International Journal of Computational Science and Engineering Research
ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/
Regular Issue , Vol. 3, Issue. 4 (October – December) , 2026 , Pages: 23 - 31
Received: 26 April 2026 ; Accepted: 29 October 2026 ; Published: 09 November 2026
https://doi.org/10.63328/IJCSER-V3RI4P3
https://www.youtube.com/watch?v=jj_Li715M14
https://archive.org/details/jsp-ijcser-v3ri4p3
Original Paper : Applied Research – Experimental Research Paper

Short-Lived Keys for Long-Running Agents: Zero-Trust Access for AI Assistants

Marpu Jagadeesh

Department of Information Technology , St Francis College Brookyln , New York, USA

** Corresponding Author: Marpu Jagadeesh ; jmarpu@sfc.edu*

Abstract: An AI assistant that can read mail, call APIs and move money is only as safe as the credentials it carries, and credentials leak: through logs, through prompts, through a compromised plugin. This paper asks what an assistant should hold so that a leak costs as little as possible, and what that costs to run. We compare five designs: a static key, an eight-hour session token, short-lived tokens scoped to one tool, scoped tokens bound to the agent's own key, and one-time tokens bound to a single action. We prove that the expected exposure from a theft is E[min(R, D)], where R is the token's remaining life and D the time to detect the misuse, give it in closed form, and show that the cost-optimal lifetime is bounded below by 2b/a. In a simulation of credential theft, the expected loss per theft was about 294,175 units for a static key, 2,732 for a session token, 115 for a 15-minute scoped token, 11.4 when that token was also bound to the agent's key, and 0.8 for one-time tokens. Over 99.4% of the static key's loss came from thefts that were never detected, which is the real argument for expiry: it caps the damage when detection fails. We also built a prototype of attenable, HMAC-chained capability tokens with proof of possession and ran 18 security tests of 1,000 trials each. All behaved as designed, including one that shows the limit of the approach: a stolen bearer token works until it expires. Verifying a token took about 10 μs, or about 167 μs with proof of possession. All loss figures are simulated under stated assumptions.

Keywords: Zero Trust, Capability Tokens, Least Privilege, Credential Theft, AI Agents, Proof Of Possession

1. Introduction

Zero-trust security replaces the idea of a trusted network with checking every request, and prefers credentials that are narrow, short-lived and tied to the party that should be using them. The principle of least privilege is much older [1]. For AI agents the question is how far to push it. Credentials that live for seconds and cover one action are safer, but they mean a request to a token service for every action, and a service that is down can stop the agent. We make five contributions. A closed-form expression for the expected exposure from a stolen credential as a function of its lifetime and the time to detect misuse, including the case where detection sometimes fails (Section 4). A bound on the cost-optimal token lifetime, and a demonstration that the common back-of-envelope formula is a lower bound, not an estimate [2]. A simulation comparing five credential designs on expected and tail loss, issuance load and failures from broker outages (Section 6). A working prototype of attenable capability tokens with proof of possession and one-time use, with measured costs and 18 security tests (Section 7). A plain statement of what these designs do not do: in particular, bearer tokens remain usable by a thief until they expire [3]. The loss figures are from a simulation with assumed parameters, and the prototype runs in a single process. Neither is a measurement of a deployed system.

2. Background and Related Work

Least privilege and capabilities: Saltzer and Schroeder named least privilege as a design principle in 1975 [1], and capabilities as unforgeable tokens of authority date from Dennis and Van Horn [2]. Our tokens are capabilities that can be narrowed but not widened. **Bearer and bound tokens:** OAuth [4] and JSON Web Tokens [5] are the common basis for API access. They are bearer tokens: whoever holds one can use it. Proof-of-possession schemes such as DPoP tie a token to a key held by the client, so that a stolen token alone is useless [6]. We implement the same idea.

Jack Sparrow Publishers © 2026, IJCSER, All Rights Reserved
www.jacksparrowpublishers.com

ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/ , Vol. 3, Issue 4 , 2026 , https://doi.org/10.63328/IJCSER-V3RI4P3

Macaroons: Birgisson and colleagues proposed to-kens made of a chain of keyed hashes, to which anyone can add restrictions (caveats) but from which nobody can remove them without the root key [7]. This suits agents, which often delegate part of their authority to a sub-task or a sub-agent. Our prototype follows this design.

Zero trust: NIST's zero-trust architecture describes per-request authorization, with policy decisions made by an engine and enforced at a gateway [3]. Our broker and gateway are an instance of it.

Threats specific to agents: Indirect prompt injection can cause an agent to leak or misuse what it holds [8]. A credential in the agent's context, or in a tool's logs, can be extracted. This is the main reason we model theft as an ordinary event and not an exotic one.

What is new here: The designs are standard. We contribute the quantitative comparison, the exposure law

Proposition 1 (Expected exposure)**.** $E(T) = \mathbb{E}[\min(R,D)]$ and the bound on the optimal lifetime, and the measured costs in the context of agent workloads

$$E(T) = (1 - p_{\text{miss}})d\left(1 - \frac{d}{T}\left(1 - e^{-T/d}\right)\right) + p_{\text{miss}}\frac{T}{2} \qquad (1)$$

3. Design and Threat Model

3.1. Parties

An *agent* performs tasks using *tools*. A *broker* issues tokens after consulting a policy that maps a task to a scope. A *gateway* sits in front of every tool and verifies each request. A *detector* watches for misuse and adds (1)

$$E(T) \approx \frac{T}{2} \quad \text{when } T \ll d$$

it is close to T/2. For

$$E(T) \to d \quad \text{when } T \gg d \text{ and } p_{\text{miss}} = 0$$

it tends to d. Proof. For non-negative variables,

$$\mathbb{E}[\min(R,D)] = \int_0^\infty P(R > t)P(D > t)\,dt$$

With detection,

$$P(R > t) = 1 - \frac{t}{T} \quad (0 < t < T)$$

$$P(D > t) = (1 - p_{\text{miss}})e^{-t/d} + p_{\text{miss}}$$

The integral of the first part is

$$\int_0^T \left(1 - \frac{t}{T}\right)e^{-t/d}\,dt = d - \frac{d^2}{T}\left(1 - e^{-T/d}\right)$$

, and of token identifiers to a revocation list that the gateway consults. Figure 1 shows the flow and Figure 2 the steps for one action. the second is Eq. (1). Combining gives

$$\int_0^T \left(1 - \frac{t}{T}\right)dt = \frac{T}{2}$$

3.2. Token Designs :

Static key. Never expires except at rotation, every 90 days. Covers all tools.
Session token. Lasts 8 hours and covers all tools.
Scoped token. Covers one tool and lasts T minutes, refreshed as needed.
Scoped and bound. As above, plus a caveat naming the hash of the agent's public key. Each request must carry a signature by that key over the token and the action.
One-time token. Bound to one action hash and one nonce, valid for seconds, accepted once.

3.3. Threat model

A theft event gives an attacker a copy of whatever credential the agent is holding at that moment. The attacker starts using it at once, at a rate that depends on the tools in scope, until the token expires or the misuse is detected. Detection takes a random time with mean d, and with probability p_{miss} it never happens [8]. We do not model theft of the broker's root key or of the gateway. A stolen *key* (for the bound designs) occurs together with the token with probability p_k. We assume the gateway and broker are correct and that clocks are synchronized closely enough for expiry to work.

4. Research Related Work

Let T be the token lifetime. If theft happens at a uniformly random moment of the agent's activity, the token's remaining life R is uniform on $(0, T)$. Let D be the time until misuse is detected, exponential with mean d with probability $J(T) = aE(T) + \frac{b}{T}$ and infinite otherwise. The attacker's value extraction rate is κ per minute. The loss from a theft is $\kappa \min(R, D)$.

The proposition has a consequence worth stating. With $p_{\text{miss}} > 0$, exposure grows linearly in T forever, with slope $p_{\text{miss}}/2$, whatever the quality of detection on the other thefts. A long lifetime is dangerous because it gives unlucky thefts, the ones nobody notices, a long time to run.

Proposition 2 (Loss scales with scope and lifetime)**.** *Expected loss per theft is $\kappa E(T)$. For $T \ll d$ and $p_{\text{miss}} = 0$, halving the scope (so halving κ) and halving T have the same effect on loss.*

This follows from $E(T) \approx T/2$ for $T \ll d$. The two levers differ in cost: narrower scope

Jack Sparrow Publishers © 2026, IJCSER , All Rights Reserved
www.jacksparrowpublishers.com

ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/ , Vol. 3, Issue 4 , 2026 , https://doi.org/10.63328/IJCSER-V3RI4P3

costs more token requests per task (one per tool), and shorter lifetime costs more refreshes.

Proposition 3 (Cost-optimal lifetime). Suppose N agents each refresh n_s scoped tokens continuously, every T minutes, at a cost c per request. Thefts occur at rate θ per agent per day with extraction rate κ. Daily cost is J(T) = a E(T)+b/T with a = Nθκ and b = Nn_s ·1440·c. Then every minimiser of J satisfies

$$T_0 = \sqrt{\frac{2b}{a}} \qquad (2)$$

Proof. Differentiate

$$E'(T) = \frac{1}{T^2}\int_0^T t\,P(D > t)\,dt \le \frac{1}{2},$$

which is at most because $P(D > t) \le 1$. At

$$T_0 = \sqrt{\frac{2b}{a}},\ \text{we have}\ \frac{b}{T_0^2} = \frac{a}{2},\ \text{and}$$

$$J'(T_0) = aE'(T_0) - \frac{b}{T_0^2} \le 0.$$

Since J' is non-positive at T_0 and J is minimized where J' crosses zero from below, no minimizer can lie below T_0 (taking the smallest minimizer if there are several). The usual textbook shortcut treats exposure as $T/2$ and gives $T_0 = \sqrt{2b/a}$.

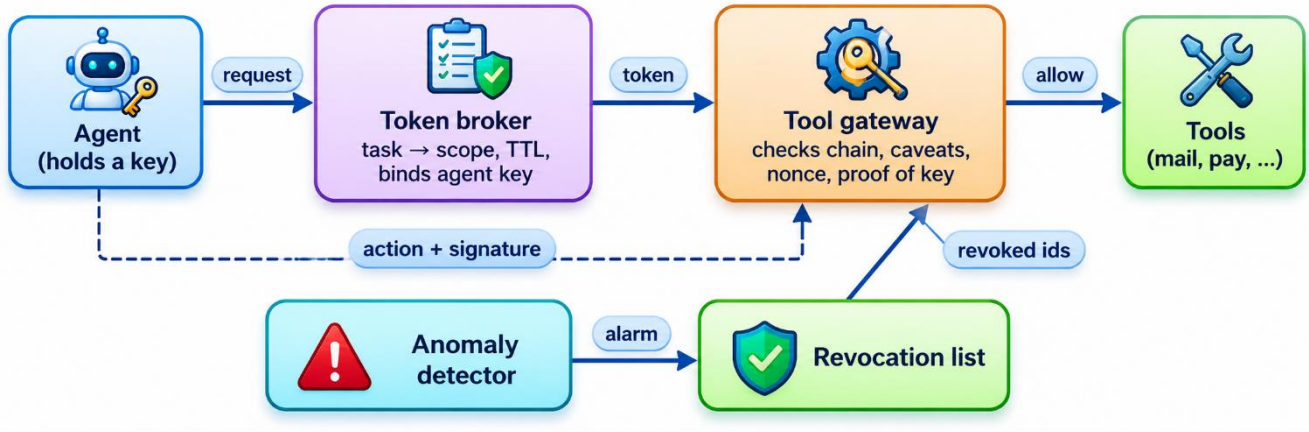


Figure. 1 The broker issues tokens scoped to a task, lifetime and the agent's key. The gateway verifies each request and rejects revoked tokens. The dashed arrow is the signature the agent attaches to each action.

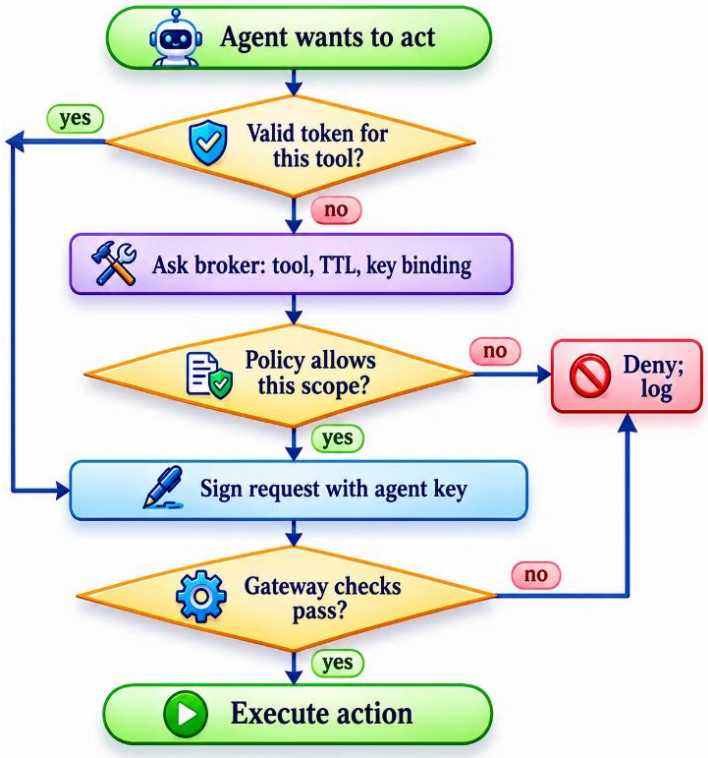


Figure 2: Request path for one action.

Jack Sparrow Publishers © 2026, IJCSER , All Rights Reserved
www.jacksparrowpublishers.com

ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/ , Vol. 3, Issue 4 , 2026 , https://doi.org/10.63328/IJCSER-V3RI4P3

The proposition shows that this is a lower bound on the optimal lifetime. The bound is tight when d is large against T, and loose when detection is fast, since quick detection makes loss flatten while token requests keep costing.

Proposition 4 (Per-action and bound tokens). A one-time token bound to action x can be used by a thief only for x, and only if the thief obtains it before the agent uses it. If such a token exists for a fraction ϕu of the time, expected loss per theft is ϕu E[v], where v is the value of an action. A token bound to the agent's key yields loss equal to pk times the loss of the same bearer token. Both follow from the definitions. They show why these designs help so much, and also what they need: a request to the broker per action, and a key that is kept separately from the token.

Proposition 5 (Requests and outages). Let a task run for τ minutes with a fresh token of lifetime T at its start and refresh before expiry. It needs $n_{\text{requests}} = 1 + \left\lceil \frac{(\tau - T)_+}{T} \right\rceil$ token requests. If each request fails because of a broker outage with probability pout, the task fails with probability $P(\text{task fails}) = 1 - (1 - p_{\text{out}})^n \approx np_{\text{out}}$ for n requests. This is elementary, and its use is to show that outages translate into task failures in proportion to the number of requests.

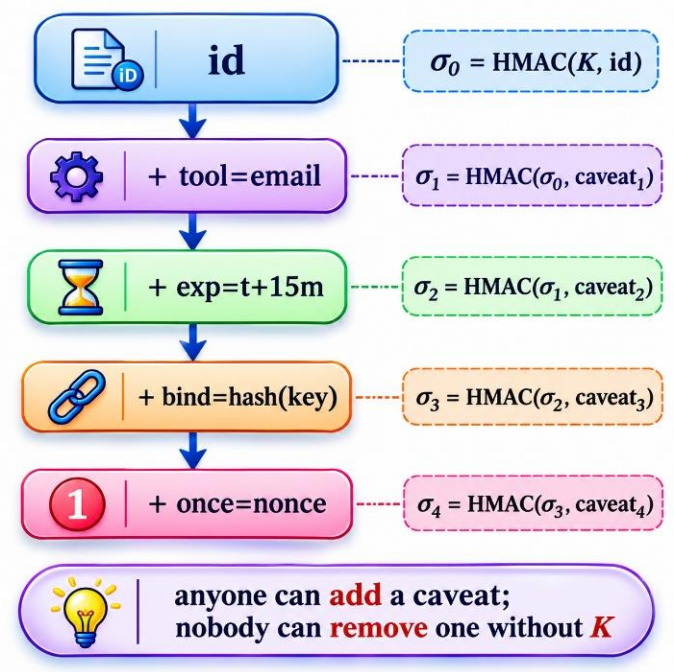


Figure. 3 Token structure. Each caveat extends an HMAC chain. A holder can add a caveat to narrow the token, but cannot remove one without the broker's root key K.

5. Experimental Setup

Tools and values. Four tools with values (1, 5, 20, 100) per action, used by tasks with probabilities (0.45, 0.30, 0.18, 0.07). An attacker can execute actions at rates (10, 4, 2, 0.2) per minute on each tool, so the value extraction rate per tool is (10, 20, 40, 20) per minute. With all tools in scope the attacker uses them all: $\kappa = 90$ per minute. usable until $\min(R, D + \rho)$. With one tool in scope, the expected rate is $\kappa = 19.1$, a factor of 4.7 lower.

Detection. The time to detect misuse is exponential with mean $d = 20$ minutes, and detection fails altogether

$\sigma_0 = \text{HMAC}(K, \text{id})$ and $\sigma_i = \text{HMAC}(\sigma_{i-1}, \text{caveat}_i)$

with probability $p_{\text{miss}} = 0.05$. When both token and key are stolen is $p_k = 0.10$. A one-time token exists for $\phi_u = 6.7\%$ of the time (about 2 seconds in every 30).

Costs. $N = 100$ agents, theft rate $\theta = 0.02$ per agent per day, and each token request costs $c_i = 0.02$ units plus, because a request can fail, an interruption cost of 5×0.005. That gives a cost per request of 0.045. Each agent refreshes $n_s = 2$ scoped tokens continuously. Broker outages occur with probability $p_{\text{out}} = 0.005$ per request.

Monte Carlo. We simulate 400,000 thefts for each design and report means, 95th and 99th percentiles. The exposure law is checked against 200,000 simulated thefts per point. Token requests per hour are simulated from a

Jack Sparrow Publishers © 2026, IJCSER , All Rights Reserved
www.jacksparrowpublishers.com

ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/ , Vol. 3, Issue 4 , 2026 , https://doi.org/10.63328/IJCSER-V3RI4P3

stream of one action every 30 seconds with each action using a random tool. Task durations for the outage analysis are log-normal with median 4 minutes.

Prototype. In Python 3 with hmac, hashlib and the cryptography package (Ed25519). Timings are on one core of the sandbox machine without tuning.

Table. 1 Loss per theft (units; 400,000 simulated thefts) and token requests per agent-hour. ± is the 95% CI half-width.

Design	Mean	95th	99th	Req./h
Static key	294,174.6 ± 4,593.7	15,324	9,371,551	0.0
Session token, 8 h	2,732.1 ± 16.8	8,034	34,692	0.1
Scoped, 60 min	276.3 ± 1.0	929	1,623	4.0
Scoped, 15 min	114.6 ± 0.3	347	537	14.9
Scoped, 5 min	44.1 ± 0.1	134	186	34.7
Scoped, 1 min	9.4 ± 0.0	28	38	91.3
Scoped 15 min, bound	11.4 ± 0.2	82	262	14.9
Per-action, one-time	0.8 ± 0.0	1	20	120.0

6. Results and Analysis

6.1. Loss By Credential Design

Table 1 and Figure 4 give loss per theft and the load on the broker. The differences are very large. A static key has an expected loss of 294,175 units per theft, about 108 times the session token's 2,732. The tail is worse: the 99th percentile for a static key is 9,371,551 units, because a theft that is never detected can run until the next rotation. Narrowing the scope and shortening the lifetime cut the loss by a factor of 24 from the session token to the 15-minute scoped token (115 units), which is a combination of the scope effect (a factor of 4.7) and the lifetime effect (the exposure E(480) = 30.2 minutes against E(15) = 6.0, a factor of 5.0). Shortening to 5 minutes gives 44 and to 1 minute 9.4. Binding the token to the agent's key cuts it by another factor of 10 to 11.4, and a one-time token reaches 0.8, 138 times lower than the 15-minute scoped token. The price is paid in token requests. A 15-minute scoped token needs about 15 requests per agent-hour, a 1-minute token about 91, and a per-action token 120 (one for every action). The session token needs one request every eight hours.

6.2. The Exposure Law

Figure 5a compares Eq. (1) with simulation for perfect detection and detection delays of 5, 20 and 60 minutes. The largest relative error over all settings (including pmiss = 0.05) is 0.6%, consistent with Monte-Carlo noise. For lifetimes much shorter than d the curves follow T/2, and for long lifetimes they flatten at d if detection is certain. The second panel shows where the loss comes from. For the static key, 99.4% of the expected loss comes from the 5% of thefts that are never detected: about 291,514 units, against only 1,711 from the thefts that are detected. For the 8-hour session token the share is 39%, and for the 15-minute scoped token only 6%. This is the central quantitative point of the paper. Fast detection helps all designs, but it cannot be perfect, and expiry is the only mechanism whose effect does not depend on detection. Lifetime caps the loss from the cases where detection fails.

6.3. What lifetime to choose

Figure 6a shows the daily cost of continuous refresh (Proposition 3). With a = 38.2 and b = 12, 960 the lower bound of Eq. (2) is T0 = 26 minutes, while the numerical optimum is $T^* = 51$ minutes, about twice as large, at a cost of 766 per day against 995 predicted by the shortcut. The shortcut gets both the lifetime and the cost wrong, because d = 20 minutes is comparable to the lifetimes in play and detection limits the loss. The optimum is broad. The cost is 2,617 per day at 5 minutes, 1,108 at 15, 766 at the optimum, 807 at 100 and 1,172 at 480 minutes. A 15-minute token is only 45% above the optimum, and 100 minutes 5%. Where the optimum sits depends on the inputs (Figure 6b). It falls from 116 to 27 minutes as detection gets slower (mean delay from 2 to 180 minutes). It rises from 10 to 274 minutes as the cost of a token request rises from 0.005 to 0.3. It also falls as the chance of missed detection rises, from 67 minutes with perfect detection to 39 minutes when 20% of thefts are missed.

6.4. Tasks, outages and detection delay

When each task gets its own token, the number of requests depends on how long tasks run compared with the lifetime (Proposition 5). With a median task of 4 minutes, a 1-minute lifetime needs 7.11 requests per task on average and fails 3.42% of tasks in a broker outage; a 5-minute lifetime needs 1.86 and fails 0.92%; a 15-minute lifetime needs 1.13 and fails 0.57%, and the failure rate bottoms out at 0.5% (the outage rate of a single request) once lifetimes exceed most tasks (Figure 7). Only 9% of tasks run longer than 15 minutes. A lifetime that covers the bulk of the task distribution removes the extra failures, so for task-scoped tokens the lower end of the lifetime is set by task length and not by security. Detection speed matters most for long-lived credentials. As the mean detection delay goes from 2 to 180 minutes, the expected loss of the session token rises from 1,241 to 11,070 units, whereas the 15-minute scoped token goes from 38 to 139 and the 1-minute token barely moves. The static key's loss is

Jack Sparrow Publishers © 2026, IJCSER , All Rights Reserved
www.jacksparrowpublishers.com

ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/ , Vol. 3, Issue 4 , 2026 , https://doi.org/10.63328/IJCSER-V3RI4P3

almost insensitive to detection delay (300,553 to 311,587) because it is dominated by the undetected 5%.

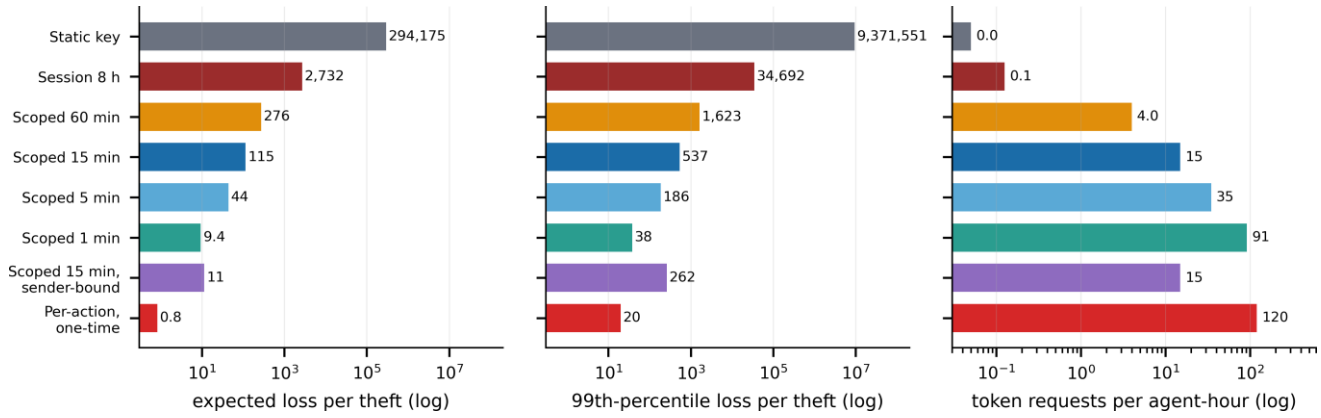


Figure 4: Left: expected loss per theft. Middle: 99th-percentile loss. Right: token requests per agent-hour (all log scale). Shorter, narrower and more tightly bound tokens cut loss by orders of magnitude and raise the request load.

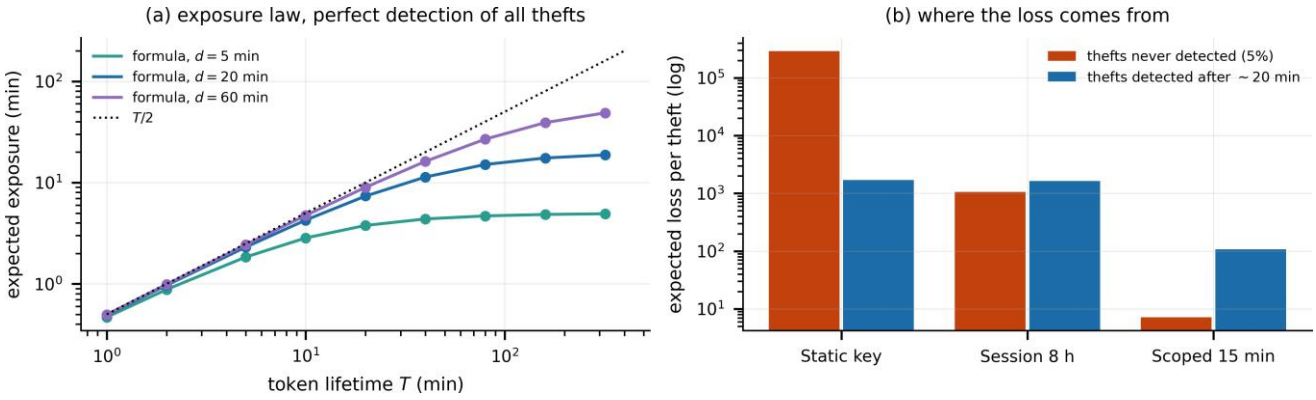


Figure 5: (a) Expected exposure against lifetime: formula (lines) and simulation (points), for different detection delays with perfect detection. (b) Expected loss per theft split into thefts that are never detected and thefts that are detected after about 20 minutes.

6.5.Revocation Delay

A revocation list has to reach every gateway, and the time it takes adds to the detection delay: a token is usable until min(R, $D + \rho$), where ρ is the propagation delay. We varied ρ from 0 to 90 minutes (Figure 8). Long-lived credentials are sensitive to it.

The session token's expected loss per theft rises from 2,721 to 9,387 units, a factor of 3.4. The 60-minute scoped token rises by 2.1 times. The 15-minute scoped token levels off at about 143 units (a factor of 1.2), because by the time a slow revocation arrives the token has usually expired, and the 1-minute token does not change at all.

Bound tokens follow the same shape at a tenth of the level. So fast revocation matters for long-lived tokens, and short lifetimes make the system tolerant of a slow revocation channel, which is useful when gateways are many and spread out. At a delay of 10 minutes, the session token loses 3,538 units per theft and the 15-minute scoped token 142.

7. Prototype and Security Tests

7.1.Costs

The prototype mints tokens as HMAC-SHA256 chains over caveats, lets a holder attenuate a token by adding a caveat without the root key, and verifies by recomputing the chain and checking each caveat.

Minting a token with no caveats takes 3.7 μs and verifying a token with two caveats 10 μs, about 95,919 verifications per second per core. Cost grows linearly with caveats, reaching 60 μs for 16 caveats (Figure 9a), and tokens are small: 57 bytes with two caveats and 119 with 16. Attenuating a token costs about 3.8 μs regardless of length, since it extends the chain by one step.

Proof of possession is much more expensive. The agent signs each request with Ed25519, which took 47 μs, and the gateway verifies it, which took 143 μs. A full bound-request check took 167 μs, about 5,974 per second per core. This is still negligible beside the network call to the tool, but it is the dominant cost in the design.

Jack Sparrow Publishers © 2026, IJCSER , All Rights Reserved
www.jacksparrowpublishers.com

ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/ , Vol. 3, Issue 4 , 2026 , https://doi.org/10.63328/IJCSER-V3RI4P3

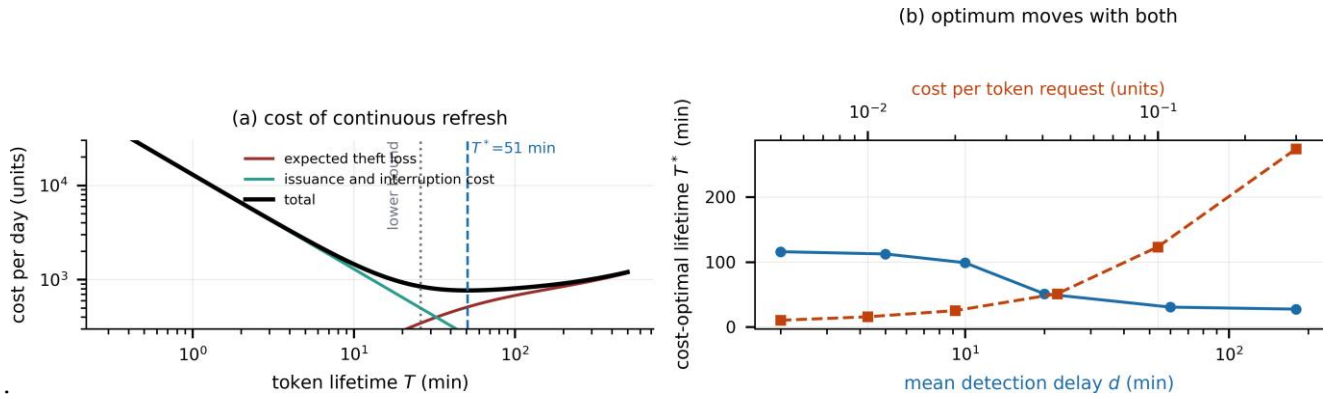


Figure 6: (a) Daily cost against token lifetime: expected theft loss, issuance and interruption cost, and their sum. The dotted line is the lower bound of Proposition 3. (b) The cost-optimal lifetime against the mean detection delay (bottom axis) and the cost per token request (top axis).

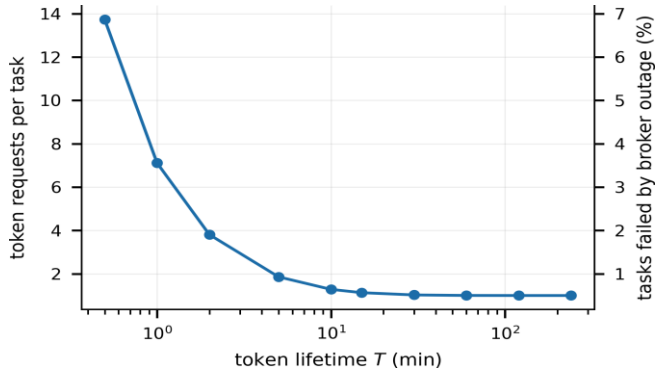


Figure 7: Token requests per task and the resulting failure rate from broker outages (0.5% per request), against token lifetime for tasks with median duration 4 minutes.

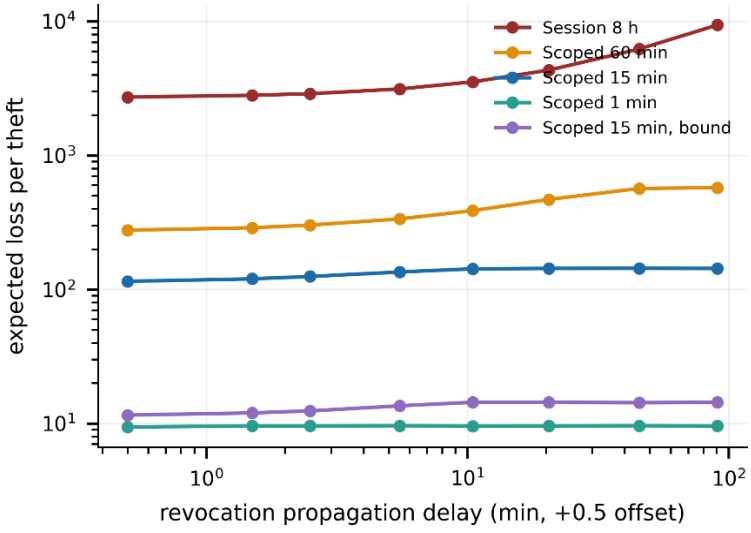


Figure 8: Expected loss per theft against revocation propagation delay. The curves for short-lived tokens flatten because expiry arrives before the revocation does.

7.2.Security Tests

Table 2 lists 18 tests, each run 1,000 times with fresh values. The prototype accepted exactly the requests it should and rejected the rest in every trial. Three results deserve a comment. A token whose caveat is replaced to widen its scope, or whose caveat is removed to lift a spend limit, fails because the signature chain no longer verifies. A holder who adds a caveat to "widen" a token only adds a restriction, and the original limit still applies. And a one-time token is accepted once and rejected on replay or on a different

Jack Sparrow Publishers © 2026, IJCSER , All Rights Reserved
www.jacksparrowpublishers.com

ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/ , Vol. 3, Issue 4 , 2026 , https://doi.org/10.63328/IJCSER-V3RI4P3

action. The row "bearer token stolen, used within lifetime" is accepted by design, and it is the limitation that motivates everything else in this paper. A bearer token works for whoever holds it until it expires or is revoked. Binding it to the agent's key removes that: the stolen bound token is rejected both with no signature and with the attacker's own key, and a signature captured for one action cannot be reused for another.

7.3.Discussion and Limitations

Simulated losses. The loss figures follow from assumed tool values, attack rates, detection delays and theft rates. They show the relative effect of the designs and the shape of the trade-offs, not the dollar value of a real incident. The huge static-key figure is dominated by the 5% undetected thefts and a 90-day rotation, and it moves proportionally with both.

Independence of theft and activity. We assumed theft occurs at a uniformly random moment of a token's life. In practice, theft may correlate with task starts or with the moment a token is placed in a log, which could make exposure longer or shorter than the formula says.

Revocation. The main experiments treat revocation as instantaneous once misuse is detected. Section 6.5 the mediation and review mechanisms discussed earlier in this series are needed. Scoping helps here too, since an agent that holds only a mail token cannot spend money.

Prototype scope. The tokens are HMAC chains verified by a party that holds the root key, so the gateway can also mint tokens. A deployment that needs verification without that power would use asymmetric signatures, at the costs shown in Figure 9. We tested a single process, and not network transport, clock skew or concurrency.

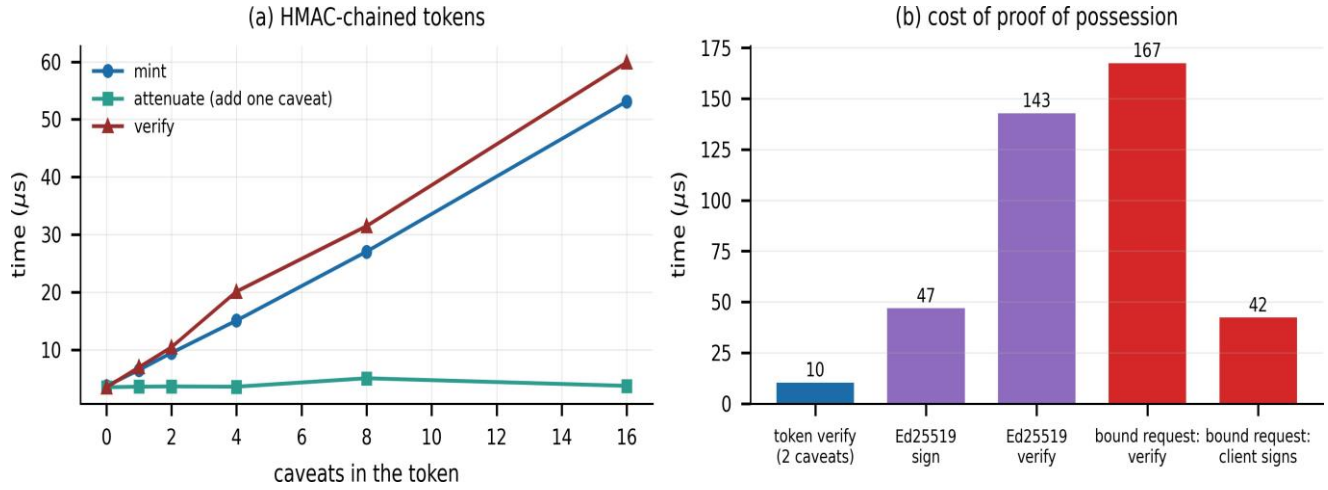


Figure 9: (a) Cost of minting, attenuating and verifying HMAC-chained tokens against number of caveats. (b) Cost of the signature operations needed for proof of possession.

Table 2: Security tests on the prototype (1,000 trials each). The right column is the share of trials in which the prototype behaved as expected.

Test	Expected Behaved	Test
valid request in scope	Accepted	100%
wrong tool	Rejected	100%
over spend limit	Rejected	100%
expired token	Rejected	100%
caveat replaced to widen scope	Rejected	100%
caveat removed to lift spend limit	Rejected	100%
holder narrows token, stays inside	Accepted	100%
forged signature	Rejected	100%
holder 'widens' by adding caveat	Accepted	100%
one-time token, first use	Rejected	100%
one-time token, different action	Rejected	100%
bound token, owner with key	Accepted	100%
bound token stolen, no key	Rejected	100%

Jack Sparrow Publishers © 2026, IJCSER , All Rights Reserved
www.jacksparrowpublishers.com

ISSN: 3107 - 8605 (Online) , http://www.ijcser.com/ , Vol. 3, Issue 4 , 2026 , https://doi.org/10.63328/IJCSER-V3RI4P3

8. Conclusion

For an AI assistant, the credential it carries sets the cost of a leak. In simulation, a static key had an expected loss per theft of about 294,175 units, a session token 2,732, a 15-minute scoped token 115, the same token bound to the agent's key 11.4 and a one-time token 0.8. Most of the static key's loss was from thefts that detection never caught, and that is the reason to prefer expiry: it limits the damage even when detection fails. Exposure has a closed form, and the textbook formula for the best lifetime is only a lower bound, off by a factor of about two in our setting. A prototype of attenable tokens with proof of possession passed all 18 security tests and cost tens to a few hundred microseconds per request, and the one test that a thief passes is the bearer token used within relaxes that.

Broker availability: The broker becomes a critical dependency. Outage rate and retry behaviour dominate failures for the shortest lifetimes, and a deployment would add caching, redundancy and grace periods, each of which weakens the exposure bound. Our outage model is simple.

Key theft: Proof of possession protects the token only as long as the agent's key stays separate from it. If the key is in the same process, in the same logs or in the same prompt, pk will be close to one. Hardware or process isolation for the key matters as much as the token design.

What tokens cannot do: They limit what a thief can do with a credential, not what a prompt-injected agent does with its own legitimate credential. For that, its lifetime. The next steps are to test the design against real agent traffic and real outages, to study delegation chains in which sub-agents receive narrower tokens, and to study how well agents keep their keys apart from their tokens.

References

[1]. J. H. Saltzer, M. D. Schroeder. The protection of information in computer systems. Proc. IEEE, 63(9):1278–1308, 1975.
[2]. J. B. Dennis, E. C. Van Horn. Programming semantics for multi-programmed computations. Communications of the ACM, 9(3):143–155, 1966.
[3]. S. Rose, O. Borchert, S. Mitchell, S. Connelly. Zero Trust Architecture. NIST Special Publication 800-207, 2020.
[4]. D. Hardt (ed.). The OAuth 2.0 Authorization Framework. RFC 6749, 2012.
[5]. M. Jones, J. Bradley, N. Sakimura. JSON Web Token (JWT). RFC 7519, 2015.
[6]. D. Fett, B. Campbell, J. Bradley, T. Lodderstedt, M. Jones, D. Waite. OAuth 2.0 Demonstrating Proof of Possession (DPoP). RFC 9449, 2023.
[7]. Birgisson, J. G. Politz, U´. Erlingsson, A. Taly, M. Vrable, M. Lentczner. Macaroons: cookies with contextual caveats for decentralized authorization in the cloud. In NDSS, 2014.
[8]. K. Greshake et al. Not what you've signed up for: compromising real-world LLM-integrated applications with indirect prompt injection. arXiv:2302.12173, 2023.

Parameters

Tool values (1, 5, 20, 100); attack rates (10, 4, 2, 0.2) per minute; task mix (0.45, 0.30, 0.18, 0.07). Detection mean 20 min, miss probability 0.05. Static key rotation 90 days; session token 8 h. pk = 0.10; ϕu = 0.067. N = 100 agents, θ = 0.02 thefts per agent per day, ns = 2, request cost 0.02 plus interruption 5 × 0.005. Outage probability 0.005 per request. Task duration log-normal, median 4 minutes, log-sd 1. Monte Carlo: 400,000 thefts per design, 200,000 per exposure-law point. Prototype timings: 20,000 repetitions per size, 5,000 for signatures. Code and figures are in the supplementary archive. All figures were generated from code at 600 dpi and printed at their stored width or smaller.

Declaration

Conflicts of Interest: The authors declare no conflict of interest.

Author Contribution: All authors wrote the main manuscript text and also consent to the submission.

Ethical approval: Not applicable.

Consent to Participate: All authors consent to participate.

Funding: Not applicable, and No funding was received

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Personal Statement: We declare with our best of knowledge that this research work is purely Original Work and No third party material used in this article drafting. If any such kind material found in further online publication, we are responsible only for any judicial and copyright issues.

Generative AI Statement: The authors declare that generative AI was not used in the preparation or creation of this manuscript. The alternative text (alt text) accompanying the figures in this article was generated by Frontiers with the assistance of artificial intelligence. Reasonable efforts have been made to ensure the accuracy and appropriateness of the generated alt text, including review by the authors wherever possible. If you identify any inaccuracies or issues, please contact the editorial team.

Publisher's Note: The statements, opinions, and claims presented in this article are exclusively those of the authors and do not necessarily reflect the views of their affiliated institutions, the publisher, editors, or reviewers. Any products discussed or evaluated, as well as any claims made by their manufacturers, are not warranted, approved, or endorsed by the publisher.

Acknowledgements

We thank everyone who inspired our work

Jack Sparrow Publishers © 2026, IJCSER , All Rights Reserved
www.jacksparrowpublishers.com