TECHNICAL EBOOK

intellizu

MFA Done Right

What engineering teams actually need to know

MFA methods compared · Phishing-resistant auth · Enrollment & recovery
Legacy environment patterns · Rollout planning · Operational concerns

© 2025 Intellizu · intellizu.com

Table of Contents

CHAPTER 1

Why MFA is Hard

The gap between enabling MFA and having reliable, secure, user-friendly multi-factor authentication across a real organization is substantial. Anyone who has tried to roll it out broadly has felt this. The cryptography is the easy part. The operations are not.

The Gap Nobody Talks About

Most MFA guides focus on how authentication factors work — TOTP algorithms, U2F protocol details, push notification flows. That knowledge matters, but it rarely explains why organizations end up with partial MFA coverage, frequent helpdesk escalations, and shadow exceptions that quietly undermine the whole program.

The real complexity lives in: who is responsible for MFA across dozens of systems, how do you handle the 5% of users whose phone broke or who are traveling, what happens when a critical vendor application doesn't support modern authentication at all, and how do you measure whether MFA is actually providing protection versus just adding friction.

Five Reasons MFA Projects Stall

- Coverage gaps: MFA gets enabled on the main IdP but dozens of legacy systems still allow direct authentication, bypassing the control entirely.
- Recovery debt: Enrollment is prioritized, recovery is an afterthought. Then someone's device is lost and the helpdesk doesn't have a safe, documented process.
- Fragmented ownership: Security owns the policy, IT owns the identity provider, dev teams own the applications. Nobody owns the end-to-end experience.
- User friction that creates workarounds: Overly aggressive MFA prompts push users to find exceptions — shared accounts, permanent sessions, or badgering IT until they get an exemption.
- No monitoring: MFA is deployed and treated as done. Bypass attempts, failed authentications, and unusual patterns go unreviewed.

What Good Actually Looks Like

A mature MFA program has consistent coverage across systems — not just the IdP, but every path that can be used to access sensitive resources. It has documented, tested recovery processes that helpdesk can execute safely. It has monitoring that surfaces anomalies. And it has a user experience calibrated to risk — high-friction for privileged access, lower-friction for routine use — rather than maximum friction everywhere.

■ Key Insight

The question to ask is not 'do we have MFA?' but 'what paths to our critical systems bypass MFA?' Most organizations are surprised by how many they find.

Setting Realistic Expectations

Full MFA coverage across a mature organization typically takes 12–24 months when done carefully. That timeline accounts for application inventory, legacy integrations, helpdesk readiness, user communication, and exception handling. Organizations that try to do it in 90 days tend to end up with partial coverage and a backlog of urgent exceptions that erode the policy over time.

CHAPTER 2

MFA Methods Compared

Not all second factors are equal. The differences in attack surface, user friction, deployment complexity, and recovery burden are significant enough to affect which method you choose for which population. Here is an honest assessment of each.

SMS One-Time Passwords

SMS OTP sends a short-lived code via text message. It is the most widely understood method and has near-universal user familiarity. It requires no app installation and works on any phone, including basic handsets.

Weaknesses

- SIM swapping: Attackers convince carriers to transfer a target's phone number to an attacker-controlled SIM. Once successful, they receive all SMS codes. The carrier is the weak link, not your system.
- SS7 vulnerabilities: The telephony signaling protocol has known flaws that allow interception of SMS in specific attack scenarios.
- Real-time phishing: Automated phishing kits can relay codes in real time. A user entering a code on a phishing page gives the attacker seconds to use it.
- No device binding: Codes are delivered to a phone number, not a specific device. There is no cryptographic proof of possession.

SMS is better than nothing, but NIST 800-63B has deprecated it as a standalone second factor for high-assurance use cases. Use it where you have no better option, understand the risks, and plan to migrate.

TOTP (Time-Based One-Time Passwords)

TOTP generates 6-digit codes from a shared secret and the current time (RFC 6238). Users store the secret in an authenticator app and enter the rotating code at login.

Strengths: No carrier dependency, works offline, widely supported, cheap to implement, no per-authentication cost.

Weaknesses

- Phishable: A user can be tricked into entering their TOTP code on a fake login page. Real-time relay attacks work against TOTP just as they do against SMS.

- Shared secret risk: The TOTP secret must be stored server-side. A breach of that storage compromises all TOTP credentials.
- Device loss: If a user loses their phone without backup codes or cloud sync, recovery requires helpdesk intervention.
- Time sync: Codes are valid for 30 seconds. Clock drift causes authentication failures that are confusing to diagnose.

Push Notifications

Push MFA sends a prompt to the user's registered device asking them to approve or deny a login attempt. Duo, Microsoft Authenticator, and Okta Verify are common implementations. The user experience is frictionless when it works — one tap.

Weaknesses

- MFA fatigue (push bombing): Attackers with stolen credentials send repeated authentication requests hoping the user will tap Approve to stop the notifications. This attack has succeeded against major organizations including Uber and Cisco.
- Number matching mitigates this: Modern push implementations display a number on the login page that must match the number shown in the push prompt. It should be enabled.
- Requires internet on the mobile device: Push fails when the user has no connectivity.

■ Action Item

If you use push MFA, verify number matching is enabled in your configuration. The default setting in some systems does not require it. This is a simple configuration change with significant security impact.

FIDO2 / WebAuthn Hardware Keys

FIDO2 uses public-key cryptography with origin binding. The authenticator signs a challenge that includes the relying party origin. This means a credential registered at login.example.com cannot be used at login.examp1e.com — phishing the credential is cryptographically prevented.

Hardware keys are the highest-assurance second factor available for most use cases. They are not phishable, not duplicable, and do not require a network connection.

- Best fit: Privileged users, admins, executives, remote employees who regularly handle sensitive access.
- Cost: $25–$70 per key (usually issued in pairs for redundancy).
- Recovery: More straightforward than apps — losing a key means using the backup key, then re-enrolling.

Passkeys

Passkeys use FIDO2 credentials stored in platform authenticators (Face ID, Touch ID, Windows Hello) or in password managers with sync. They can replace passwords entirely, combining authentication and the second factor into a single phishing-resistant operation.

Passkeys are the direction the industry is moving. Browser support is mature. IdP support is growing. For new deployments targeting modern browsers and managed devices, passkeys are worth evaluating as the primary credential rather than an add-on.

For legacy systems and diverse device fleets, passkeys are aspirational for now. Enterprise management of passkeys is still maturing and the recovery story for enterprise-managed credentials is not fully standardized. Expect this to change over the next 2–3 years.

Method Selection Summary

Method	Phish-resistant	Friction	Cost	Legacy support
SMS OTP	No	Low	Low	High
TOTP app	No	Low	Free	High
Push + number match	Partial	Very low	Medium	Medium
FIDO2 hardware key	Yes	Low	High	Low
Passkeys	Yes	Very low	Free	Low

CHAPTER 3

Enrollment and Recovery

Enrollment and recovery are where most MFA deployments break down in practice. Getting a user enrolled is not the end of the work — it is the beginning of an ongoing operational responsibility.

Designing Enrollment Flows

The enrollment experience sets the tone for MFA adoption. A confusing or error-prone flow leads to incomplete enrollment, workaround-seeking, and IT helpdesk escalations before the rollout is even complete.

- Communicate before you require. Users who encounter an unexpected MFA prompt during work-critical moments will circumvent it if they can. Advance notice explaining what's changing, why, and what they need to do reduces both confusion and resistance.
- Provide enrollment windows, not enforcement cliffs. Give users time to enroll voluntarily before enforcement begins. Monitor enrollment rates and follow up with non-enrolled users.
- Make the default path easy. Most users will take whatever path is presented first. If that's an authenticator app, walk them through it clearly. QR code scanning is well-understood; manual secret entry is not.
- Test enrollment on actual device configurations before rollout. Corporate-managed iOS behaves differently than personal Android. MDM restrictions can block app installations.

Backup Codes: The Undervalued Safety Net

Backup codes are single-use codes issued at enrollment that allow authentication when the primary second factor is unavailable. They are simple, effective, and consistently undertreated.

Common failure modes: backup codes are generated and never saved, saved to the same device they were meant to be a backup for, or not offered by the system. Some identity platforms don't generate backup codes by default — verify this in your configuration.

- Require explicit acknowledgment that backup codes have been saved before completing enrollment.
- Document where users should store them — password manager, or printed and stored securely.
- Set a policy for when backup codes expire and require regeneration.
- Log backup code usage — it is a signal worth monitoring for misuse.

Account Recovery: The Hard Part

When a user loses access to their second factor and can't use backup codes, they need an out-of-band recovery process. This is where security and usability collide most directly.

The core risk is that recovery processes become a social engineering attack surface. If an attacker knows the recovery process — call IT, answer some questions, get access reset — they will attempt it. The recovery process needs identity verification that doesn't rely solely on knowledge factors the attacker might have.

Recovery process requirements

- Identity verification resistant to social engineering. For employees, manager approval or HR system verification is more reliable than security questions.
- Supervisor or secondary approver in the workflow. A single helpdesk agent approving a recovery is a single point of failure for social engineering.
- Audit logging with alert on recovery events. Bulk recovery requests or recovery for privileged accounts warrant immediate review.
- Post-recovery re-enrollment tracking. Recovery should result in re-enrollment, not a permanent exception.

■ Pre-Rollout Checklist

Before enforcing MFA broadly: confirm backup codes are enabled, document the helpdesk recovery process, train helpdesk on identity verification steps, and run a tabletop exercise simulating a lockout scenario.

Device Management Considerations

- Personal device BYOD: Lower cost, higher enrollment rates, operational complexity when employees leave (you can't remotely wipe a personal device's authenticator app).
- Managed device only: Better security posture and device health signals for conditional access, but requires MDM coverage — often incomplete in mixed environments.
- Hardware tokens for specific populations: High-assurance option for admins, contractors without corporate devices, and users in regulated roles.

CHAPTER 4

MFA in Legacy Environments

Many organizations have a core identity provider with MFA enabled, but access paths through legacy systems that bypass it entirely. This chapter covers practical patterns for extending MFA coverage without requiring application rewrites.

The Bypass Problem

Enabling MFA at the SSO layer protects applications that authenticate through SSO. It does not protect applications that have their own authentication, systems that accept direct LDAP binds, or services with local accounts. An attacker with stolen credentials can often still authenticate directly to these systems even when the primary IdP enforces MFA.

■ Key Question

For each critical system: what happens if an attacker has valid credentials? Can they authenticate directly to the system without going through your IdP? If yes, that is a gap regardless of whether your IdP enforces MFA.

Authentication Proxy Pattern

A proxy layer sits in front of a legacy application and handles authentication before forwarding requests. The application never changes. The proxy validates the user against the central IdP, enforces MFA, and then sets whatever session or header the legacy application expects.

This pattern works well for web applications — Apache/NGINX modules, OAuth2 Proxy, Cloudflare Access, or Pomerium can all serve this role. The proxy handles OIDC/SAML flows and translates the result into a form the legacy application understands.

- OAuth2 Proxy: Open-source, supports most OIDC providers, adds MFA requirement in front of any HTTP application.
- Cloudflare Access: Zero-config proxying with MFA enforcement at the edge. Works well for internal applications exposed through Cloudflare Tunnel.
- NGINX auth_request: Custom authentication subrequest — flexible but requires a validation service to be written and maintained.

RADIUS for Network and VPN MFA

Network devices, VPNs, and some legacy applications support RADIUS for authentication. A RADIUS proxy like Duo Security, JumpCloud RADIUS, or a self-hosted FreeRADIUS configuration can intercept RADIUS authentication requests and inject an MFA challenge.

The typical flow: user connects to VPN, enters username and password, RADIUS request goes to the proxy, proxy validates credentials and sends a push notification or prompts for a TOTP code, user approves, RADIUS access-accept is returned to the VPN.

SSO Integration for Legacy Web Apps

Some legacy applications support SAML or LDAP but not OIDC. Most modern IdPs can issue SAML assertions, so SAML integration is often the path to bring these applications into the SSO fold. The complication is attribute mapping — the IdP and application may have different expectations about how user attributes are represented.

For applications with no standard SSO support and no proxy option, the pragmatic answer is often to place them on a network segment that requires VPN with MFA to access, rather than adding MFA to the application itself.

When You Truly Can't Integrate

Some systems genuinely cannot be integrated with MFA through any available mechanism. For these, the options are: compensating controls (network isolation, privileged access workstation requirement, enhanced logging), acceptance with documented risk, or vendor negotiation and replacement.

Document exceptions explicitly, assign ownership for mitigation, and include them in regular risk reviews. A known gap with compensating controls is far better than an unknown gap discovered during an incident.

CHAPTER 5

Phishing-Resistant MFA

The terms 'MFA' and 'phishing-resistant' are often used interchangeably. They should not be. Most MFA deployments are not phishing-resistant, and understanding why matters for how you prioritize your security investments.

Why TOTP and Push Are Phishable

A TOTP code is a 6-digit number valid for 30 seconds. A convincing phishing page that captures a user's credentials can relay those credentials and the TOTP code to the real service in real time. Tools like Evilginx and Modlishka make real-time credential and OTP interception accessible to attackers with modest technical skill.

Push MFA is slightly harder to phish but susceptible to push bombing (fatigue attacks). Number matching significantly raises the bar — the user must confirm a code shown on the real login page — but it does not prevent a sufficiently motivated attack that includes a real-time proxy.

What Makes FIDO2/WebAuthn Phishing-Resistant

FIDO2 credentials are cryptographically bound to the origin at registration. When a user registers a credential at login.example.com, the public key stored server-side is scoped to that exact origin. If the user is on a phishing page at login.examp1e.com, the origin won't match, the credential won't work, and authentication fails — automatically, without requiring the user to notice anything.

This is not user-dependent protection. A user doesn't need to notice a typosquat domain or suspicious certificate. The cryptographic binding handles it. This is why CISA's guidance and the US Federal Zero Trust Strategy both require phishing-resistant MFA for privileged access.

Prioritizing Who Gets Phishing-Resistant MFA First

- Privileged accounts (domain admins, cloud root accounts, security tooling access): highest priority, smallest population.
- IT and engineering with production access: second priority.
- Executive team: frequent target of spear-phishing.
- Finance and HR with sensitive data access: high-value targets for wire fraud and data theft.
- General employee population: lower priority but include in the long-term plan.

Migration Paths

- Enable FIDO2/WebAuthn in your IdP alongside existing methods.
- Issue hardware keys or enable platform authenticators for the priority population and run a pilot.
- Create access policies that require FIDO2 for privileged actions while allowing other methods for lower-sensitivity access.
- Progressively expand the FIDO2-required scope as enrollment grows.
- Set a sunset date for TOTP/SMS in high-risk access paths.

Platform Authenticators: The Zero-Cost Path

FIDO2 doesn't require hardware keys. Modern operating systems include platform authenticators: Touch ID, Face ID, Windows Hello. These are FIDO2 credentials stored in the device's secure enclave, providing the same origin-binding protection as hardware keys.

For managed device fleets, platform authenticators are often the fastest path to phishing-resistant MFA — no procurement, no distribution, no physical key management. The limitation is that the credential is device-bound. If a device is lost, the credential goes with it.

■ Practical Note

If your IdP is Okta, Entra ID, Ping, or most enterprise-grade providers, WebAuthn/FIDO2 is already available in your subscription. Enabling it is a configuration change, not a new purchase. Check whether it's enabled before budgeting for hardware keys — platform authenticators on managed devices may cover your priority population at no incremental cost.

CHAPTER 6

Operational Concerns

MFA is not deploy-and-forget. The operational surface is significant: helpdesk load, edge cases, monitoring, and the slow drift of exceptions that accumulate over time.

Helpdesk Burden

MFA-related helpdesk tickets are predictable and can be planned for. The most common: device lost or broken, new device setup, backup codes not saved, locked out after failed attempts, and MFA not working with a specific application. Most can be addressed by self-service flows if the IdP supports them and they're properly configured.

- Self-service number update: Users should be able to re-enroll MFA after identity verification without calling IT, if the risk tolerance supports it.
- Self-service device management: Allow users to see enrolled devices and remove ones they no longer have.
- Temporary access bypass: Some IdPs support a short-term bypass code that helpdesk can issue for users who are genuinely locked out. These should be time-limited (hours, not days), logged, and audited.
- Measure ticket volume by category: Persistent high volume for 'lost MFA device' signals an enrollment or backup code problem, not just a support burden.

Travel and Connectivity Edge Cases

- TOTP works offline — it only requires a correctly synchronized clock. If travelers have app-based TOTP, connectivity is not required for MFA.
- Hardware keys work entirely offline — a strong reason to issue them to frequent travelers.
- Establish a pre-travel MFA check: ensure backup codes are saved before departure, especially for executives.
- Document your process for international teams dealing with SMS delivery failures rather than discovering it during a production incident.

Monitoring MFA Bypass Attempts

- Failed MFA attempts: A spike in MFA failures for a specific account is a signal. Persistent failures across many accounts suggest credential stuffing with stolen passwords.

- Push bombing patterns: Many push requests in a short period for one account is a clear indicator of a fatigue attack in progress.
- Backup code usage: Rare events that warrant review. Frequent backup code use suggests enrollment health problems.
- Geographic anomalies: Login from a new country combined with MFA success warrants review — especially if the geographic shift is rapid.
- Recovery events: Each helpdesk-assisted recovery should generate an alert or require human review.

Managing the Exception Backlog

Every MFA program accumulates exceptions. Some are legitimate: a service account that can't complete interactive MFA, a shared workstation scenario, a vendor system with no MFA support. Others are legacy workarounds that nobody remembers creating.

Exceptions need owners, expiration dates, and compensating controls. Review exceptions quarterly. Remove the ones that no longer have a business justification. For the rest, ensure compensating controls are real and monitored.

Rate Limiting and Lockout Policy

- For TOTP: 5–10 failed attempts triggering a temporary lockout (15–30 minutes) is a reasonable balance against brute force.
- For push: Rate limit the number of push requests per period rather than locking after failures.
- For hardware keys: FIDO2 authenticators have built-in PIN lockout — usually 8 attempts — that protects the key itself.
- Alert on lockouts: Multiple lockout events in a short window indicate an active attack worth investigating.

CHAPTER 7

Building a Rollout Plan

A successful MFA rollout is phased, measured, and designed to avoid lockout incidents. The failure mode to avoid is 'big bang' enforcement that locks out users at scale with an underprepared helpdesk.

Phase 1: Foundation (Weeks 1–4)

- Complete authentication path inventory: every system, every login mechanism, every bypass path.
- Choose MFA methods by population: what works for admins vs. employees vs. contractors.
- Configure IdP: enable MFA methods, verify backup codes are on, set up recovery flows.
- Train helpdesk: recovery procedures, temporary bypass codes, escalation paths.
- Prepare user communications: what's changing, timeline, what they need to do.

Phase 2: Pilot (Weeks 5–8)

- Enroll IT team and security team first. They will surface configuration issues before the broader rollout.
- Monitor helpdesk tickets for new categories. Unexpected friction surfaces here.
- Validate recovery processes with real test cases.
- Verify monitoring alerts are firing correctly for failure patterns.
- Assess application compatibility — any apps that break with MFA enabled need remediation before broad rollout.

Phase 3: Privileged Accounts (Weeks 9–12)

Enforce MFA for all accounts with elevated privileges before touching the general employee population. This is the highest-value coverage and the population most likely to be targeted by sophisticated attackers. The population is small enough to manage closely.

- Admin accounts, cloud root/break-glass accounts, CI/CD service accounts with production access.
- Consider hardware keys or platform authenticators for this population.
- No exceptions for privileged accounts. If a privileged account genuinely can't use MFA, it's a design problem that should be fixed.

Phase 4: Employee Rollout (Weeks 13–20)

Broad employee rollout in waves, by department or location. Give each wave an enrollment window (2 weeks is typical) before enforcement begins. Communicate directly with non-enrolled users as the enforcement date approaches.

- Track enrollment rates daily during the window.
- Have helpdesk staffing ready for the days before and after enforcement.
- Be ready to extend enforcement dates for specific groups if enrollment is below 80%.
- Keep a small quota of temporary bypass codes available for edge cases on enforcement day.

Measuring Success

- Enrollment rate by population: percentage of users with at least one enrolled MFA factor. 95%+ is the target for each enforced population.
- Coverage rate: percentage of authentication events that went through MFA. Gaps indicate bypass paths.
- Helpdesk ticket volume by category: should stabilize after rollout. Persistent high volume indicates UX or enrollment health problems.
- Exception count and trend: should decrease over time, not grow.

■ Avoiding Lockout Incidents

Enforce only after 90%+ enrollment, have a rollback plan (know how to disable enforcement quickly), and test enforcement in a non-production environment first. The most common serious incident in MFA rollouts is bulk lockout — usually caused by enforcement going live before a significant portion of users are enrolled.

How Intellizu Can Help

MFA rollouts surface broader authentication architecture questions: where are the bypass paths, how does identity management work across your stack, what happens when legacy systems can't integrate. These questions are often better answered after a structured look at how authentication actually works across your environment — not just how it's supposed to work.

Intellizu works with engineering teams and managers to stabilize and evolve production systems. Our Systems Assessment is a focused engagement that maps your current authentication and access landscape, identifies gaps, and produces a prioritized roadmap. For teams that need ongoing engineering support — building proxy integrations, configuring IdP policies, running the rollout — we

work as an engineering retainer: embedded capacity that brings the implementation alongside the strategy.

If you're working through an MFA program and want a second opinion on your approach, or a partner to help execute it, we'd be glad to talk.

intellizu.com

© 2025 Intellizu. This ebook is provided for informational purposes.