta tenant takeover can lead to **Active Directory domain compromise** if the environments are connected (which is common via federation or cloud sync tools). APT campaigns increasingly exploit this interplay: for instance, one campaign in 2022 saw attackers trick helpdesk to reset Okta MFA and then use access to compromise AD domains via the VPN and legacy integration. Red teams should plan multi-stage operations that reflect this: cloud to on-prem pivot (steal SAML token -> use for VPN or internal app) or on-prem to cloud pivot (AD credentials -> abuse federation to Okta). Each step offers chances for *post-exploitation* and persistence as discussed next.

**APT Case Studies and Attribution**

Understanding how real threat actors operate against financial institutions informs the techniques above. We highlight notable APT groups and incidents that align with our scenario – focusing on identity compromise, Okta/SSO breaches, and Active Directory attacks in the finance sector.

* **LAPSUS$ Okta Breach (Jan 2022)** – The LAPSUS$ group, known for brazen hacks, hit Okta via a third-party support contractor (Sitel). In that incident, LAPSUS$ obtained remote access to a support engineer’s machine and ultimately to **Okta administrative consoles**. Okta acknowledged that ~366 customers were potentially affected by that breach. LAPSUS$ techniques often relied on **social engineering and MFA bypass**. For example, they targeted an external contractor for Uber and bombarded them with MFA push requests (the “MFA fatigue” method), successfully gaining entry to Uber’s VPN . In the Okta case, it’s reported they may have similarly leveraged remote desktop access and token theft. Once in Okta’s support tools, LAPSUS$ had the ability to impersonate support personnel and potentially reset customer tenants’ security settings. This incident underlines how a *targeted attack on identity infrastructure* can have downstream impact on many organizations. Red teams referencing LAPSUS$ will note their pattern of **targeting identity providers and bypassing MFA through human factors** – a modus operandi repeated in multiple companies (Microsoft, Cisco, Uber, Okta). The group was unusually public, even advertising recruitment for insiders. For attribution, LAPSUS$ is more of a criminal gang than a nation-state APT, but their success against Okta admin access shows even non-state actors can execute identity takeover if defenses are weak.
* **Scattered Spider / Muddled Libra (2023)** – Also known as 0ktapus or UNC3944 by some vendors, *Scattered Spider* is a threat group that in 2023 focused on helpdesk social engineering to compromise identity systems (heavily targeting telecom and hospitality sectors, including MGM and Caesars Entertainment). Okta’s August 2023 security advisory and subsequent reports tie this group to campaigns where **multiple Okta customer tenants were hijacked** by obtaining Super Admin roles . The attackers, likely based in the US/UK and in their teens, used voice phishing (vishing) to convince IT staff to reset MFA on admin accounts, then logged in and abused federation (as detailed earlier) . One publicized case is the MGM Resorts breach: the attackers gained domain admin on MGM’s network after first compromising Okta and presumably using that to access VPN or internal systems. Attribution-wise, Scattered Spider isn’t a nation-state but an organized group using APT-like techniques (highly targeted, some custom tooling). They demonstrated how **Okta tenant takeover** can lead to full enterprise breach. Red teams can learn from their playbook: start with stolen credentials (maybe via SMS phishing for an Okta login), exploit MFA weaknesses (push spam or support call), become an Okta admin, then pivot everywhere. Scattered Spider also reportedly enrolled their own devices for MFA once they got initial access, to secure persistence on the accounts. This matches MITRE technique T1098.005 (device registration) we described, reinforcing that real attackers do add their phones or tokens to defeated accounts for continued access.
* **APT29 (Cozy Bear) and the SolarWinds Campaign (2020)** – An example of a nation-state APT attacking identity infrastructure was the SolarWinds supply chain compromise attributed to Russia’s SVR (APT29). After initial infiltration via trojanized software, the attackers’ *Stage 3* goal was to **steal the ADFS token-signing certificate and forge tokens to access cloud resources**. This was publicly reported by FireEye/Mandiant during their incident response: the attackers moved from on-prem to cloud by using a **Golden SAML** attack . By extracting the certificate from AD FS (which federated AD to Azure AD/O365), they could generate SAML authentication assertions at will, impersonating any user to cloud apps. Notably, this bypassed MFA and even password changes, since the SAML token was accepted as valid authentication. The campaign (UNC2452, dubbed Nobelium by Microsoft) also exemplified **stealth and persistence**: even after discovery, if the token-signing cert wasn’t rotated, the attackers could still come back using forged tokens. Attribution was confirmed to a Russian state actor (likely APT29) given the targeting of US government and tech companies. For financial orgs, this shows the risk isn’t just theft of money – nation-states may target banks for intelligence or as a pivot into other targets. SolarWinds also illustrates the **fusion of techniques**: they planted malware for initial access, did extensive AD recon and privilege escalation (dumping credentials, moving laterally), and then used federation abuse (Golden SAML) to reach cloud email accounts of interest. Red teams emulating APT29 might similarly chain on-prem compromise with cloud token forgery.
* **Lazarus Group / APT38 (Bank Heists)** – North Korea’s Lazarus Group has a subset called APT38 focused on financial theft. This group was behind the 2016 Bangladesh Bank cyber heist where $81M was stolen via fraudulent SWIFT transfers. While that attack primarily involved malware (custom tools to manipulate SWIFT software and evade detection), Lazarus also compromised Active Directory domains in banks as part of establishing control. APT38 is known to lurk for months, escalating privileges, often deploying **keyloggers and extracting admin credentials**, then moving to mission-critical systems like the SWIFT terminal. In some APT38 cases, they deployed destructive malware after theft to cover tracks. Attribution is clear – it’s a state-backed team focused on theft to fund the regime. The relevance to identity attacks: even though their main goal was money, they needed to compromise Windows domains and often **create hidden accounts or schedule tasks** to maintain persistence. We can infer that any large financial hack (Bangladesh, Chile, etc.) likely involved undermining or bypassing authentication systems. For instance, to install their fraudulent payment orders, Lazarus must have taken over an administrator’s account or a service account in the bank’s network. One report from DOJ indicates they stole passwords and installed backdoors across numerous banks. Red teams can reference Lazarus tactics like spear phishing employees with malware, using tools like Mimikatz for credentials, and *living off the land* (using legitimate admin tools) once inside. While Okta wasn’t in wide use during the 2016 era, Lazarus today would certainly target any SSO platform a bank uses as it yields broad access.
* **FIN7/Carbanak** – FIN7 is a financially motivated group (often considered an APT due to skill) that heavily targeted banks, payment processors, and even retail point-of-sale systems. Their *Carbanak* malware campaigns (mid-2010s) were notorious for banking intrusions. FIN7 typically started with phishing (malicious attachments) to get an initial foothold, then moved to **harvest domain credentials and escalate privileges**. In an infamous Carbanak case, once they owned the domain, they even pushed malware to ATM network controllers to make ATMs spit cash at specific times – a highly coordinated attack. Carbanak also accessed SWIFT systems to send transfers. Attribution is tricky (some overlap with Russian-speaking hacker underground) but FIN7 has had members arrested. From an identity standpoint, FIN7’s success often relied on **dumping Active Directory accounts** and maintaining persistence by creating new local admins on critical servers. They might not have dealt with Okta, but modern FIN7 analogs would absolutely go after any centralized auth service or password vault. One could map Carbanak’s approach to MITRE tactics: initial email, run malware, memory-scrape creds, move laterally with legit credentials, and then abuse legitimate tools (like remote desktop or banking software itself). Red teamers emulating FIN7 should focus on post-exploitation of core financial apps after getting domain admin, rather than fancy SAML attacks. It’s a blend of cybercrime and APT methods.

**Advanced Exploitation Techniques**

We now dive deeper into the technical bag of tricks attackers use once they have a foothold, specifically focusing on advanced exploits in federated identity environments (Okta & AD). These techniques enable **stealth, persistence, and wide access** beyond the initial beachhead.

* **Golden SAML (Forging Federated Tokens)** – Golden SAML is an attack that allows an intruder to authenticate as *any user* to *any service* that trusts a SAML Identity Provider, without needing the user’s credentials or interacting with the IdP at all. This is achieved by stealing the SAML token-signing certificate from the IdP (for instance, from an AD FS server or in some cases from Okta if one could compromise its key – though Okta as a cloud service doesn’t expose it, so this is usually an AD FS scenario). With the private signing key, an attacker can generate a **forged SAML Response (“authentication object”)** that the Service Provider (e.g., AWS, Office 365, Salesforce, or a core banking app integrated via SAML) will accept as valid. Crucially, they can populate this SAML token with any attributes – meaning they can impersonate a highly privileged account or even a nonexistent account with elevated claims. Golden SAML was first publicized by CyberArk in 2017 as a variation of the Golden Ticket (Kerberos) attack. APT actors used it in the SolarWinds campaign – once they had the signing cert, they issued tokens for Office 365 access to email accounts of interest. From a defensive view, Golden SAML is hard to detect because the login is “normal” except that it didn’t actually go through the IdP at that moment. From an offensive perspective, if a red team manages to compromise an AD FS server or backup where the certificate is stored, running **mimikatz’s lsadump::ExportCert** or similar to extract the token signing cert enables this technique. They could then use tools like *Shimit* or CyberArk’s POC to create SAML tokens. The advantages are enormous: the tokens bypass MFA (since the SP thinks IdP did it) and even password changes (since SAML is out-of-band). Essentially, until the cert is changed, the attacker can come and go at will by minting fresh SAML tokens. This is **stealthy persistence** of the highest order in a federated environment.
* **Okta API Exploitation & Session Hijacking** – We touched on how stolen session tokens can be used to create Okta admins. Let’s expand on what an attacker can do once they have admin API access in Okta. Okta’s APIs allow programmatic control of the tenant: listing or searching users, updating users, managing groups, configuring authentication policies, etc. For example, an attacker can use the **Okta Users API** to create a new user account (e.g., username “svc\_admin99”) and assign administrative roles to it. They might name it something innocuous to blend in (like “Service Account – Backup”). Because this is done via API or admin console, it will have a record but can be overlooked if not properly monitored. Another vector: Okta’s **System Log** might be reviewed by defenders, so attackers attempt to minimize noisy actions. Instead of directly creating a new global admin (which raises eyebrows), they might **promote an existing low-privilege account**. In one case, an attacker with stolen session tokens tried to elevate a legitimate engineer’s account to super admin. The BeyondTrust incident described by Krebs shows the attacker using the impersonated account to *attempt to create* an admin – implying Okta’s APIs or console were used in that short window. Another advanced trick: use Okta’s **inline hooks or automation** – an attacker could insert malicious scripts or connections. This is less documented in the wild, but conceptually, a compromised admin could add a suspicious *Identity Provider* (as per inbound federation earlier) or even modify the *authentication policies* (e.g., turn off MFA for certain IPs to ease their future logins). Additionally, consider **session hijacking beyond the initial cookie**. If the attacker can compromise a user with high privileges, they can hijack any active session for that user across devices (Okta sessions are often valid for a duration). Attackers might try to steal browser cookies via malware on the endpoint as well (for instance, some infostealer malware are now grabbing Okta-specific tokens if present). A very recent trend in attacks (not limited to Okta) is stealing **session tokens from memory or logs** to bypass MFA entirely. For example, some ransomware groups search victim systems for authentication tokens to cloud services (like AWS keys, Okta tokens, etc.). In summary, advanced Okta exploitation means using its own features and data against it: harvest all user data (to plan further social engineering against high-value targets), weaken the security settings (disable notifications or MFA on certain accounts), and establish backdoors (new admins, or a persistent API token with wide scope). Okta’s flexibility is a double-edged sword – an admin account (legitimate or hijacked) is a goldmine for an attacker.
* **Active Directory Persistence (Golden Tickets, DCSync, and Beyond)** – Once Domain Admin is achieved, attackers employ multiple techniques to ensure long-term presence in AD. One such method is **DCSync**, where using tools like *Mimikatz* or *Impacket’s secretsdump*, the attacker pretends to be a Domain Controller and requests password hash data via the Directory Replication Service (DRS) protocol. With DCSync, they can dump the hashes of high privilege accounts including KRBTGT (needed for Golden Ticket) and others without even touching the DC’s LSASS process directly. After obtaining, say, the KRBTGT hash, a **Golden Ticket** can be crafted at any time, allowing the attacker to generate Kerberos TGTs for arbitrary users with arbitrary group memberships (this is analogous to Golden SAML’s arbitrary cloud access, but for the AD domain). Another persistence trick is creating **Silver Tickets** – forging service tickets (TGS) for specific services (like SQL service or others) by stealing that service’s account hash. Silver Tickets don’t give total domain access but can be more stealthy and target a particular server or application. Attackers also often create **backdoor accounts or groups**: e.g., using net user /add to make a new domain user, then net group "Domain Admins" newuser /add to sneak them into admin roles. They may disable or divert audit logs (using *AuditPol* or GPO changes). More advanced, some attackers deploy an **AD DS filter driver** or malicious SSP/DLL on domain controllers to capture credentials (this is extremely advanced and rare for red teams but seen in some nation-state cases). *Skeleton key* malware is another: it patches the DC’s authentication process in memory to accept a master password for any account (so the attackers can login as any user with a universal password while legitimate authentication still works normally). Additionally, compromising **AD Certificate Services (AD CS)** can enable forging of authentication certificates (so-called *ESC8* and *ESC11* abuses, as documented by SpecterOps) which let attackers authenticate as any user via PKI. In essence, the AD environment, once owned, offers countless ways to hide and persist. APTs will mix multiple: e.g., Golden Ticket for repeated access plus a couple of hidden user accounts as contingency, plus maybe a scheduled task that re-adds those accounts if removed. Red teams, depending on rules of engagement, might demonstrate persistence by Golden Ticket generation (in a lab) or showing how an attacker could maintain admin access even after incident response actions like password resets. **Token forgery and credential replication attacks (like Golden Ticket and Golden SAML)** are particularly devastating because they undermine the trust model of the authentication system itself.
* **Federation and OAuth Abuse** – Beyond SAML, attackers target other federation technologies. OAuth token theft is one: steal a refresh token for an OAuth application and you can access that app’s API until the token is revoked. We’ve seen incidents where attackers snatch tokens for cloud services (for example, stealing Slack OAuth tokens to access chats, or stealing Azure AD refresh tokens). In one campaign, attackers stole OAuth tokens from Okta’s own GitHub repositories to download source code. There’s also **OIDC id\_token manipulation** – if there are flaws in how an app validates tokens, an attacker might be able to modify the JSON web token (JWT). Another angle is **WS-Trust/WS-Fed** if used in legacy federations; some older protocols have downgrade vulnerabilities that clever attackers exploit to bypass MFA (for instance, manipulating a WS-Trust flow to get a token without MFA by presenting it as if from a “legitimate” source). Attackers also abuse **Conditional Access weaknesses**: if a cloud environment has trusted IPs where MFA is disabled, the attacker will try to come from those IPs to silently bypass MFA. For example, if the bank’s Okta or Azure AD trusts the office IP range to not require MFA, the attacker could compromise an on-prem server at that office (via VPN or malware) and then authenticate from there to bypass MFA. This isn’t a technical exploit per se but abusing misconfigurations. A diligent red team will examine the target’s identity federation configuration for these soft spots: look for allowed networks, known device requirements (maybe the policy says skip MFA for domain-joined devices – then the attacker makes a device look domain-joined), and old protocols allowed (like legacy Exchange auth which might accept just a password + legacy token). In summary, advanced exploitation is often about **abusing design and trust**: forging what is trusted (SAML/kerberos tokens, OAuth tokens) and finding holes in the trust assumptions (like an IdP trusting another IdP, or an app trusting a certificate). These are sophisticated but represent the cutting edge of identity attacks in APT scenarios.

**“Crown Jewels” – High-Value Financial Targets via Identity**

Adversaries ultimately care about the prize: the critical assets of a financial institution. Here we analyze what those crown jewels are, how they tie into Okta/AD, and how attackers pivot to them:

* **SWIFT Payment Systems** – The SWIFT network is a top target for financially motivated APTs (e.g., APT38) because it enables large wire transfers between banks. In modern banks, the SWIFT interface (Alliance Access or similar) often runs on internal servers, sometimes with heavy integration into the bank’s Active Directory for user auth or relies on local credentials. For instance, the Bangladesh Bank’s SWIFT software credentials were compromised by Lazarus. If the SWIFT system is integrated with AD (say operators log in with their domain accounts), compromising AD Domain Admin effectively gives the attacker control to create or use accounts that can initiate transfers. Even if SWIFT terminals use separate credentials, an attacker with domain foothold can install keylogger or remote access tools on the SWIFT operator’s workstation to capture those credentials. In some cases, banks might integrate SWIFT access with an SSO solution like Okta – perhaps using RDP or Citrix published via Okta for remote SWIFT access. If Okta is in that chain, an Okta admin could potentially **impersonate a SWIFT user’s session** or create new ones. An anecdotal scenario: a red team with Okta Super Admin could create a temporary user, assign them to the SWIFT application in Okta with an appropriate role, and then use that to log into the SWIFT interface (if SSO is SAML/OIDC, they might forge the SAML attribute for SWIFT role). The true “crown jewel” attack was seen in the Bangladesh Bank heist – attackers sent fraudulent SWIFT messages that moved $81M to offshore accounts. From an identity perspective, achieving that required either stealing the credentials of the SWIFT application or manipulating the authentication process. Thus, a likely path: compromise AD domain, move to SWIFT server, get SWIFT DB access or credentials, then send messages. In red team terms, demonstrating access to SWIFT is often a goal: one could simulate it by showing the ability to execute transactions on a test system once domain control is obtained.
* **Core Banking Systems** – Banks run core accounting and transaction processing systems (like Finacle, Temenos T24, FIS, Jack Henry, etc.). These might be mainframe-based or modern client-server apps. Integration with AD varies – some legacy mainframes might not integrate with AD, but many modern cores have web interfaces or thick clients that use AD for single sign-on or user management. If Okta is used, it could front-end web portals for core banking (for example, an internal web app that staff use to lookup accounts or approve loans). So an Okta compromise could directly allow access to these core apps if the attacker can SSO as an authorized user. More directly, AD compromise is huge for core systems: often core banking user accounts (for bank staff) are tied to their domain accounts via LDAP sync. By creating domain accounts or adding themselves to groups, attackers can try to provision access. For example, if a core banking app says “AD group Bank\_Tellers have access to retail accounts,” an attacker who adds an account to Bank\_Tellers group (via domain admin) can now log into the core system as a teller. Adversaries also target core databases (containing balances, transaction records, customer PII) – if those databases are MSSQL and the server is domain-joined, domain admin could directly connect to the DB or even **drop a payload** to manipulate data. FIN7’s bank intrusions reportedly sought ATM control and potentially core banking access to quietly set up fraudulent transactions. A red team exercise might consider demonstrating how, with domain admin, they could access a core banking database or application, perhaps pulling dummy customer data or initiating a sample transaction (obviously without actual financial impact in a test). This underscores that identity takeover is not the end – it’s the means to **fully compromise transactional integrity** in financial orgs.
* **Trading and Payment Processing Platforms** – Investment banks and trading firms have high-value systems like trading terminals, market data systems, and algorithmic trading platforms. Similarly, retail banks have payment processing gateways (credit/debit card processors, online banking portals). These systems increasingly integrate with central identity stores for convenience and audit. For example, a stock trading desk might use an app that authenticates via Okta to allow traders to execute trades. An attacker who can impersonate a trader via Okta could make unauthorized trades (which could be monetized or used to manipulate market positions). Another example: an online banking web application might use OAuth tokens issued by an identity provider – if an attacker steals or forges a token for an online banking user with high privileges (like an application support admin), they might be able to trigger fraudulent money movements or extract all user data. The crown jewels here are the ability to perform unauthorized financial transactions or to steal sensitive financial information (like credit card numbers, personal data, etc.). In 2022, an attacker group compromised a cryptocurrency exchange’s Okta tenant and from there was able to access wallets (this is an illustrative scenario – Okta is used by some fintech for customer login, and a breach can be catastrophic). Persistent identity compromise also allows long-term access to these systems: an attacker might not cash out immediately but quietly monitor communications (if they access an email archive via O365 after Okta compromise, for instance, they could gather intel on where big money moves happen, then act). For each high-value system, understanding **integration with identity** is key: does it use AD groups for authorization? Does it rely on Okta SAML for login? Does it use a static local account (which might be easier or harder for attacker depending on if they can crack those credentials)? Red teams often enumerate critical servers and see logged-in sessions or saved creds that can lead to those systems. If an Okta-managed SSO is the gateway to any of them, owning Okta is like getting the master key.
* **Long-Term Persistence in Identity Systems** – Truly owning a financial org means *staying undetected*. Attackers aiming for long-term access will establish persistence in both Okta and AD to survive remediation. In Okta, one could **create a new administrative user or service account** that blends in. Another method: generate an **API token** as an admin and store it – even if the interactive admin user is logged out or has its password changed, the API token might remain valid until revoked, allowing re-entry. Attackers might also adjust **factor settings** – e.g., register a secondary email or phone on a global admin account, so that if MFA or password resets are attempted by defenders, the attackers still have a way in via that secondary. The inbound federation trick is also a persistence method – it’s not easily noticed that a new IdP exists, and even if the initial admin account used by the attacker is locked out, that inbound trust might still allow access as other users (because the trust is stored in configuration). In AD, golden tickets serve persistence until Kerberos keys are rotated (which might never happen if IR isn’t thorough). Attackers can also leave **multiple backdoors**: perhaps add a scheduled task on a DC that creates a local admin on some server every night, or leave a web shell on an Exchange server (if Exchange is on-prem) that they can use to regain a foothold. Some APTs will plant completely separate access like a hidden TeamViewer or RDP backdoor on a system that is less monitored. But since our focus is identity, consider an elegant AD backdoor: create a fake *Domain Controller* by adding data in AD (there have been cases where attackers register a new DC in AD so that they can perform DCSync at will because AD thinks it’s a legit DC). Also, **pass-the-hash and silver tickets** can be reused whenever needed – if an attacker left with a bunch of NTLM hashes (say for a service account that rarely changes password), they can later use those to regain privileges. For Okta, persistence might also mean accessing the connected apps: for example, if the Okta tenant is locked down after a breach, but the attacker had created an O365 global admin via Okta (since Okta can provision users to O365), they may still have a route in via O365 directly. Every integrated system is a potential persistence point – the attacker will look for any identity or account that wasn’t evicted. In one case, after an Okta compromise, an attacker created a new *AWS API key* in a linked AWS account (because Okta was SSOing into AWS), so even after Okta was secured, they had cloud access through that key. Red teams should document multiple persistence options to demonstrate resiliency of their hypothetical attack: e.g., “Even if <X> account is discovered, we have Y method as backup.” In summary, the crown jewels (financial systems) are the end targets, but to continually access them, attackers make sure to *own the identity infrastructure long-term*. An entrenched attacker in a bank might operate undetected for months, periodically siphoning funds or data, until a major anomaly gives them away.

**Conclusion**

The offensive scenarios detailed in this paper highlight how critical identity systems are in the security of financial institutions. Okta and Active Directory – the cloud and on-prem identity pillars – present rich targets for attackers and, when compromised, give virtually unchecked access to a victim’s most sensitive assets. We explored a comprehensive attack chain: from the initial phish to bypassing MFA, exploiting device enrollment, leveraging helpdesk trickery, hijacking Okta sessions, forging SAML tokens, and ultimately reaching into the heart of banking systems like SWIFT or core ledgers. Each step mirrors techniques observed in real APT campaigns (LAPSUS$, Scattered Spider, APT29, APT38, FIN7, and others) – underlining that these aren’t theoretical in a vacuum but actual methods adversaries used to steal millions or spy on critical data.

For professional red teams, these tactics offer a blueprint to simulate the worst-case scenarios for a financial sector client. By emulating these methods, red teams can help organizations identify gaps in their identity protections and understand the potential business impact of an identity breach. **Offense informs defense**: while we did not cover mitigations, the hope is that shining light on how identity can be attacked – from MFA fatigue to Golden SAML – will encourage robust hardening of these systems. In the cat-and-mouse game of cybersecurity, attackers will keep innovating, but understanding their playbook is the first step for enterprises to fortify their “crown jewels.”

**References** (Offensive techniques and case studies cited)

1. Alistair G. *“MFA security part 1: How attackers bypass multi-factor authentication.”* Bugcrowd Blog, Jan 22, 2025. Describes common MFA bypass methods (phishing, conditional access policy abuse, etc.) relevant to how attackers subvert MFA in enterprises.
2. MITRE ATT&CK. *“Account Manipulation: Device Registration (T1098.005).”* Technique entry

. Documents how adversaries register new MFA devices (e.g., in Okta or Duo) to bypass MFA, with real examples.

1. DarkReading – Kelly Jackson Higgins. *“FireEye’s Mandia: ‘Severity-Zero Alert’ Led to Discovery of SolarWinds Attack.”* Mar 5, 2025

. Discusses how the SolarWinds attackers bypassed MFA at FireEye by enrolling a new device, tipping off the incident.

1. Ron Marom. *“Bypassing Okta’s Passwordless MFA: Technical Analysis and Detection.”* Rezonate Blog, June 10, 2024. Provides a step-by-step example (OktaTerrify tool) of exploiting Okta FastPass via endpoint compromise, illustrating advanced token theft and persistence.
2. Brian Krebs. *“Hackers Stole Access Tokens from Okta’s Support Unit.”* Krebs on Security, Oct 2023. Investigative report on the Okta support breach where session tokens in HAR files were stolen, and attackers attempted to create admin accounts in customers’ Okta tenants.
3. Okta Security Advisory. *“Cross-Tenant Impersonation: Prevention and Detection.”* Aug 31, 2023. Details the attack pattern where threat actors socially engineered Okta admin password/MFA resets and added a malicious identity provider for federation abuse.
4. Reco.ai (Gal Nakash). *“The MGM Resorts Cyber Attack: How Attackers Gained Highly Privileged Access.”* Sep 21, 2023

. Summary of the MGM breach and Okta’s warnings, attributing tactics to Scattered Spider (including MFA reset social engineering and device enrollment for persistence).

1. Jai Vijayan. *“Uber: Lapsus$ Targeted External Contractor With MFA Bombing Attack.”* DarkReading, Sep 19, 2022. Describes the Uber breach where LAPSUS$ used MFA fatigue (push bombing) on an external contractor, and notes LAPSUS$ similarly targeted Okta and others with such techniques.
2. Aon Cyber Labs. *“Bypassing MFA: A Forensic Look at Evilginx2 Phishing Kit.”* 2021

. In-depth technical breakdown of using Evilginx2 for adversary-in-the-middle phishing to steal session cookies, including an attack flow diagram and forensic artifacts.

1. BleepingComputer – Catalin Cimpanu. *“Golden SAML Attack Lets Attackers Forge Authentication to Cloud Apps.”* Nov 27, 2017. Explains the Golden SAML technique in plain terms and its implications (bypassing 2FA, working from outside the network, forging any user privileges).
2. Wavestone RiskInsight. *“Decoding CyberArk’s ‘Golden SAML’ Attack.”* May 2018

. A consultant’s perspective on Golden SAML, comparing it to Golden Ticket, and listing the prerequisites (stealing the IdP’s private key) and impact.

1. MITRE ATT&CK. *“Forge Web Credentials: SAML Tokens (T1606.002).”*

. Confirms that if an adversary possesses a token-signing certificate, they can forge SAML tokens with arbitrary claims (effectively Golden SAML), which is a technique used post-compromise.

1. MITRE ATT&CK – APT38 Profile. *“APT38 (aka Lazarus, Bluenoroff).”*. Describes APT38 as North Korean financially motivated hackers behind bank heists including Bangladesh, notes their operations and targets in financial sector over years.
2. Sygnia. *“Detection and Hunting of Golden SAML Attack.”* 2021. Threat research outlining how Golden SAML attacks can be detected, referencing that attackers (like the Solorigate perpetrators) used this to bypass ADFS authentication.
3. Ars Technica – Dan Goodin. *“4 Okta customers hit by campaign that gave attackers super admin control.”* Sep 2023

(Referenced via content in Okta/Reco article). Summarizes the cross-tenant attacks on multiple Okta customers, highlighting the abuse of super admin and federation.

1. CrowdStrike. *“Scattered Spider Techniques.”* Jan 2023

. As cited in Reco’s blog, describes how Scattered Spider gathered credentials (OTP codes, etc.), enrolled their own devices for MFA, and if needed fell back to phishing portals or MFA bombing.

1. Krebs on Security. *“Okta: MFA Reset Social Engineering Incidents.”* (Krebs references). Notes that around the same time as the support breach, casino hacks (MGM, Caesars) involved attackers tricking employees to reset MFA for Okta admin accounts – a tactic Okta later confirmed in an advisory.
2. Mandiant Report (via Justice.gov). *“APT38: Un-usual Suspects.”* 2018. Provides details on APT38 operations where they not only stole funds but also sometimes destroyed data in targeted banks, and notes their overlap with Lazarus.
3. DarkReading – Lawrence Abrams. *“Silver SAML: Emerging Successor to Golden SAML.”* 2021

. (Not directly cited above, but context) Explains a new variation where attackers forge SAML responses in cloud contexts (pointing that identity attacks continue to evolve post-Golden SAML).

1. Microsoft Security Blog. *“Detecting Solorigate and Related Golden SAML Attacks.”* 2021. Confirms that even if an environment evicts an attacker from the network, failing to rotate the token-signing cert means Golden SAML can persist, as seen with the SolarWinds actors.