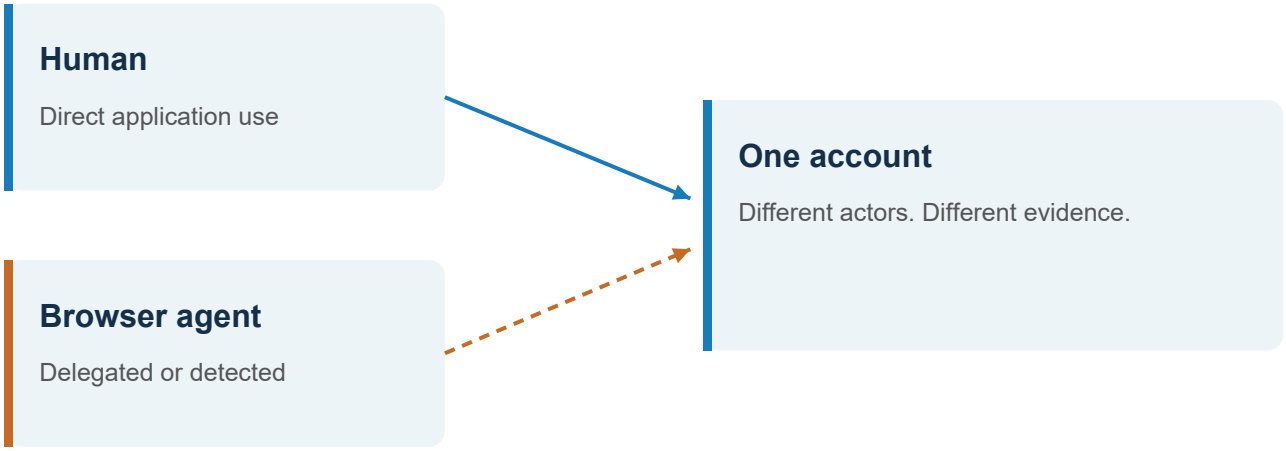
SolvxAI

RESEARCH / WHITE PAPER 09

Who Is Acting in Your Session?

Browser agents, authenticated access, and accountability



Authentication identifies the account. Accountability also needs the actor and authority.

EVIDENCE / CONTROLS / LIMITS

September 2026 · Edition 1.0 · SolvxAI Research

Executive summary

An authenticated session establishes which account a request belongs to. It does not, by itself, establish whether a person clicked a button, an authorized agent executed an instruction, or compromised software acted through that person's browser. As browser agents become capable of navigating applications and submitting forms, this distinction becomes an application-governance concern.

Published evidence establishes that the risk is practical. Researchers have redirected browser agents through malicious page content, demonstrated account-takeover paths using existing authenticated sessions, and exposed weaknesses in the tools that constrain automated browsing. Separately, a government security advisory documented compromised browser extensions capable of stealing session information. These evidence categories establish different things and should be read separately. [1, 2, 3, 4]

SolvxAI's response combines an administrator-controlled external-agent guard, session-level reporting of detected automation, and a distinct authorization and control path for its own browser agent. With enforcement enabled, the guard blocks supported automation signals on application surfaces. With enforcement disabled, passive observation can still associate detected automation with the authenticated account and its recorded application activity. SolvxAI's own browser sessions use explicit grants and control transitions that support more specific attribution. [11]

The resulting benefit is a clearer account of access and activity: which account was used, whether automation was observed, what the application recorded, and, for the governed internal agent, how control was assigned and changed. This evidence supports review and incident response. It does not establish the human's intent, make an external provider's identity independently verifiable, or turn all browser activity into a complete forensic recording.

The central conclusion is that access control and accountability should be designed together. Blocking detected external automation reduces one route into an authenticated application. Monitoring preserves useful evidence when administrators intentionally allow that route. Explicit agent authorization gives stronger control where the application can govern the execution protocol. None of these measures eliminates the need for ordinary application authorization, endpoint security, or engineering review.

Scope and evidence standard

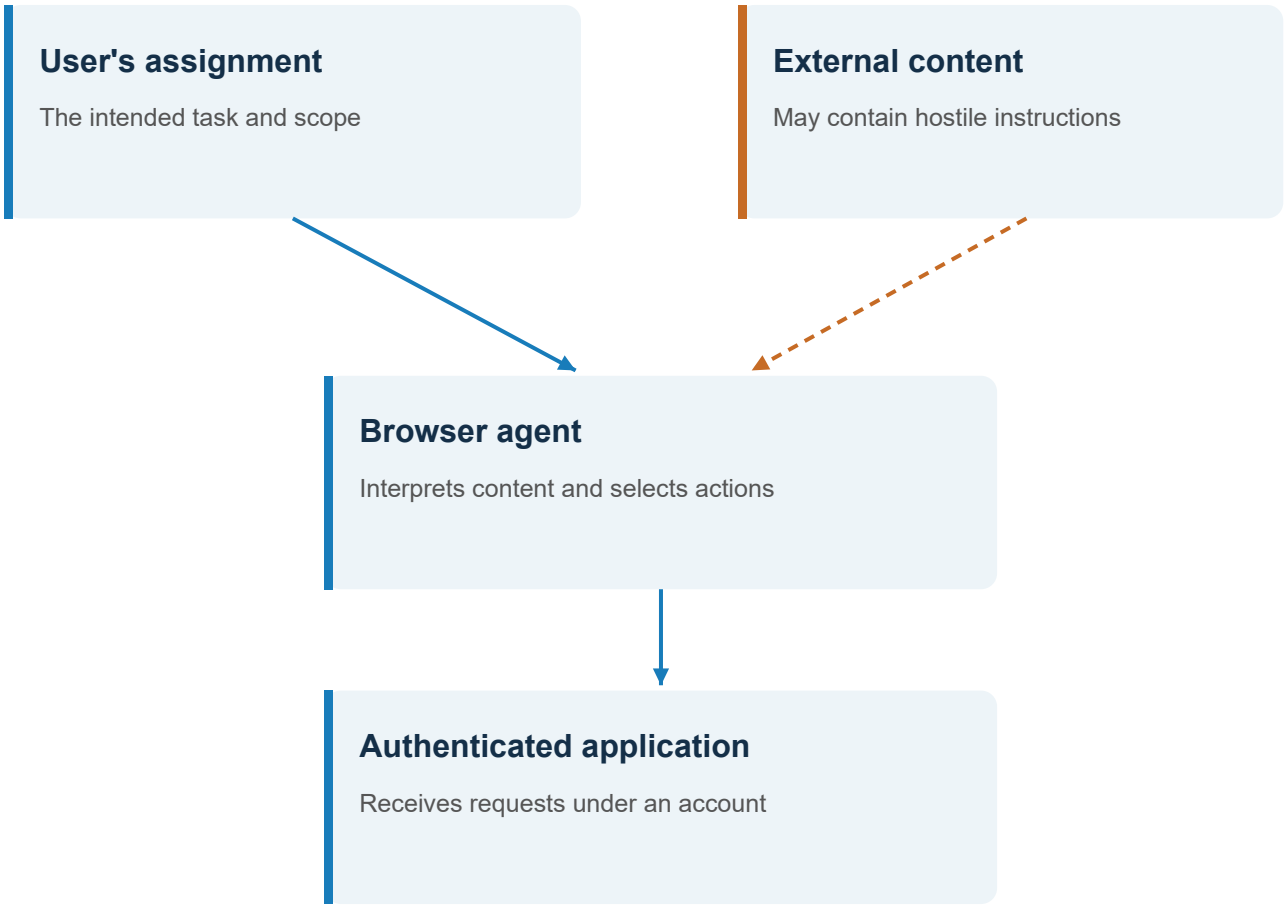
This is a SolvxAI-authored evidence review and implementation assessment, edition 1.0, with a research cutoff of 15 September 2026. It covers authenticated browser use and the associated application logs. It does not report a penetration test of SolvxAI, a comparative benchmark of current agent products, or a measured reduction in customer incidents. Product descriptions concern the implementation reviewed for this edition; availability and effective policy depend on the deployed release and administrator configuration. [11]

Sources are identified as government advisories, maintainer disclosures, vendor research, a research preprint, or professional security guidance. Public demonstrations are not presented as evidence of customer losses. Historical vulnerabilities are dated and patched versions are stated where the source provides them. The report's engineering scenarios and control recommendations are analytical judgments, not observed incidents at SolvxAI.

FIGURE 01 / CONCEPTUAL MODEL

One session. Competing instructions.

An authenticated request does not establish who supplied the instruction.



The governance question

Was this action within the user's delegated authority, or did another source redirect the agent? A valid session alone cannot answer that question.

Synthesis of the cited demonstrations [1, 2, 5]. Not an incident at SolvxAI.

1. The authenticated-browser threat model

Browser automation can operate at several layers. An extension may interact with page content; an agent may navigate through browser-control tools; local software may issue mouse and keyboard input. The application can receive requests through an already authenticated browser while the immediate operator changes. Access to the application therefore need not involve a new login or a defeated authentication challenge.

There are three distinct routes to concern. First, a user may intentionally delegate a task to an external agent that lacks application-specific constraints. Second, a legitimate agent may encounter attacker-controlled content and follow instructions inconsistent with its assignment. Third, a browser extension or endpoint may be compromised, exposing authenticated sessions to someone the user never authorized. Only the third necessarily involves compromised software; the second can exploit how a legitimate agent interprets content. The cited demonstrations and extension advisory illustrate these different routes. [1, 2, 4, 5]

Four identities that should remain distinguishable

Identity or role	Question it answers	Evidence needed
Account principal	Which authenticated account supplied access?	Authentication and session records
Operating actor	What person or software performed the interaction?	Control protocol or qualified observation
Delegating authority	Who authorized the agent and with what limits?	Grant, scope, consent, and policy state
Content originator	Who supplied material that influenced the agent?	Source provenance and task context

These roles can diverge. A user may authorize a comparison of two datasets while a malicious instruction inside a retrieved document asks the agent to export unrelated information. The authenticated principal remains the user throughout. The document author supplied the unwanted instruction. A session log containing only the user's name cannot resolve that difference.

For an engineering application, the assets at risk include confidential well data, imported datasets, model inputs, selected calculation assumptions, saved results, and decisions derived from those results. An incorrect saved input can matter even if no information leaves the organization. These are plausible consequences of operating the interface with the user's permissions; this report does not assert that such an incident has occurred in SolvxAI.

A boundary with practical limits

The external guard observes characteristics available at the application boundary. It cannot directly inspect every external agent's instruction history, know the source of every mouse event, or attest the integrity of the computer on which the browser runs. An attacker with control of that endpoint can also influence browser-reported evidence. Consequently, the security question is how to reduce exposure and improve evidence under imperfect visibility, while keeping authorization enforced in the application itself. [11]

2. Documented browser-agent demonstrations

Comet: authenticated sessions used across sites

Brave's security researchers reported a Comet proof of concept on 20 August 2025. A request to summarize a Reddit page caused the agent to follow instructions embedded in a comment, obtain account information, read a one-time login code from an already authenticated Gmail session, and disclose the information through a Reddit reply. The demonstrated consequence was a path to account takeover. [1]

This was a researcher-controlled demonstration, not a published count of victim accounts. Brave documented an initial fix, an incomplete retest, and further disclosure; its article includes an update that the attack class had not been fully mitigated at that time. This report makes no claim that the same exploit works against the current Comet release. [1]

The relevant lesson for application owners is that an agent can use legitimate session access for an action the user did not request. SolvxAI's guard addresses the entry of detectable external automation into its own application. The example does not prove that the guard would have stopped this specific Comet exploit, nor that it protects the user's email or other websites.

Scamlexity: purchases, phishing, and misleading instructions

Guardio Labs published its Scamlexity research on 20 August 2025. Its Comet tests included a fake shopping site, a banking-phishing flow, and instructions hidden behind a simulated CAPTCHA. In some shopping trials the agent supplied saved personal and payment details; other trials refused or required human involvement. The CAPTCHA demonstration downloaded a harmless file. It did not establish successful malware execution. [2]

The banking test used a phishing page observed in the wild, but the agent interaction remained a researcher-run test. These distinctions matter: an active phishing site is not evidence that an actual bank customer lost money through that particular agent. The research demonstrates inconsistent security decisions in the tested conditions, without estimating population-wide failure rates. [2]

For SolvxAI, the analogous concern is the transition from reading a page to taking a consequential application action. Session logging makes observed activity easier to investigate, while controlled internal-agent permissions can constrain whether interaction is permitted. External-agent monitoring alone does not introduce a new confirmation step before every save, export, or calculation.

Interpretation

Together, these demonstrations support treating browser agents as software with consequential authority, rather than assuming every authenticated interaction reflects contemporaneous human judgment. They do not support treating all agent use as hostile. A useful application policy must accommodate authorized automation, preserve uncertainty in attribution, and avoid making stronger prevention claims than the detector can demonstrate.

3. Tool vulnerabilities and extension compromise

Browser Use: a confirmed, versioned security defect

The Browser Use maintainer advisory GHSA-x39x-9qw5-ghrf, published 4 May 2025 and assigned CVE-2025-47241, describes a bypass of its allowed-domain check. The advisory identifies versions up to 0.1.44 as affected and 0.1.45 as patched. Its stated impact includes unauthorized browsing and potential access to local or internal services. This is a specific software defect with a documented remediation, not a claim that all present-day Browser Use installations remain vulnerable. [3]

The associated research preprint, The Hidden Dangers of Browsing AI Agents, was submitted on 19 May 2025. Its authors describe a broader examination of agent behavior, prompt injection, domain validation, and credential exposure. The preprint and the maintainer advisory overlap in evidence and should not be counted as independent demonstrations of the same flaw. The preprint's findings also should not all be attributed to the narrow CVE. [6]

For an application owner, the lesson is about dependency and delegation risk. An external automation framework's promised restrictions may themselves contain defects. SolvxAI can restrict supported automated access to its application, but it does not patch another vendor's URL parser or prevent that tool from visiting unrelated systems.

Compromised Chrome extensions: an actual campaign

On 30 December 2024, Singapore's Cyber Security Agency published an advisory about an ongoing Chrome-extension compromise campaign. It listed extensions confirmed to contain malicious code and explained that stolen authenticated sessions and cookies could enable impersonation without the victim's username or password. The list included Cyberhaven's security extension and several AI-branded extensions. The advisory also stated that no reports had been observed locally at that time. [4]

This is evidence of actual malicious extension distribution. It is separate from prompt-injection research and does not establish that the named model providers authored or distributed the malicious extensions. An extension name containing ChatGPT or Gemini is not proof of official provider involvement. [4]

The implication for SolvxAI is limited but important. A browser session is not sufficient evidence of the operator's identity or intent. However, an automation detector is not an anti-malware product: stolen sessions, direct authenticated requests, and compromised endpoints may produce no supported automation signal. Extension governance, endpoint protection, and session revocation remain separate responsibilities.

4. Later evidence and provider defenses

Cloud and local execution

Brave's 8 June 2026 research described a Tabstack agent redirected from summarization into submitting conversation context to an external form. The researchers reported the issue on 13 May and independently verified a fix on 1 June. The article also described manipulated suggestions in Cotypist, an on-device autocomplete tool. Cotypist required a human keystroke to accept its output and did not have the same autonomous action capability. [5]

The reported distinction is important: running a model locally does not automatically remove prompt-injection risk, but the consequences depend on its powers. An autocomplete suggestion and an autonomous form submission are materially different outcomes. This report does not extrapolate either demonstration into a claim about every cloud or local agent. [5]

Anthropic's browser-use evaluation

Anthropic's 24 November 2025 account of Claude browser-use defenses described stronger model training, classifiers, and adversarial testing, while acknowledging residual risk. Its reported evaluation used an adaptive attacker with up to 100 attempts per environment. The reported success rate belongs to that evaluation setup; it is not a forecast of the chance that any ordinary browser session will be compromised. [7]

The useful conclusion is that provider defenses are substantive and evolving, while application owners still need their own boundaries. SolvxAI's argument does not depend on asserting that providers have no safeguards or that their products are intrinsically malicious.

OpenAI's treatment of outbound links

OpenAI describes URL-based data exfiltration as a risk: a URL can carry sensitive information to the destination even when the agent never displays that information in its response. Its published approach checks whether a requested URL was independently observed on the public web, with additional handling for unverified links. OpenAI explicitly limits the claim: this protection does not make every page trustworthy or all browsing safe. [8]

This illustrates why an application's session controls and a provider's model or network controls address different layers. SolvxAI's guard governs detectable access to SolvxAI. It does not inspect every outbound connection made by external software or provide a general-purpose data-loss-prevention boundary.

Security guidance

OWASP's Excessive Agency guidance connects harmful actions to excessive functionality, permissions, or autonomy, and recommends minimum necessary privileges, user approval for consequential actions, and authorization enforced outside the model. It identifies logging and monitoring as measures that can limit damage rather than prevent excessive agency by themselves. [9] This distinction underlies the assessment of SolvxAI's enforcement and monitoring modes in the following sections.

5. Accountability in engineering work

The practical question after an unexpected change is rarely just who was logged in. A reviewer needs to establish the affected dataset, the operation attempted, the authority under which it ran, and the evidence connecting the operation to a human or an agent. In reservoir and production engineering, this may determine whether a saved result can be reproduced and whether subsequent decisions used the intended inputs.

Consider an illustrative workflow: an engineer opens a well, an external agent begins interacting with the tab, a calculation is run, and a result is saved. If only the account name is retained, a later reviewer may incorrectly conclude that the engineer personally selected every parameter. If automation is detected only after the calculation, the evidence is weaker still. The correct report preserves the detection time and the recorded calculation event without inventing an earlier takeover time.

Three levels of attribution

Evidence level	Defensible statement	Statement that exceeds the evidence
Authenticated account	This activity used the recorded account's session.	The account owner personally performed every action.
Observed external automation	Automation evidence was recorded in this session at this time.	The identified provider performed every event in the session.
Governed agent session	The application verified a grant and recorded control transitions.	The agent's reasoning was correct, or all activity was approved individually.

These levels are not competing labels for the same fact. They answer different questions. Account ownership can be verified while the external operator remains unknown. A grant can be valid while the result still needs engineering review. An agent badge can be useful even when it cannot resolve individual clicks.

OWASP's logging guidance recommends recording the event's time, location, actor, and action, and considers interaction identifiers and analytical confidence useful attributes. It also warns against recording credentials and raw session secrets. [10] In this report, an application security-session identifier is a correlation record; it should not be confused with an authentication token that grants access.

Responsibility and authorization

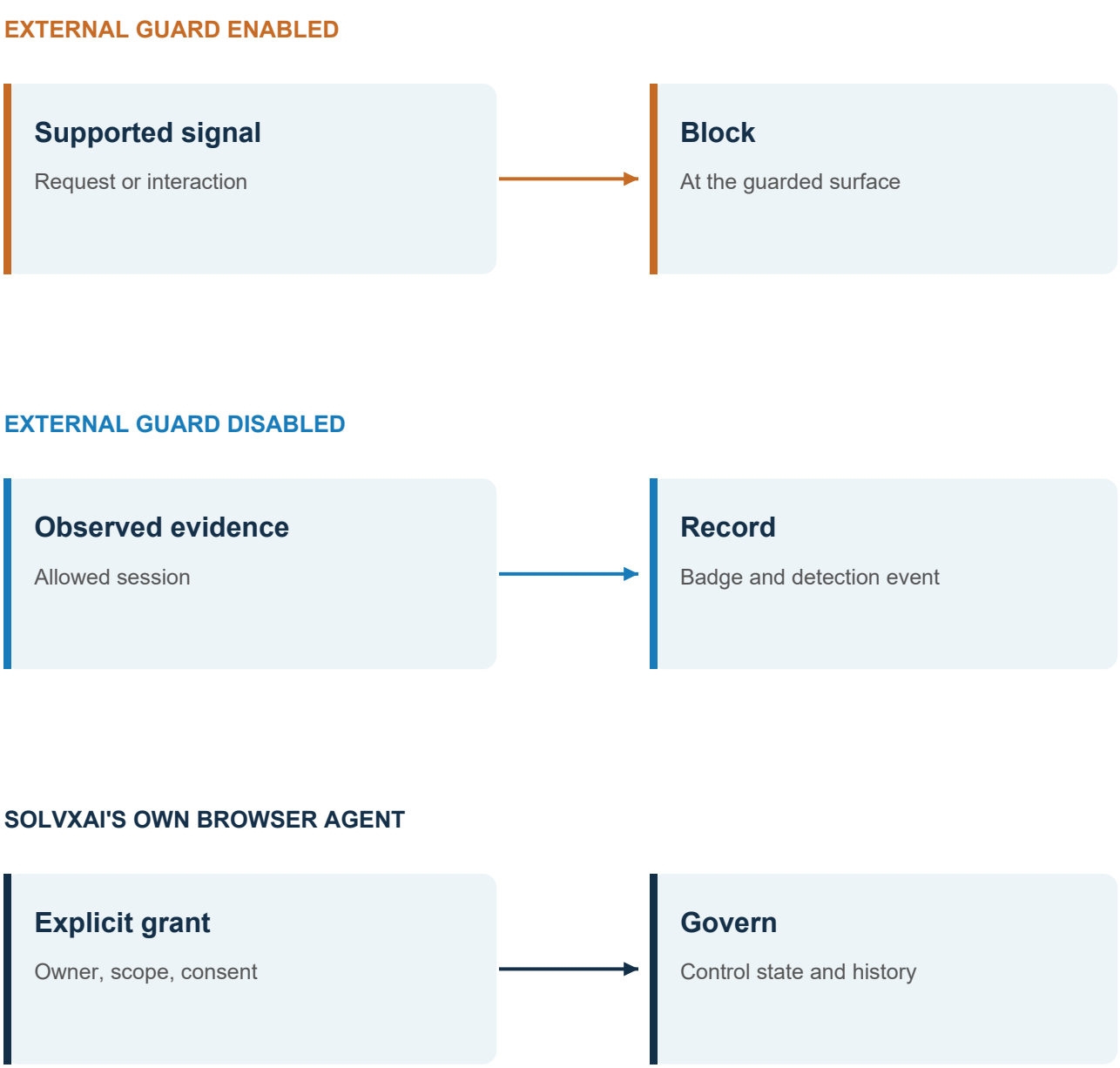
The phrase "on behalf of" should be interpreted as a connection to the account whose application access was used. For detected external automation, it is not evidence that the person expressly approved the agent, approved every action, or supplied the instruction that caused a problematic operation. Responsibility must be assessed from the broader evidence, including consent, organizational policy, observed behavior, and applicable review processes.

SolvxAI's logging design improves the available evidence by preserving the account association and detected automation context. Its internal browser-agent protocol can additionally record explicit authorization and control changes. Neither mechanism replaces a reviewer who understands the engineering and the limits of the record. [11]

FIGURE 02 / CONCEPTUAL MODEL

Three controls, three different jobs

Enforcement, observation, and governed delegation address different questions.



Disabling the guard does not enroll an external agent into SolvxAI's grant protocol. Coverage remains limited to supported signals and recorded events. [11]

6. SolvxAI's external-agent guard

The reviewed implementation has two complementary enforcement points. A server request check can deny supported automation signatures before an application page loads. A client-side gate can detect supported browser characteristics and interaction patterns after the page is available. With enforcement enabled, qualifying detections prevent access through that application surface or stop the observed interaction. Public marketing and research pages remain accessible. [11]

Observed categories include browser signatures, automation flags, and qualifying input on hidden pages. The guard's Admin Center policy is platform-wide, revisioned, and retains administrative history. It is not a per-workspace or per-user allowlist. Without a configured policy, the external guard is off. Operators requiring blocking must verify that enforcement is enabled. [11]

What the guard remedies

The immediate improvement is an application-controlled decision about supported external automation. Access does not have to be accepted merely because an external tool can reuse a valid browser session. Administrators gain a way to interrupt detected automated use and inspect associated detections, independently of whatever safeguards the outside tool provides.

This is a partial preventive control with observable scope. It can deny a supported server signature or suppress a detected client interaction; it does not mediate every direct backend API request. Absence of those signals does not establish human operation. No detection sensitivity, false-positive rate, or adversarial bypass rate is established by this report.

Availability and timing

The implementation uses cached server policy state and a bounded client wait for policy availability. On some policy-read failures it retains a prior state or allows access when no reliable enabled state is available. It therefore does not provide an unconditional fail-closed guarantee. Policy changes also have propagation time; they should not be described as an instantaneous global cutoff. [11]

If a browser action has already submitted a request, blocking later activity does not necessarily cancel server work or reverse a saved result. Administrators should verify the outcome of an interrupted calculation or write through the application. This distinguishes stopping subsequent interaction from transaction rollback.

Evidence from blocked attempts

Browser-reported blocked visits appear in the guard's administrative detection view, subject to reporting limits. A request refused by the server before the workspace loads appears in server logs and does not automatically create a full browsing-session timeline in Feedback Security. Browser reports and server denials are complementary evidence streams, with different coverage. Treating the former as a complete census of blocked traffic would be incorrect. [11]

7. External-agent session logging

Passive observation is independent of the external guard's enforcement switch. In an allowed application session, the observer can report supported automation evidence and attach it to the existing security-session model. The server requires the reporting user to own the session and have access to its organization. It validates the accepted evidence categories and derives the provider label from those categories. [11]

The Security view can consequently show the account holder, available network and device context, an external-agent badge, and a timestamped detection event alongside recorded application activity. Existing application events can retain their normal page, workspace, well, or calculation context where those events are instrumented. This is application activity logging, not a continuous screen recording, keystroke recorder, or capture of the external agent's prompts. [11]

Session creation and later detection

When supported automation is present at session initialization, the implementation avoids reusing a security-session identity inherited from an ordinary user tab. If automation is detected later, it annotates the current session and records the observation time. It does not rewrite earlier events or assert that automation started at the beginning of the visit. Repeated evidence is deduplicated, while newly observed evidence can refine the classification. [11]

These rules preserve a useful distinction between what is known and what is inferred. A late detection event establishes that evidence was recorded then. It cannot determine whether an external agent operated the tab a minute earlier, or whether the user subsequently resumed manual work. External software does not participate in SolvxAI's authenticated takeover and resume protocol, so its per-action control ownership remains unresolved.

Provider badges and uncertainty

The current implementation can associate a supported Claude browser signature with a Claude label. Other supported automation evidence receives a generic External agent label. There is no verified OpenAI-specific detector in this edition, so the system should not guess OpenAI, Codex, or ChatGPT from generic automation alone. Browser signatures and client reports are observations rather than authenticated provider identity. [11]

A SolvxAI browser-agent session can also receive external-automation evidence. The implementation preserves the independently verified internal-agent association and can show both badges. This is preferable to replacing a known grant with an uncertain external label, but it does not resolve which operator caused every subsequent event.

Record lifecycle

The implementation includes retries for reporting failures, but delivery is not guaranteed under persistent failure or client interference. Security session and activity records follow the existing 90-day retention logic; the blocked-visit store has separate 30-day retention. Authorized administrative deletion is possible. These records are useful operational evidence, but this edition does not claim immutable storage, cryptographic non-repudiation, or indefinite retention. [11]

FIGURE 03 / CONCEPTUAL MODEL

A detection time is not a takeover time

How to read a session that contains both activity and automation evidence.

T1 Application activity

A page view or calculation is recorded.

T2 Automation detected

Supported evidence is recorded at this time.

T3 Further activity

Session context continues; the operator may change.

T4 Review

Interpret the timeline alongside outcomes and authority.

Established

Account, recorded events, and observation time

Unresolved

Exact takeover time, intent, and every action's operator

Illustrative sequence, not customer data. External-session evidence model [11].

8. SolvxAI's governed browser agent

SolvxAI's own browser agent has an explicit application protocol. A user authorizes a browser session and objective; the grant is associated with the owner, organization, workspace, and conversation. Session identity is verified against that grant when the Security session is created. This provides a stronger basis for the internal-agent badge than an external browser characteristic. [11]

The browser-agent controls distinguish general browser access from permission to interact with the application. Interaction includes operations such as clicking, typing, selecting, and navigating, which can invoke existing application write handlers. Separate per-session authorization is required for those capabilities. Observation and scrolling can remain available without interaction permission. This control is broader than an approval of a particular database write, and should not be described as transaction-by-transaction consent. [11]

Control history

The Security timeline can include the internal-agent session start, tab connection, task assignment, requests to resume, resumed agent control, user takeover, pauses, and session completion or stop events. Recorded application activities can be related to the bridge's control state. This gives an administrator evidence about how the session was operated, beyond the account name alone. [11]

The distinction remains bounded by instrumentation. A timeline entry does not prove that every browser operation generated an audit event, that every external side effect succeeded, or that the agent's explanation is correct. Review must still examine the operation result and any applicable engineering checks.

Policy changes and stopping

Internal browser access and interaction permissions have a separate global policy from the external-agent guard. Policy revisions invalidate older grants, including after access is re-enabled. New agent actions are checked against the applicable state; an owner can stop a session. A save already submitted through the interface may still finish and require review. [11]

This separation is central to the product design. Allowing SolvxAI's internal browser workflow is a decision to use a controlled delegation mechanism. Turning off the external guard simply permits outside automation to reach the interface subject to ordinary account permissions. It does not enroll that software into the internal grant protocol or confer the same stopping and attribution guarantees.

Why internal origin is insufficient by itself

The security value comes from explicit scope, consent, server checks, and recorded transitions. It does not come from asserting that a SolvxAI-operated model is immune to manipulation. Internal agents can also make mistakes or process hostile content. Their results still require appropriate validation, and their tools should expose only the authority needed for the task. This is consistent with the least-privilege and downstream-authorization principles described by OWASP. [9]

9. Controlled use with enforcement disabled

A controlled evaluation may intentionally permit external agents while observing their use. In this mode, SolvxAI's external guard is disabled, but supported automation can still be reported to the Security session. The intended result is visibility into allowed testing: which account was involved, which session contained automation evidence, when evidence was recorded, and what application events were retained. [11]

The word "controlled" describes an operating arrangement, not an automatic consequence of switching the guard off. In the current implementation the switch is platform-wide. It does not create a restricted allowlist for a particular provider, user, organization, or evaluation. Other reachable accounts in the deployment are also subject to the changed enforcement policy. A narrowly scoped evaluation may therefore require a separate deployment or additional independently enforced restrictions.

Recommended operating conditions

The following are recommended practices for a monitored evaluation; they are not all built-in features of the guard:

- Define an evaluation owner, permitted tasks, affected accounts, and a time window before access is enabled.
- Use test data or a suitably limited account when the intended outcome does not require production authority.
- Verify that the Security view receives a fresh detection event and the application events relevant to the evaluation.
- Assign a reviewer and specify which unexpected behavior requires intervention, rather than assuming a dashboard is being watched.
- Review saved results and exports after interrupted or ambiguous operations; a missing completion event is not proof that nothing changed.
- Restore the agreed policy and preserve necessary review evidence before normal retention or administrative deletion removes it.

Monitoring and incident response

Active monitoring requires people or systems to examine the records and act. The implementation provides session evidence and administrative views. This report does not claim an automatic security operations center, complete alerting coverage, an outbound data-loss-prevention service, or a tamper-resistant record of every action. Those capabilities must be independently verified where an evaluation relies on them.

If suspicious behavior is detected, enabling the guard can block subsequent interactions that match supported signals. It is not a substitute for revoking a compromised authentication session or addressing an unsafe endpoint. External software may continue using other sessions or request paths that the detector does not recognize. The response must follow the actual source of authority, not merely the presence or absence of a badge.

Interpreting a quiet dashboard

No external-agent badge means that no supported evidence has been recorded for that session. It does not establish that the session was exclusively human. Reporting failures, uninstrumented activity, or automation without a supported signature can all leave an incomplete record. Operational decisions should preserve this uncertainty rather than converting an absence of observations into a claim of safety. [11]

10. Control coverage and remaining gaps

The following assessment maps plausible risks to the reviewed implementation. It is an analytical control mapping, not the result of replaying the published exploits against SolvxAI.

Risk or review problem	SolvxAI contribution	Remaining boundary
Detected outside automation uses an authenticated interface	External guard can deny supported requests or interactions when enabled.	Unknown or concealed automation may not be detected.
Automated activity looks like ordinary user work	Session badge and detection event add actor context.	External per-action ownership and intent remain uncertain.
Human and agent work share one tab over time	Detection time and existing history are preserved.	Earlier events cannot be retrospectively attributed.
Outside automation appears in an internal-agent tab	External evidence can coexist with the verified internal badge.	Mixed control is not fully resolved for every event.
Internal agent authority should be withdrawn	Policy changes invalidate grants; session controls stop further authorized execution.	Already submitted work can require outcome review.
An extension steals a session or endpoint is compromised	Application logs may aid investigation where requests are observed.	The guard is not endpoint protection or comprehensive token-replay detection.
An agent exports data through another service	Blocking its detectable SolvxAI access reduces one opportunity.	No general outbound-content inspection is provided by this feature.

Prevention, detection, and investigation

These are different control purposes. Enforcement attempts to stop an interaction. Detection records an observation. Investigation combines observations with application outcomes to explain an event. A stronger detection view improves investigation but does not necessarily improve prevention. Conversely, an early server denial may be effective prevention without generating a complete user activity timeline.

The control set is valuable when its purpose is stated precisely. SolvxAI can govern supported external automation at its boundary, retain evidence of observed involvement when access is allowed, and provide a dedicated authorization path for its own agent. It cannot infer every external instruction or guarantee that all unsupported automation has been excluded.

Conditions for stronger claims

Claims about universal blocking, reliable provider identification, complete action attribution, or quantified risk reduction would require additional evidence. Examples include independent adversarial testing, explicit provider authentication, broader event coverage, and production measurements with known denominators. This edition supplies none of those results and makes none of those claims.

The same discipline applies to assurance language. The design is consistent with several established security principles, but this paper is not an OWASP certification, a compliance attestation, or proof that any specific legal accountability requirement has been satisfied. Security and engineering review should assess the actual deployed system and its operating procedures.

11. Evaluation priorities and conclusion

Measuring the controls meaningfully

A future evaluation should define representative browser and agent configurations before execution, including ordinary human use and supported accessibility workflows. It should record which actions were attempted, which signals were observable, which interactions were blocked, and which evidence reached storage. A successful unit test establishes a programmed behavior under its setup; it does not measure coverage of all external automation.

Detection sensitivity should use known automated sessions as its denominator. False-positive measurement should use known human sessions, including background-tab and visibility transitions. Reporting completeness should compare events expected from a controlled workload with events actually retained. Provider-label accuracy should compare the detector's label with an independently established operator; a generic unknown classification should remain a valid outcome.

Testing should also examine policy propagation, unavailable policy reads, lost telemetry, restored tabs, inherited session storage, changes of organization, and external input during an internal-agent session. Interrupting work should include checking whether an already submitted operation completed. These proposed measurements target the limits described in this paper rather than trying to produce a single, ambiguous "agent security" score.

Priorities for further assurance

If customers require stronger guarantees, priorities include authenticated external-agent registration, server-side enforcement of agent-specific permissions, broader action and outcome correlation, monitoring of telemetry loss, and independently protected audit retention. Each would require its own design, implementation, and validation. They are future assurance opportunities, not capabilities asserted for the current external-agent detector.

The strongest long-term distinction between human and delegated work comes from an explicit protocol that carries the agent's identity, the user's authority, and the permitted scope. Browser observations remain useful for discovering automation that has not joined that protocol. The two approaches can complement each other without treating a browser signature as equivalent to an authenticated grant.

Conclusion

The published record supports taking authenticated browser-agent access seriously. It includes practical demonstrations of redirected agent behavior, a patched automation-tool vulnerability, and a documented extension compromise campaign. These establish credible failure modes; they do not establish a universal attack rate or prove that a particular detector prevents those attacks.

SolvxAI remedies a specific application-governance gap by combining configurable blocking of detected external automation with session-level evidence of observed agent involvement. Its own browser agent operates through a distinct, explicitly authorized control path. When administrators allow external-agent use for controlled evaluation, observation remains available and the authenticated account remains associated with the record. [11]

The benefit is transparent, qualified accountability: a clearer basis for understanding which authority was used, where automation was observed, and what the application recorded. The limits are equally important. An agent badge is evidence to interpret; a monitored session is an opportunity to supervise; and a guard is one preventive layer within the wider security of the application and the user's environment.

References

All external sources below were accessed on 15 September 2026. Dates identify the cited publication or edition, rather than the current vulnerability status of a product. References 1-10 are external sources; reference 11 documents SolvxAI's own implementation assessment.

1. Artem Chaikin and Shivan Kaul Sahib, Brave. Agentic Browser Security: Indirect Prompt Injection in Perplexity Comet. 20 August 2025. Vendor-authored original vulnerability research. Supports the authenticated-session account-takeover demonstration; Brave is also a browser vendor. Historical disclosure with subsequent retest updates, not a current-product benchmark.

2. Nati Tal and Shaked Chen, Guardio Labs. Scamlexity: We Put Agentic AI Browsers to the Test - They Clicked, They Paid, They Failed. 20 August 2025. Security-vendor research. Supports controlled shopping, phishing, and hidden-instruction demonstrations; outcomes varied and the demonstrated download was harmless.

3. Browser Use maintainers; discovery credited to ARIMLABS.AI researchers. GHSA-x39x-9qw5-ghrf: allowed_domains can be bypassed by putting a decoy domain in http auth username portion of a URL. 4 May 2025. Maintainer security advisory; CVE-2025-47241. Affected versions at most 0.1.44; patched version 0.1.45. Supports the specific domain-restriction defect.

4. Cyber Security Agency of Singapore. Ongoing Campaign Targeting Chrome Browser Extensions. 30 December 2024. Government security advisory. Supports confirmed malicious extension distribution and the risk of session impersonation. It does not identify a SolvxAI incident or establish official model-provider involvement.

5. Ali Shahin Shamsabadi, Hamed Haddadi, and Artem Chaikin, Brave. Indirect Prompt Injection Remains a Fundamental Security Challenge for AI. 8 June 2026. Vendor-authored original research. Supports the Tabstack and Cotypist demonstrations and their differing action authority; Tabstack's fix was verified on 1 June 2026.

6. Mykyta Mudryi, Markiyan Chaklosh, and Grzegorz Wojcik. The Hidden Dangers of Browsing AI Agents. Submitted 19 May 2025, arXiv:2505.13076. Research preprint; peer review is not established by the cited record. Used for the authors' stated research scope. Overlaps with reference 3 and is not counted as independent evidence for that CVE.

References continued

7. Anthropic. Mitigating the Risk of Prompt Injections in Browser Use. 24 November 2025. Provider-authored safety evaluation and mitigation description. Supports residual-risk and evaluation-context discussion; reported results are not deployment incident probabilities.

8. OpenAI. Keeping Your Data Safe When an AI Agent Clicks a Link. 2026; exact publication day not exposed in the retrieved page. Provider-authored security engineering account. Supports URL-based disclosure risk and the bounded purpose of a network-layer mitigation.

9. OWASP GenAI Security Project. LLM06:2025 Excessive Agency. 2025 edition. Professional security guidance. Supports constrained authority, downstream authorization, human approval, and the distinction between preventive controls and monitoring. Guidance, not an empirical estimate or product certification.

10. OWASP Cheat Sheet Series. Logging Cheat Sheet. Living guidance; accessed 15 September 2026. Supports event context, actor classification, confidence, and exclusion of secrets from logs. No fixed publication date was established from the retrieved page.

11. SolvxAI. External-agent guard, browser-session authorization, and Feedback Security implementation assessment. 15 September 2026; application source snapshot 48f2aeee. First-party implementation evidence. Reviewed request enforcement, client detection and observation, policy history, session ownership checks, provider-label derivation, bridge grants and control state, event presentation, and retention behavior. Source code is not publicly linked; this report states the assessed behavior and its limits. This evidence does not certify production deployment, comprehensive instrumentation, or independent security effectiveness.

Publication and disclosure note

This report is published by SolvxAI and describes a product approach in which SolvxAI has a commercial interest. External findings are attributed to their original publishers and their evidentiary limitations are preserved. The control mapping and engineering examples are SolvxAI's analysis. No customer screenshots, names, IP addresses, well identifiers, credentials, or private session records are reproduced.

The report may be shared and cited with attribution to SolvxAI and the edition. External source material remains subject to its original terms. Readers evaluating a historical vulnerability should consult the maintainer's current advisory and release information. Readers evaluating SolvxAI should verify the deployed release, active policy, observed reporting, and any additional controls on which their operating arrangement depends.