WebFrolic

WHITE PAPER

Only as Secure as the Weakest Link

What credential breaches teach us about identity security, and how WebFrolic is designed around those lessons.

Sandeep Dhankar, Founder & CTO

WebFrolic · October 2026 · hello@webfrolic.com

Contents

About this paper

PART I — THE CURRENT STATE OF IDENTITY SECURITY

What credential breaches have in common

Not all identities are equal

How expectations of identity systems have changed

The gaps that still exist

PART II — THE WEBFROLIC APPROACH

A bit of philosophy: absolute security is a myth

Making proven practices first-class

How we do it differently

Features that set WebFrolic apart

Conclusion

References

About this paper

This paper shares what we learned from studying how credential breaches have happened over many years, and the design approach we took as a result when building WebFrolic. Part I describes the current state of identity security: the recurring failure patterns, how expectations of identity systems have evolved, and the gaps that remain. Part II describes how WebFrolic addresses those gaps and where it departs from established conventions.

We welcome questions and feedback. Readers can reach the author at hello@webfrolic.com.

PART I

The current state of identity security

1 What credential breaches have in common

We reviewed how a large number of credential breaches have happened over time. Passwords were a common element in almost all of them, and the same patterns kept appearing. Industry data supports this: compromised credentials were an initial access vector in 22% of the breaches reviewed in the 2025 Verizon Data Breach Investigations Report [1].

1.1 Infrastructure failures

- **Database backups were easy to reach.** Backups often receive less protection than the production systems they copy. In the 2022 LastPass breach, attackers used credentials and keys taken from an employee to copy a backup of encrypted customer vault data from a third-party cloud storage environment [2].
- **Databases were reachable from the internet with default credentials.** In early January 2017, attackers wiped and held for ransom roughly 2,000 MongoDB databases that had been exposed to the internet without password protection, a result of insecure default configurations rather than a software flaw [3].
- **Insiders damaged or removed data.** Employees with legitimate access sabotaged or exfiltrated data, which no external control can stop on its own.

1.2 First-degree implementation failures

These are failures in the most basic controls around a password:

- **Passwords stored in plain text.** In 2009, an SQL injection flaw at RockYou exposed about 32 million usernames and passwords that were stored in plain text [4].
- **Weak password hashing.** Yahoo's 2013 breach of more than 1 billion accounts exposed passwords hashed with MD5, an outdated algorithm that can be cracked [5]. Current guidance calls for salted, deliberately slow password hashing schemes [6, 7].
- **No rate limiting.** Without limits on failed attempts, attackers can try stolen or guessed passwords at scale. NIST requires verifiers to limit consecutive failed authentication attempts to no more than 100 [6].
- **No multi-factor authentication.** OWASP describes MFA as "by far the best defense against the majority of password-related attacks" [8].

1.3 Second-degree implementation failures

These failures come from controls that exist but are implemented incompletely:

- **Backdoors into applications,** added for convenience, testing or support, which bypass normal authentication.
- **Half-built one-time passcodes (OTPs),** for example:
  - OTPs that can be reused instead of being invalidated after first use.
  - OTPs that are not bound to a single user, so a code issued to one account can be used on another.
- **No lifecycle for API keys.** API keys often keep working after the account that created them is deleted, leaving valid credentials with no owner.

1.4 Customer support operations

Customer support is often the weakest link when it comes to social engineering. The FBI and CISA describe attackers who "use social engineering techniques to convince IT help desk personnel to reset passwords and/or MFA tokens" [9].

1.5 The result: credential stuffing at scale

Together, these failures produced a large and valuable collection of stolen credentials, which made credential stuffing easy: replaying leaked username and password pairs against other sites. Two things make the attacker's job easier still: missing rate limits, and the human habit of reusing the same password for convenience. In single sign-on provider logs studied for the 2025 DBIR, credential stuffing accounted for a median of 19% of all daily authentication attempts, and 25% at enterprises [1].

2 Not all identities are equal

The impact of a stolen credential depends on whose credential it is:

Identity	What is at stake if its credentials are stolen
Customer	That customer's information and account.
Employee	The company's information and internal systems.
Administrator	The company itself: administrators can change who has access to everything else.

This introduces a third category, which we call **third-degree failures:** the abuse of privileged identities. Historically, breaches of privileged identities have come from:

- **Long-standing, unchecked permissions,** caused by the absence of user lifecycle management and access governance. Access is granted for a task and never removed.

- **Targeted phishing.** Credential stuffing and the absence of strong authentication methods make privileged accounts much easier to take over.

3 How expectations of identity systems have changed

Single sign-on products grew in the workforce because the stakes for enterprise identity are higher. Managing the employee lifecycle was the essential first step, so that an employee's access to company systems could be controlled from one place. Several technologies built on that foundation:

- **SAML and OAuth 2.0** let companies control employee authentication from a single control plane.
- **SCIM** gained popularity for automated, more granular control of the user lifecycle, provisioning and deprovisioning accounts across applications [10].
- **VPNs and device signals** improved the security posture of privileged identities by adding context about where and from what device a user signs in.
- **Role-based access control** separated permissions according to job requirements.
- **Access request, review and approval workflows** were automated to improve compliance.

3.1 The net impact

- **Almost all of this investment went into enterprise identity,** meaning employees.
- **Specialized identity tools emerged,** along with specialists to administer them.
- **Customer-facing applications gradually adopted two-factor authentication.**
  - Possession-based second factors became popular, especially SMS OTP and time-based one-time passwords (TOTP).
  - SMS OTP brought toll fraud and SMS phishing. In 2023, Twitter restricted SMS-based 2FA to paying subscribers, with its owner claiming telecom fraud through fake 2FA texts cost the company about $60 million a year [11]. NIST now classifies SMS and voice OTP as "restricted" authenticators [6].
  - TOTP authenticator apps such as Google Authenticator rose in popularity.
- **Passwords got a bad reputation.**
- **Password complexity requirements grew,** making passwords harder for anyone to remember. NIST now says verifiers "SHALL NOT impose other composition rules" such as requiring mixed character types [6].
  - This drove the need for, and the rise of, password managers.
  - A password manager changes the nature of the password as an authentication method. A password is a knowledge factor (something you know), but a long, random password that lives only in a password manager behaves more like a possession factor (something you have). Formal frameworks such as NIST SP 800-63B still classify it as a memorized secret.

4 The gaps that still exist

Because the stakes are higher, most investment has gone into safeguarding enterprise identity. Several gaps remain.

4.1 The lurking employee identity

Almost every company has a customer-facing portal for its customers and a single sign-on portal for employees to use internal applications. Employees are treated as privileged identities, but they can also have an account on the customer-facing portal, and that account often carries elevated privileges. This creates several problems:

1. A customer can become an employee and, after leaving, go back to being a regular customer.
2. People ask for elevated privileges when they become employees, but are rarely as proactive about giving them up when they leave.
3. The employee's account on the customer portal is still exposed to every attack that applies to ordinary customers, without the stronger controls of the workforce system.
4. It opens the door to social engineering, since an attacker can target a privileged account through the less protected customer channel.

4.2 Continuous change in identity provider systems

Security is not like most other product domains. A security system cannot be half-built. A typical product that delivers 90% of its features may be considered great, but a gap of even 0.1% in a security product can render the whole system ineffective. Every feature or logic change needs to be evaluated in its entirety for chain effects that could create a vulnerability somewhere else. That kind of comprehensive review is expensive, so product teams often limit the area they test. Many security bugs and breaches are the result of new features that were not tested thoroughly against the rest of the system.

4.3 Fragmented implementation of identity flows

Practice makes perfect, and expertise comes from doing the same thing again and again. Identity has a very large surface, and it is hard for any one person to have deep exposure to all of it. Unlike many products, security leaves no room for error, yet Conway's Law applies whenever multiple people and teams build a system: its structure mirrors the organization that built it. The result is often gaps between components that make chained attacks possible. A flow that works as expected in isolation can behave differently when combined with other flows in a particular order. This is fundamentally a problem of state management across flows, and AI tools have made it easier for attackers to discover and explore these chained flaws.

One common flaw: a system checks that a user has authenticated with a required number of authentication methods before a privileged action, such as modifying a resource, but lets the same user enroll a new authentication method without that check. The user can then satisfy the requirement with a method they just added, which defeats its purpose. OWASP recommends requiring re-authentication with an existing enrolled factor before any change to MFA factors, because the current session alone may have been hijacked [12].

4.4 Shared accounts

For privileged access, shared accounts are a necessary evil in the absence of a native alternative. Their audit capabilities are limited, because every action appears to come from the same account, and their credentials are often exposed or shared in plain text. Some companies store these credentials in encrypted vaults, which limits one kind of exposure but does not solve the underlying problem: several people still hold the same secret.

In many cases, people use their personal accounts to generate API keys for third-party integrations, which causes outages when the person leaves the company. Credentials for shared accounts also leak. In 2023, attackers breached the customer support system of a major identity provider using a service account credential that an employee had saved to a personal browser profile [13].

4.5 Vendor sprawl and complexity

Access request flows, user lifecycle management and audit trails are core functions that identity providers often do not provide. Companies integrate additional services to fill these gaps. The complexity of these integrations, and the need for these services to support multiple identity providers, create gaps of their own. More often than not, the lowest common denominator is chosen for end-to-end support.

- **These integrations create additional threat vectors through vendor breaches.** A system that grants permissions in an identity provider holds privileged credentials to manage permissions across the whole system. A compromise of that vendor ripples across every identity provider it connects to. In 2025, attackers stole OAuth tokens from a sales chat integration and used them to export data from many companies' Salesforce instances, searching it for credentials such as AWS access keys [14].
- **Workarounds and unregulated backdoors** are created to keep things running when one of these integrations fails.

4.6 Holes in the security policy

- **Exceptions.** A security policy is often strong on paper, but exceptions are made for various reasons, and each exception creates a hole.
- **Unused privileges.** Some users hold privileged permissions they never use, which adds risk with no benefit.
- **Misunderstood privileges.** Some users hold privileged permissions without understanding their significance and impact.

PART II

The WebFrolic approach

WebFrolic was built to address these gaps and several more. The following sections walk through the high-level design decisions behind the product.

5 A bit of philosophy: absolute security is a myth

We start from the understanding that absolute security is never practical, especially for online systems connected to other systems. Three ideas follow from that:

1. **Apply the right amount of friction.** We decide how much friction to present to a user based on what we know: their device, the resource being accessed, their credentials, their usage behavior and other signals.
2. **Keep an audit trail for every action.** A complete record lets us inspect, learn and follow up. The fear of being caught also deters many bad actors from acting in the first place.
3. **Contain the blast radius.** Security incidents will happen. Even the most secure system depends on how it is configured and who has access, and both involve people. The system should detect incidents quickly and offer ways to limit their impact. For example, if a flawed authenticator allowed people to create accounts, it should be easy to report on every account created through that authenticator and to require additional verification from those users.

6 Making proven practices first-class

We studied the root causes of many breaches to learn what fails, and we also looked at the common security controls that work as expected. One of our guiding principles was to take those proven practices and make them first-class constructs in the product, rather than optional add-ons.

6.1 Rate limits

Rate limits limit the impact of an incident in two ways:

1. **They remove the incentive for credential stuffing.** Credential stuffing relies on the probability that people reuse credentials across systems, which only pays off when attackers can make many attempts. Rate limiting makes the attack a poor choice on cost and benefit [8].

2. **They slow down a compromised privileged account.** If a privileged account is taken over, rate limits stretch the time needed to make large-scale changes. That window gives defenders a chance to respond before much damage is done.

WebFrolic provides rate limits designed from real-world usage at several levels, each of which a business can customize:

Level	How it works
User	Limits apply per user. Some users, especially service users, legitimately need higher limits, so the user-level limit is configurable.
Session	Usage across all of a user's sessions counts toward that user's limit, so opening more sessions does not raise it.
Tenant	Because WebFrolic is delivered as a service, tenant-level limits prevent one tenant from affecting others (the "noisy neighbor" problem).
IP address	Many legitimate customers sit behind shared proxies, so businesses can register proxy CIDR ranges and treat them accordingly. Attackers also spread traffic across large proxy pools, which is why IP limits are combined with the other levels [8].
Operation	Specific operations, such as account creation and account recovery, have their own limits to curb particular kinds of abuse.

6.2 Network firewalls

IP-based network firewalls are a critical part of infrastructure security and are widely used to protect production systems and databases. WebFrolic brings the same construct to nearly every configuration and feature in the identity provider. Allow lists and deny lists of IP addresses and CIDR ranges are supported for:

1. **Roles:** a role can be tied to a network filter that controls where the role can be assumed from.
2. **Identity types:** separate network rules for customers, employees and administrators are evaluated at authentication to decide whether the user may sign in from their current network.
3. **Tenants:** in B2B scenarios, each tenant can set network filters for its own members.
4. **SCIM integrations:** provisioning requests are accepted only from approved networks.
5. **Tenant integrations:** applications that manage tenants through an integration can be limited to approved networks.
6. **Application access:** access to individual applications can be restricted by network.
7. **Service accounts:** each service account can have its own network rules.
8. **API keys:** an API key bound to an IP range is useless to anyone outside that range, even if the key itself leaks. This matters even more for AI agents, which are often granted access to third-party systems through API keys. We consider this one of the most important controls in the product, and a separate article

> describes how WebFrolic secures API keys in detail.

6.3 Audit trails

Audit records are essential for handling incidents, meeting compliance requirements and understanding usage patterns. In many systems they are added incrementally, as demand arises or after someone notices they are missing. We took the opposite route and started with them. Audit capture is part of the architecture itself, so no action can happen without being recorded.

Every action on every resource is captured in an immutable audit trail, together with the user, session and device information.

6.4 Visibility into active and past sessions

Seeing and managing sessions is a critical feature of any identity system: it lets people verify how an identity is being used and revoke access when needed. In WebFrolic, administrators and end users alike can see and control their active and past sessions. The same visibility and control extends to API keys.

6.5 Managing state on the server

State is central to identity flows such as signup, account recovery, authentication, credential enrollment and OAuth 2.0 authorization. Most of these flows implicitly require a series of steps in a given order. The state records which steps have been completed, which step is current and which remain.

Our research found that many identity systems keep this state on the client, which makes them prone to chained attacks: anything held on the client can be manipulated by the client. Many systems also store only the current step rather than the full history of the flow. That looks simpler, but it makes chained attacks easier.

WebFrolic stores all state on the server. It also stores the extended state, meaning past steps, the current step and the possible next steps, along with validation rules for the transitions between them. This protects against client-side manipulation and chained attacks.

7 How we do it differently

7.1 Separate ceremonies and flows

It is tempting to use the same path for resetting a password during account recovery, changing it on purpose and setting it at signup. Without strong state management, that shared path becomes an opening for chained attacks.

WebFrolic separates flows into three categories, each with its own preconditions and state rules:

Flow	Purpose
ENROLLMENT	An authenticated user adds a credential to their account. The flow first confirms an authenticated session before the user can enroll a new credential.
SIGNUP	A new user creates an account. The flow checks the person against existing unique identifiers to detect an existing account, and evaluates the policy rules for profile attributes and credential protocols, such as employee or tenant email domains. To finish, the user must enroll a credential that supports registration and complete their profile as the policy requires. A successful signup also triggers workflow automation rules, which can match combinations of profile attributes to provision the account.
VERIFICATION	An interim state in which the user proves their identity by verifying credentials, building up their claim step by step before an authenticated session is issued. Verification transactions are limited in time and bound to the session.

Credentials are also categorized by the ceremonies they can take part in, based on their protocol. A TOTP, for example, cannot be used for registration, because it carries no information that identifies the user.

Ceremony	Purpose
REGISTRATION	Creating an account with a credential that can identify the user.
AUTHENTICATION	Proving identity to sign in.
RECOVERY	Regaining access to an account using a credential designated for recovery.

Explicit recovery ceremonies and recovery credentials close a major gap in existing systems, where account recovery often succeeds with only part of a user's credentials.

7.2 Identity flows as transactions

Our industry applies transaction rules to sensitive operations such as database changes and payments. Transactions ensure that multiple operations happen in a strict order, and sometimes the transactions themselves must be batched and coordinated in a particular sequence.

Identity flows have a lot in common with this. WebFrolic models every type of flow as a transaction, including registration, authentication, step-up authentication, recovery, recovery enrollment and role selection. Depending on the context, these transactions can be combined in different orders. For example, a single OpenID Connect sign-in can start a chain of nested transactions, which then complete in reverse order:

Started, in order	Completed, in order
1. OIDC sign-in transaction	1. Recovery enrollment completed
2. Authentication transaction	2. CAPTCHA challenge completed
3. Step-up authentication	3. Step-up authentication completed
4. Risk check	4. Authentication completed
5. Recovery enrollment	5. Network rules evaluated
6. User profile update	6. Session created
	7. OIDC token returned

Managing the dependencies between transactions and executing their steps in a valid order is critical. Implementing the flows is only half of the story; the other half is capturing an audit trail that gives a complete picture. WebFrolic records each transaction within a session request, how the transactions relate to each other and the order in which they ran.

7.3 All identities in one place, with different rules

When identities are spread across multiple systems, companies have to map the same person across those systems, which creates additional attack surface and gaps. The problem is serious and was hard to solve, so most products chose to support either customer identity or enterprise identity, but not both.

We took an opinionated approach and classified identities into three high-level categories:

Classification	Who they are
Customers	End users of the business's online services and websites.
Employees	People employed by the company. They have higher privileges and can manage customer identities.
Administrators	People who manage the identity systems themselves.

Separating customers from employees was straightforward. It requires support for different profile attributes per category: an employee may have an employee ID that does not apply to customers, and vice versa, while both share common attributes. Lifecycle rules can also differ. For example, employees cannot deactivate or delete their own accounts, while customers can.

For administrators, we evaluated real-world use cases and spoke with customers. Administrators are often people with specialized skills and higher privileges who may or may not be full-time employees. That led to a separate classification with its own rules, including a setting that controls whether the organization allows non-employees to be administrators.

The following features make it practical to manage these different identities in one place:

1. **User segment types,** which apply to:
    - Authenticator assignments. For example, hardware-bound passkeys can be required for employees while customers can enroll any passkey.
    - Application assignments.
    - Group assignments.
    - Role assignments.
2. **Authenticator assurance level requirements,** configurable for each classification (Section 7.4).
3. **Network firewall rules,** configurable separately for each classification (Section 6.2).
4. **Session controls,** configurable for each classification:
    - Maximum session duration.
    - Inactive session timeout.
    - Maximum number of active sessions.
5. **Profile attributes,** so different information can be captured for each classification. An employee profile might require attributes such as employee ID and department that do not apply to customers, while other attributes are common to both.
6. **Separate dashboards** for administrators, employees and customers. Classification rules are evaluated on each dashboard's paths to enforce access.

7.4 Authenticator assurance levels

Not all credentials are the same. Based on their characteristics, authenticators can be classified as knowledge, possession or inherence factors, or a combination of them. WebFrolic lets businesses specify the required authenticator assurance level across many parts of the product. We believe this fills a critical gap in how MFA is generally understood today: not all combinations of two factors offer the same protection.

For the most part we follow NIST SP 800-63B-4 [6]. To reflect real-world usage, we added two intermediate levels. For example, two possession factors such as TOTP and SMS do not qualify for AAL2, which requires two distinct factor types, but together they are more secure than a single TOTP. We call this level AAL1_Strong. This balances flexibility, ease of use and security.

WebFrolic level	Requirement	Relation to NIST SP 800-63B-4 [6]
None	No authenticator required.	Not a NIST level.
AAL1	Any knowledge or possession authenticator.	Corresponds to AAL1.
AAL1_Strong	Any two authenticators of the same type, either knowledge or possession.	WebFrolic extension between AAL1 and AAL2.
AAL2	Any two authenticators of different types.	Corresponds to AAL2, which requires two distinct factors. NIST also requires verifiers to offer a phishing-resistant option at AAL2.
AAL2_Strong	Any two authenticators of different types, with at least one phishing-resistant.	WebFrolic extension between AAL2 and AAL3.
AAL3	Two or more authenticators of different types, with at least one phishing-resistant and device-bound.	Corresponds to AAL3, which requires a cryptographic authenticator with a non-exportable private key; syncable authenticators are not allowed.

7.5 Authenticator characteristics

Just as not all identities are equal, not all credentials are equal. Every authentication protocol depends heavily on how the authenticator implements it, and that is mostly on the client side. From the server's perspective, we rely on the signals and characteristics we can verify.

- **Administrator-defined characteristics.** When there are not enough verifiable signals, administrators can specify an authenticator's characteristics. For a SAML authenticator, for example, an administrator can declare whether the authentication should be treated as knowledge, possession, inherence or phishing-resistant.
- **Detailed options for each protocol:**
  - OTP: configurable code length and character types.
  - Email: allow lists and deny lists of domains, and an option to accept business domains only.
  - WebAuthn (passkeys): passkeys are distinguished as synced passkeys, which are copied across a user's devices, and device-bound passkeys, such as those on security keys, which never leave a single device [15]. Administrators can bring their own root certificates for custom attestation, choose whether to validate attestation and user verification, and maintain allow lists and deny lists of authenticator models by AAGUID.

Most traditional systems support only one enrollment of a given protocol per user. WebFrolic supports multiple enrollments for each protocol, so a user can register, for example, more than one passkey.

For credentials such as passwords, where only one should exist, the product enforces a maximum of one enrollment. Adding a new password deletes the previous one.

7.6 Credential management

Past breaches show that account takeover usually happens in one of three ways:

- The attacker takes over the account through the account recovery route.
- The attacker uses a hijacked but valid session to enroll their own credentials, then signs in with them [12].
- The attacker uses publicly accessible password reset links.

Our approach to credential management is built on giving users better visibility into, and communication about, every credential change. The main design decisions are:

1. **Dedicated recovery methods.** Account recovery uses its own route and its own credentials, instead of letting a user recover with only part of their normal credentials.
2. **Recovery rules by classification.** Recovery rules can differ for customers, employees and administrators:
    - **Required assurance level for recovery.** The level is checked at signup and at authentication. Users who do not meet it are prompted to enroll additional recovery credentials until they do.
    - **Option to delete authentication credentials after a successful recovery.** This gives a clean break after recovery. In traditional systems, an attacker holding leaked credentials can keep using them even after the real owner recovers the account. It also reflects a simple assumption: a user who goes through recovery has already lost their credentials, so it is safest to remove them.
    - **Option to lock the account for a period after recovery.** For sensitive users such as employees and administrators, this gives the real owner time to see the recovery notifications and act if someone else is taking over the account.
3. **Support staff can only add recovery credentials.** By design, support staff can help a user regain access only by enrolling a recovery credential, not by resetting existing credentials. The real account owner therefore still receives recovery notifications, even if an attacker manages to use the privileged support route [9].
4. **Recovery revokes all API keys.** Completing a recovery revokes every API key the user holds.

The assurance level trap

If a user does not have enough credentials enrolled, they are prompted to enroll more until they meet the required assurance level. The highest assurance level the user has attained for any role, application or dashboard is recorded on their profile.

That same level then applies whenever the user wants to change their credentials. This can trap users who just want to update a credential or add information: because they once assumed a role that required a high assurance level, they now need that level for routine changes. From a security perspective, however, this rule is critical.

One way out is the recovery flow, but recovery is destructive because it revokes all of the user's API keys. For these cases, administrators with higher privileges can delete a credential that the user no longer has access to. The user can then enroll new credentials without going through recovery.

7.7 Service users

This single feature fills one of the biggest security gaps in many systems today. Service users are like regular users, with these exceptions:

- **No credentials.** They have no password or key that can be shared or leaked.
- **Time-limited assignment.** They are assigned to regular users for a specified duration.
- **One user at a time.** Only one person can assume a service user at any moment.
- **Traceable activity.** Every action taken as a service user is recorded along with the original user and their original session.
- **Full functionality.** Once a user assumes a service user, they can use the rest of the platform as that identity, including signing in to third-party single sign-on applications. This is what distinguishes a service user from a privileged role.

Service users can also have their own network firewall rules (Section 6.2) and their own authenticator assurance level requirement (Section 7.4).

Use cases

- **Third-party service integrations:**
  - People often use their personal accounts and personal API keys to integrate with third-party services, or create a placeholder account whose credentials are shared among several people.
  - These accounts usually carry broad privileges so that the integration works. Service users remove the need for shared credentials and therefore the risk of those credentials leaking. Several high-profile breaches trace back to leaked shared or service account credentials [13].
  - Integrations no longer break when the person whose account they depended on leaves the organization.
- **Break-glass access:** a service user can hold elevated privileges and be assigned to several regular users, who assume it only when needed.
- **No standing privileges:** how are service user privileges different from role privileges? When someone assumes a role, their identity stays the same. When someone assumes a service user, they act as that identity, and the identity carries across connected applications. That is not possible with roles.
- **AI agents:** AI agents behave much like regular users in their online activity, so service users are a natural way to model agent accounts. With machine identities already outnumbering people 82 to 1 in the average organization [16], this is becoming a requirement rather than an option.

7.8 A lifecycle for every resource

We evaluated the requirements, challenges and pain points that users of many identity systems have shared with us. A common theme: onboarding and joiner flows are always available and easy, but change, leaver and departure flows rarely get the same attention.

We made sure every resource has a defined lifecycle covering creation, provisioning, change, deprovisioning and deletion. Some examples:

Terms of service and privacy policy

- Administrators choose whether the terms of service are shown only at signup or at every authentication.
- Administrators set the current terms of service version. Users who have not yet accepted that version are asked to accept it when they sign in.

Profile attributes

- Requirements change over time. Attributes defined today may need to change, and more information may be needed later.
- Every change to the profile attribute configuration automatically increments a version number.
- Administrators can require a specific profile version. Users are then guided to complete any missing or new attributes, or to update values that no longer pass validation.

Authenticators

- Administrators can mark each authenticator as optional, recommended or required for each user segment.
- Administrators can deprecate an authenticator.
- Every change to the authenticator configuration automatically increments a version counter. Administrators can require users to comply with a specific version, and the policy is evaluated and enforced at every sign-in.

Users

- WebFrolic offers configurable ways to onboard different types of users through different mechanisms, according to the business's needs.
- End users can deactivate their accounts, with a configurable period before the account is deleted. Users can reactivate their account at any point before deletion.
- User invitations expire after a configurable period if they are not accepted.

7.9 Everything in one place

As described in Section 4.5, identity and compliance requirements are rarely met by one system. They are met by integrating several systems and making them work together. Two use cases show the risk:

- **Access requests and reviews.** A downstream provisioning system lets employees request access to roles and applications. To do this, it integrates with the identity system and the resource systems, with the ability to read and change data across all of them. Its integration API keys are highly privileged and a serious risk if stolen [14].
- **Customer support.** Every identity system has customers who run into account problems. Support staff can see account information and can also manage user credentials.

Our approach is to reduce this risk surface by bringing the functionality into the identity system itself. That removes integration challenges and their risks, and it is simpler and more efficient. With this in mind, WebFrolic includes the following modules:

- **Access request workflows:** an employee service desk portal where staff can browse and request groups, applications and secrets marked as discoverable. Groups can be self-service or require approval, every membership change records the request it came from and the workflow that handled it, and access can be reviewed and removed when it is no longer needed.
- **Automations:** workflow automations manage employee lifecycle flows and provision resources automatically based on group membership, with a record kept of every action taken.
- **IT support desk for employees:** internal IT requests are handled as tickets inside the identity system. Unlike the customer support desk, it serves employees, and each ticket category can have its own automated workflow. This balances automation with human oversight for critical privilege assignments.
- **Customer support desk:** identity-related requests from both signed-in and signed-out customers are tracked as tickets. Support staff can manage or impersonate a customer's profile only while a ticket for that customer is assigned to them and active, and every action they take is audited together with the ticket that granted the access. These controls work alongside the recovery rules in Section 7.6.

7.10 Built with security and compliance in mind

In many areas, WebFrolic goes beyond the minimum requirements.

Expiring access and assignments

- Every resource association and access operation can be given an expiration time.
- A setting lets administrators require the person making a change to enter a note explaining it.

Reducing liability

- **Password storage.** Passwords are hashed with Argon2id, the algorithm OWASP recommends first [7], and the resulting hash is then encrypted. This is a form of peppering also used by Dropbox [17]. The encryption key is held in a keystore that only the application can access, and the keystore password is injected into the environment at runtime and then removed. A stolen copy of the database alone is therefore not enough to start cracking passwords.
- **Client secrets and API tokens are shown only once,** when they are generated. WebFrolic stores only a hash of each secret, so the product is not a honeypot for attackers.

- **Signed sensitive records.** All sensitive records are signed with immutable keys, and the signatures are verified whenever the records are used. Even a rogue person with database administrator credentials cannot change permissions or credentials without detection.
- **Bring your own key (BYOK).** Every customer can supply their own encryption key to control the security of their data. This places more responsibility on the customer to protect that key.

Granular controls

- Every resource has common controls for access, association and audit.
- Custom roles can be created at the global level or at the resource level, with explicit role assignments.
- Administrators can specify authenticator characteristics where needed (Section 7.5).

We use our own product

- We use WebFrolic to manage our own customers and employees.

8 Features that set WebFrolic apart

8.1 Network firewalls for API keys

We believe binding API keys to network ranges should be a baseline requirement for every security product. Some platforms already offer this for their own keys: Stripe lets customers restrict secret and restricted API keys to specific IP addresses [18], and AWS IAM policies can deny requests from outside specified IP ranges [19]. It is still uncommon for identity providers to offer it for the keys they issue, and we hope to see wider adoption given its advantages.

8.2 Assurance levels and network rules for global roles

Not all identities are equal, and the same applies to roles, especially highly privileged ones. WebFrolic lets businesses require that privileged roles be assumed only from certain IP ranges, and only after meeting a higher authenticator assurance level. We believe this fills a significant gap in identity systems today.

8.3 API keys tied to the role lifecycle

Most identity products now let a user hold multiple roles, each with different privileges. In WebFrolic, an API key created while a user has assumed a role is tied to that role's permissions and lifecycle. If the user's role assignment is removed, the API keys created under it are revoked as well.

8.4 Encryption for profile attributes

Almost every identity solution manages user profile information, and some attributes need secure storage and limited visibility. Restricting who can see an attribute based on their role helps a great deal, but those rules are hard to apply uniformly across internal APIs and data exchanges.

Explicit encryption for sensitive attributes, with visibility gated by permissions, is a step up from protection that can be bypassed by accident. WebFrolic provides this out of the box, along with controls to define and validate data types for profile attributes.

9 Conclusion

Most of the breaches described in Part I did not require new attack techniques. They exploited ordinary weaknesses: unprotected backups, weak password storage, missing rate limits, shared credentials, flows that could be chained together and support processes that could be talked around. WebFrolic is designed around those lessons. It treats rate limits, network rules, audit trails and server-side state as core constructs, separates the ceremonies that are often merged, gives every resource a lifecycle and brings customers, employees, administrators and service users into one system with rules suited to each.

WebFrolic is in early access. To discuss your identity and security posture, contact hello@webfrolic.com.

References

[1] Verizon Business, "Additional 2025 DBIR research on credential stuffing," 2025. https://www.verizon.com/business/resources/articles/credential-stuffing-attacks-2025-dbir-research/

[2] LastPass, "Notice of Security Incident," December 22, 2022. https://blog.lastpass.com/posts/notice-of-recent-security-incident

[3] The Hacker News, "Someone Hijacking Unsecured MongoDB Databases for Ransom," January 4, 2017. https://thehackernews.com/2017/01/secure-mongodb-database.html

[4] The Register, "RockYou admits security snafu exposed email login details," December 17, 2009. https://www.theregister.com/2009/12/17/rockyou_security_snafu/

[5] PCWorld, "Yahoo reports massive data breach involving 1 billion accounts," December 14, 2016. https://www.pcworld.com/article/3150670/security/yahoo-reports-massive-data-breach-involving-1-billion-accounts.html

[6] NIST, SP 800-63B-4, "Digital Identity Guidelines: Authentication and Authenticator Management," August 2025. https://pages.nist.gov/800-63-4/sp800-63b.html

[7] OWASP, "Password Storage Cheat Sheet." https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html

[8] OWASP, "Credential Stuffing Prevention Cheat Sheet." https://cheatsheetseries.owasp.org/cheatsheets/Credential_Stuffing_Prevention_Cheat_Sheet.html

[9] FBI and CISA, "Scattered Spider," Joint Cybersecurity Advisory AA23-320A, November 16, 2023. https://www.cisa.gov/sites/default/files/2023-11/aa23-320a_scattered_spider.pdf

[10] IETF, RFC 7644, "System for Cross-domain Identity Management: Protocol," September 2015. https://datatracker.ietf.org/doc/html/rfc7644

[11] IT Brew, "Twitter lands itself in the middle of the SMS 2FA debate," March 9, 2023. https://www.itbrew.com/stories/2023/03/09/twitter-lands-itself-in-the-middle-of-the-sms-2fa-debate

[12] OWASP, "Multifactor Authentication Cheat Sheet." https://cheatsheetseries.owasp.org/cheatsheets/Multifactor_Authentication_Cheat_Sheet.html

[13] BleepingComputer, "Okta breach: 134 customers exposed in October support system hack," November 2023. https://www.bleepingcomputer.com/news/security/okta-breach-134-customers-exposed-in-october-support-system-hack/

[14] Security Affairs, "UNC6395 targets Salesloft in Drift OAuth token theft campaign," August 28, 2025. https://securityaffairs.com/181632/hacking/unc6395-targets-salesloft-in-drift-oauth-token-theft-campaign.html

[15] FIDO Alliance, "Passkeys." https://fidoalliance.org/passkeys/

[16] CyberArk, "2025 Identity Security Landscape" (press release via Silicon UK), April 23, 2025. https://www.silicon.co.uk/press-release/machine-identities-outnumber-humans-by-more-than-80-to-1-new-report-exposes-the-exponential-threats-of-fragmented-identity-security

[17] Dropbox, "How Dropbox securely stores your passwords," September 21, 2016. https://dropbox.tech/security/how-dropbox-securely-stores-your-passwords

[18] Stripe, "Best practices for managing secret API keys." https://docs.stripe.com/keys-best-practices

[19] AWS re:Post, "How do I restrict access to AWS resources based on the AWS Region, source IP address, or Amazon VPC?" https://www.repost.aws/knowledge-center/iam-restrict-access-policy

> **WebFrolic is in early access.** To discuss your identity and security posture, contact hello@webfrolic.com.