MARCEL
KÜPPERS

SECURITY · STRATEGY · RESILIENCE

WHITEPAPER

AGENTIC AI SECURITY PAPERS · NR. 02

Agent Inventory: Was läuft eigentlich bei uns?

Warum die Bestandsaufnahme handelnder Systeme schwieriger ist als jede Inventur davor – und wie aus einer Liste eine Kontrolle wird.

FÜR

CISO-Office, GRC, IT-Governance, Enterprise Architecture

UMFANG

18 Seiten · Lesezeit ~14 Minuten

STAND

Version 1.0 · August 2026

KURZFASSUNG

Paper #01 dieser Reihe endet mit sechs Fragen, die eine Architektur beantworten muss, bevor ein Agent eine folgenreiche Aktion ausführt. Allen sechs ist eine siebte vorgelagert, und sie ist in den meisten Unternehmen unbeantwortet: **Welche Agenten führen bei uns überhaupt schon Aktionen aus?**

Diese Frage klingt nach Verwaltung. Sie ist es nicht. Ohne Antwort lässt sich keine der übrigen Kontrollen anwenden – man kann weder Fähigkeiten begrenzen noch Aktionen autorisieren noch einen Agenten stoppen, den man nicht kennt.

Zugleich ist die Inventur handelnder Systeme schwieriger als jede Bestandsaufnahme davor. Agenten werden eingeschaltet, nicht beschafft. Sie erscheinen als Funktion in bereits genehmigter Software. Sie werden von Menschen gebaut, die keine Entwickler sind. Und ihr Risiko hängt an dem, was sie erreichen können – eine Eigenschaft, die sich durch eine Konfigurationsänderung ändert, nicht durch ein Release.

Dieses Papier beschreibt, **wie man Agenten findet**, **was man über sie erfassen muss** und vor allem, **wie aus einer Liste eine wirksame Kontrolle wird**.

DIE ZENTRALE UMSTELLUNG

Ein Agent Inventory, das auf Selbstauskunft beruht, ist eine Umfrage. Ein Agent Inventory, das am Tool-Zugriff durchgesetzt wird, ist eine **Kontrolle**.

INHALT

1 Die Frage, die niemand beantworten kann

Warum die Bestandsaufnahme am Anfang steht und trotzdem meistens fehlt.

Fragen Sie in Ihrem Unternehmen, wie viele Server im Rechenzentrum stehen: Sie bekommen eine Zahl. Fragen Sie, wie viele Anwendungen im Einsatz sind: Sie bekommen eine Zahl mit einer Fehlerspanne. Fragen Sie, wie viele Systeme selbstständig Aktionen in Fachverfahren auslösen: Sie bekommen ein Gespräch.

Das ist kein Versäumnis der IT. Es liegt daran, dass die Frage neu ist und die vorhandenen Register für andere Fragen gebaut wurden. Der Unterschied zu früheren Inventuren liegt in drei Eigenschaften:

Agenten werden eingeschaltet. Die klassische Erfassungslogik hängt an der Beschaffung: Was gekauft wird, wird erfasst. Agentische Funktionen erscheinen dagegen als Erweiterung von Software, die vor Jahren beschafft und längst freigegeben wurde. Zwischen „wir nutzen dieses Ticketsystem“ und „dieses Ticketsystem beantwortet Anfragen jetzt selbstständig und greift dafür auf die Dateiablage zu“ liegt nur ein Häkchen in einer Administrationsoberfläche.

Agenten werden von Menschen gebaut, die keine Entwickler sind. Die Werkzeuge dafür sind bewusst so gestaltet: Ein Fachbereich beschreibt in natürlicher Sprache, was passieren soll, wählt aus einer Liste die Systeme aus, auf die zugegriffen werden darf, und veröffentlicht das Ergebnis für Kolleginnen und Kollegen. Kein Repository, kein Deployment, keine Änderung, die in einem Change-Prozess auftauchen würde.

Das Risiko hängt an der Reichweite des Agenten. Zwei Agenten mit demselben Namen und derselben Aufgabe können völlig verschiedene Schadensradien haben, je nachdem, welche Werkzeuge in ihrer Laufzeitumgebung eingebunden sind. Und diese Reichweite ändert sich durch eine Konfiguration, nicht durch ein Release.

Was inventarisiert werden muss, ist nicht der Agent. Es ist seine Handlungsvollmacht.

Was überhaupt zählt

Bevor man zählt, muss man wissen, was gezählt wird. Der Begriff „Agent“ ist unscharf, und eine Abgrenzung über das Modell führt in die Irre – nicht jede Software mit Sprachmodell handelt, und nicht jedes handelnde System nutzt ein Sprachmodell. Die brauchbare Abgrenzung folgt der Argumentation aus Paper #01:

> Arbeitsdefinition: Inventarisierungspflichtig ist ein System, das (1) Werkzeuge aufrufen oder auf andere Weise extern wirksame Aktionen auslösen kann und (2) diese Aufrufe nicht vollständig aus vorab festgelegter Programmlogik ableitet.

Damit fällt ein Skript, das jede Nacht denselben Report erzeugt, nicht darunter: sein Handeln ist determiniert. Ein Assistent, der ausschließlich Texte vorschlägt, fällt ebenfalls nicht darunter: er handelt nicht. Ein System, das aus einer Modellausgabe ableitet, welche Schnittstelle es mit welchen Parametern aufruft, fällt darunter, unabhängig davon, ob jemand es „Agent“ nennt.

2 Warum die vorhandenen Register nicht greifen

Jedes bestehende Verzeichnis hat recht – und ist genau an dieser Stelle blind.

Der naheliegende Reflex lautet: Wir haben doch Register. Das stimmt. Sie beantworten nur jeweils eine andere Frage.

REGISTER	ERFASST	BLINDER FLECK BEI AGENTEN
CMDB	Systeme und ihre Beziehungen	Modelliert Betriebsmittel, keine Handlungsvollmachten. Ein Agent ist kein Configuration Item mit stabiler Ausprägung.
Anwendungsportfolio	beschaffte Anwendungen	Der Agent ist eine Funktion eines bestehenden Eintrags.
IAM- und NHI-Verzeichnis	technische Identitäten	Kennt den Principal, nicht seinen Handlungsraum. Ein Service Principal sagt nichts über die dahinterliegenden Werkzeuge.
KI-Register nach AI Act	Anwendungsfälle und Modelle	Erfasst Zweck und Modellherkunft, nicht die Aktionsfläche.
Shadow-IT-Erkennung	nicht genehmigte Dienste	Ein Agent in einem genehmigten Dienst ist definitionsgemäß kein Schatten.
Change- und Release-Prozess	Änderungen an Software	Die risikorelevante Änderung ist eine Konfiguration, kein Release.

Das Muster ist überall dasselbe: Die Register modellieren **Systeme, Identitäten oder Zwecke**. Was sie nicht modellieren, ist die Verbindung aus beidem – welcher Handelnde welche Werkzeuge gegen welche Ressourcen einsetzen darf.

Am deutlichsten wird das beim IAM-Verzeichnis, weil es der Sache am nächsten kommt. Es weiß, dass ein Service Principal existiert und welche Berechtigungen er trägt. Es weiß nicht, dass hinter diesem Principal eine probabilistische Instanz sitzt, die selbst entscheidet, wann sie diese Berechtigungen einsetzt. Genau diese Differenz war der Ausgangspunkt von Paper #01.

3 Ein Vorfall ohne Zuständigen

Der teuerste Moment ist die Frage nach dem Vorfall.

Ein Fachbereich aktiviert in einem seit Jahren genehmigten Ticketsystem eine Assistenzfunktion. Sie soll eingehende Anfragen vorsortieren und passende Dokumente heraussuchen. Damit das funktioniert, verbindet der Administrator zwei Konnektoren: einen zur Dateiablage der Abteilung und einen zum Ausgangspostfach für Antworten an Anfragende.

Nichts daran ist unvernünftig. Es gab keine Beschaffung, keinen Change, keine neue Anwendung. Der Dienst stand bereits auf der Freigabeliste.

Sechs Wochen später fällt in der Protokollauswertung auf, dass Dokumente aus der Dateiablage an Adressen außerhalb des Unternehmens gegangen sind. Die Anfragen, die das ausgelöst haben, kamen über das Ticketsystem – von außen, wie vorgesehen.

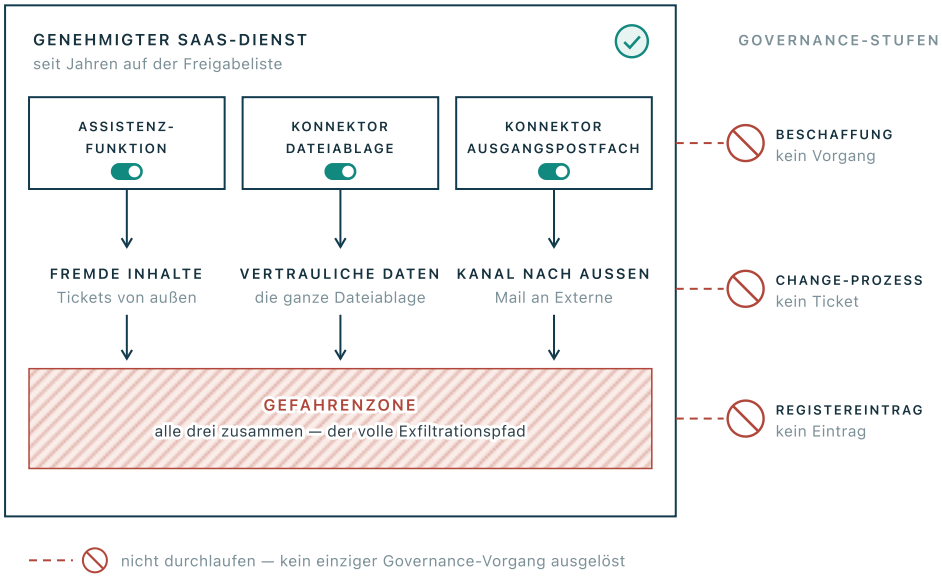

ABB. 1 **Der unbekannte Agent.** Drei Konfigurationsschritte, kein einziger Governance-Vorgang.

Der Agent hielt alle drei Eigenschaften aus Paper #01 gleichzeitig: Zugriff auf vertrauliche Daten, Verarbeitung von Inhalten, die ein Angreifer beeinflussen kann, und einen Kanal nach außen. Niemand hatte diese Entscheidung getroffen – sie entstand aus drei einzeln unauffälligen Konfigurationsschritten.

Der eigentliche Schaden zeigt sich aber erst danach, in den Fragen, die jetzt beantwortet werden müssen:

- Welches System hat gehandelt, in welcher Version, mit welcher Konfiguration?
- Unter wessen Identität, und mit welcher Befugnis?
- Wer im Unternehmen verantwortet diesen Agenten?
- Auf welche Daten konnte er zugreifen – und auf welche hat er zugegriffen?
- Gibt es weitere Agenten mit derselben Konstellation?
- Wie schalten wir ihn ab, ohne den Betrieb des Ticketsystems zu stoppen?

Für ein Unternehmen im Anwendungsbereich von NIS2 kommt Zeitdruck hinzu: Die Meldekette beginnt mit der Kenntnis, nicht mit der Klärung. Wer die ersten vier Fragen nicht innerhalb von Stunden beantworten kann, meldet unvollständig oder spät.

Ein unbekannter Agent ist kein Inventarproblem. Er ist ein Vorfall, der bereits läuft und nur noch nicht bemerkt wurde.

4 Wie man Agenten findet

Selbstauskunft findet die braven Agenten. Für die übrigen braucht es Spuren.

Der übliche erste Schritt ist eine Umfrage: Die Fachbereiche werden gebeten, ihre KI-Anwendungen zu melden. Das ist sinnvoll und liefert schnell Ergebnisse – aber es findet systematisch nur, was die Befragten selbst als meldepflichtig erkennen. Wer eine Funktion in einem genehmigten Werkzeug eingeschaltet hat, hat aus seiner Sicht nichts eingeführt.

Deshalb muss Selbstauskunft durch Spurensuche ergänzt werden. Und dabei gilt eine Eigenschaft, die den Aufwand rechtfertigt: **Jedes Signal findet eine andere Klasse von Agenten.** Kein einzelner Suchlauf ist vollständig.

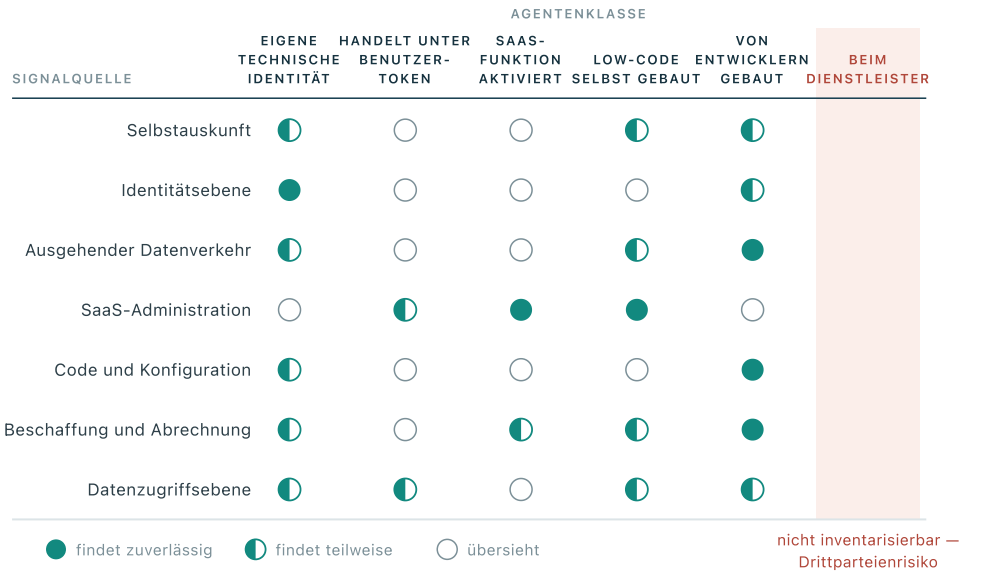

ABB. 2 **Discovery-Matrix.** Jede Quelle findet eine andere Klasse — keine findet alle.

Identitätsebene. Neu erteilte OAuth-Einwilligungen und Anwendungsregistrierungen, nicht-menschliche Identitäten mit auffallend regelmäßiger Tokenaktivität, langlebige API-Schlüssel. Findet: Agenten, die eigene technische Identitäten angelegt haben. Übersieht: Agenten, die unter der Identität ihres Benutzers handeln – und das ist der häufigste Fall.

Ausgehender Datenverkehr. Verbindungen zu den Endpunkten von Modellanbietern, auch aus Systemen, von denen man es nicht erwartet. Findet: selbstgebaute Agenten und Werkzeuge mit direkter Modellanbindung. Übersieht: Agenten, deren Modellzugriff beim SaaS-Anbieter stattfindet und die eigene Umgebung nie verlässt.

SaaS-Administrationsebene. Welche Mandanten haben agentische Funktionen aktiviert, welche Konnektoren und Werkzeugbindungen sind eingerichtet, wer darf sie einrichten. Findet: genau die Klasse aus dem Szenario in Kapitel 3. Erfordert: einen Zugang zu jeder relevanten Administrationsoberfläche, was in der Praxis der aufwendigste Teil ist.

Code und Konfiguration. Konfigurationsdateien für Werkzeugserver, Abhängigkeiten zu Agenten-Frameworks, abgelegte Systemanweisungen in Repositories. Findet: alles, was Entwicklungsteams gebaut haben. Übersieht: alles außerhalb der Versionsverwaltung – und das ist bei Low-Code-Agenten der Normalfall.

Beschaffung und Abrechnung. Rechnungen von Modellanbietern, Kreditkartenabrechnungen, Erweiterungen bestehender Lizenzverträge. In der Praxis das zuverlässigste Signal für Agenten, die an allen technischen Kontrollen vorbeigebaut wurden – bezahlt werden muss trotzdem.

Datenzugriffsebene. Auffällige Zugriffsmuster: sehr viele Einzelabrufe in kurzer Zeit, Zugriffe außerhalb von Arbeitszeiten, Lesezugriffe ohne zugehörige Benutzeraktivität. Findet: Agenten, die bereits arbeiten, unabhängig davon, wie sie entstanden sind. Liefert zugleich die Datengrundlage für die Verlaufsbetrachtung aus Paper #01.

Discovery ist kein Projekt

Der wichtigste Unterschied zu einer klassischen Inventur: Ein Agent kann seinen Schadensradius zwischen zwei Erhebungen vollständig verändern, ohne dass sich an ihm etwas ändert, was ein Änderungsprozess bemerken würde. Ein zusätzlicher Konnektor genügt.

Ein Inventar, das einmal im Jahr erhoben wird, beschreibt deshalb einen Zustand, den es nicht mehr gibt. Die Erhebung muss laufen wie die Schwachstellenerkennung: dauerhaft, mit Abweichungsmeldung.

5 Was ein Registereintrag enthalten muss

Eine Beschreibung von Handlungsvollmachten.

Die Versuchung ist groß, das Inventar als Tabelle mit Name, Fachbereich und Ansprechpartner zu führen. Ein solcher Eintrag beantwortet keine der Fragen aus Kapitel 3. Ein brauchbarer Eintrag folgt den sechs Fragen aus Paper #01 und macht sichtbar, welche davon beantwortet sind:

REGISTEREINTRAG · WER HANDELT, FÜR WEN, UND WOHER ER KOMMT

```
IDENTITÄT
  agent_id                 procurement-assistant
  laufzeitinstanz          v3.4 · pod-91347 · policy-bundle-42
  technischer_principal    sp://procurement-assistant/prod
  handelt_unter            eigener Identität | Benutzer-Token   ← Marker

VERANTWORTUNG
  auftraggebender_principal  Mensch | Dienst | Geschäftsprozess
  verantwortlicher_owner     Fachbereich Einkauf, Leitung
  security_ansprechpartner   ISB Einkauf

HERKUNFT
  entstehung               selbst gebaut | beschafft | per Update erschienen
  erstmals_gesehen         2026-06-14  (Quelle: SaaS-Administration)
```

```
REGISTEREINTRAG · FORTSETZUNG

FÄHIGKEITEN
  werkzeuge                read_offers, write_case_note, send_mail
  davon schreibend         write_case_note
  davon extern wirksam     send_mail                          ← Marker
  davon rechteverändernd   –
  erreichbare_agenten      keine

DATEN
  zugriff_auf              Lieferantenstammdaten, Angebotsablage
  höchste_klassifikation   vertraulich                        ← Marker
  verarbeitet_fremde_inhalte  ja (Lieferanten-PDF, E-Mail)    ← Marker

RISIKO
  trifecta_erfüllt         ja  – alle Marker gesetzt
  autonomiegrad            bis zu 8 Schritte ohne Freigabe
  höchste_umkehrbarkeit    schwer umkehrbar
  laufzeitautorisierung    nein                               ← Lücke

BETRIEB
  abschaltweg              Konnektor deaktivieren (getestet: nein)
  letzte_bestätigung       2026-07-01
```

Drei Dinge an diesem Eintrag verdienen Aufmerksamkeit.

Die Trifecta-Prüfung ist ableitbar, nicht zu erfragen. Ob ein Agent vertrauliche Daten sieht, fremde Inhalte verarbeitet und nach außen kommunizieren kann, ergibt sich aus den markierten Feldern. Das Register kann diese Prüfung automatisch durchführen – und genau das macht es zu einem Sicherheitswerkzeug statt zu einer Dokumentation.

Auftraggebender Principal und verantwortlicher Owner sind getrennt. Wer einen Agenten auslöst, ist nicht automatisch, wer für ihn einsteht. Bei einem Agenten, den ein Zeitplan oder ein Alarm startet, gibt es überhaupt keinen menschlichen Auftraggeber – einen Verantwortlichen muss es trotzdem geben.

Die Lücken sind Teil des Eintrags. Ein Feld „Laufzeitautorisierung: nein“ ist ein Befund. Das Register wird damit zur Arbeitsliste für die Folgepapiere dieser Reihe.

Die Reichweite reicht über den einzelnen Agenten hinaus

Paper #01 formuliert die Entwurfsregel für die Trifecta bewusst über den gesamten erreichbaren Pfad, nicht über den einzelnen Agenten. Für das Register heißt das: Das Feld `erreichbare_agenten` ist kein Beiwerk. Zwei Agenten, die einzeln unauffällig aussehen, können zusammen einen vollständigen Pfad von fremden Inhalten zu vertraulichen Daten und nach außen ergeben.

Ein Inventar, das nur Einzeleinträge führt, findet diese Konstellation nicht. Eines, das die Beziehungen zwischen den Einträgen führt, findet sie durch eine Abfrage.

6 Vom Register zur Kontrolle

Der Unterschied zwischen einer Liste und einer Grenze.

Hier entscheidet sich, ob die Arbeit etwas wert war. Ein gepflegtes Register, das niemanden hindert, ist Dokumentation. Es veraltet ab dem Tag seiner Fertigstellung.

Der Hebel liegt in der Umkehrung: **Der Registereintrag wird zur Voraussetzung für den Werkzeugzugriff.**

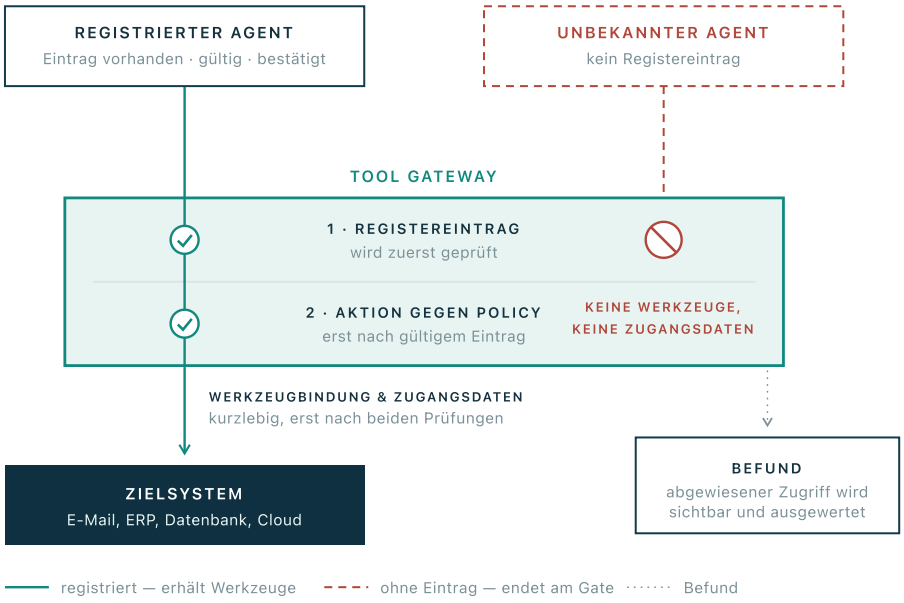

ABB. 3 **Registrierungs-Gate.** Ein Inventar, ohne das nichts funktioniert, pflegt sich selbst.

Technisch ist das kein neuer Baustein. Paper #01 beschreibt einen Tool Gateway, über den jede folgenreiche Aktion laufen muss. Dieser Gateway kennt den Handelnden bereits – er muss lediglich zusätzlich prüfen, ob für diesen Handelnden ein gültiger Registereintrag existiert. Ein unbekannter Agent bekommt dann keine Werkzeuge. Niemand hat ihn verboten; es hat ihm nur niemand welche gegeben.

Ein Inventar, das man pflegen muss, verfällt. Ein Inventar, **ohne das nichts funktioniert**, pflegt sich selbst.

Zwei Ebenen sind dabei zu trennen:

Bestand – was existiert, unabhängig davon, ob es erlaubt ist. Speist sich aus der Erkennung nach Kapitel 4 und enthält ausdrücklich auch das Unerwünschte. Ein Agent, der hier auftaucht und keinen Freigabeeintrag hat, ist ein Befund.

Freigabe – was handeln darf, mit welchen Fähigkeiten, unter welchen Grenzen. Das ist der Eintrag aus Kapitel 5. Er ist zugleich der Vertrag zwischen Fachbereich und Security: Der Fachbereich beschreibt, was der Agent tun soll; Security legt fest, was er dafür bekommt.

Die Differenz zwischen beiden Ebenen ist die eigentliche Kennzahl. Sie sagt mehr über den Zustand eines Unternehmens aus als die absolute Zahl der Agenten.

Was den Eintrag aktuell hält

- Bestätigung durch den Owner in festem Rhythmus, mit automatischem Ablauf. Ein Eintrag, den seit einem Jahr niemand bestätigt hat, verliert seine Freigabe.
- Abweichungsmeldung aus dem Abgleich zwischen Erkennung und Register: neue Werkzeugbindung, neuer Datenzugriff, neu erfüllte Trifecta.
- Kopplung an den Änderungsprozess des Anbieters: Wenn ein SaaS-Anbieter agentische Funktionen ausliefert, ist das eine meldepflichtige Änderung an einem IKT-Drittdienst, kein Produktupdate, das man geschehen lässt.
- Getesteter Abschaltweg je Eintrag. Ein Abschaltweg, der nie ausprobiert wurde, ist eine Behauptung.

7

Was ein Inventar nicht leisten kann

Vier Grenzen, die man kennen muss, bevor man sich darauf verlässt.

Vollständigkeit ist nicht erreichbar. Selbstauskunft ist unvollständig, Erkennung ist unvollständig, und beides zusammen ist es immer noch. Der richtige Anspruch ist ein Register, das die riskanteste Klasse zuverlässig erfasst – Agenten mit Schreibrechten, mit Zugriff über Sicherheitsdomänen hinweg oder mit einem Kanal nach außen. Wer Vollständigkeit verspricht, verschiebt die Arbeit ins Unendliche und liefert nichts.

Ein Eintrag beschreibt Absicht, nicht Wirklichkeit. Was im Register steht, ist die dokumentierte Fähigkeit. Was der Agent tatsächlich kann, ergibt sich aus seiner Laufzeitumgebung. Beides fällt auseinander, sobald jemand eine Konfiguration ändert. Nur die Durchsetzung am Gateway aus Kapitel 6 macht den Eintrag wahr – ohne sie bleibt er eine Behauptung mit Zeitstempel.

Agenten in fremden Tenants sind nicht inventarisierbar. Wenn ein Dienstleister einen Agenten auf Daten ansetzt, die er von uns erhalten hat, findet keine unserer Signalquellen ihn. Das ist eine Frage des Drittparteienmanagements: Sie gehört in Verträge, Sicherheitsfragebögen und Prüfrechte, nicht in die technische Erkennung.

Die Abgrenzung bleibt strittig. Die Arbeitsdefinition aus Kapitel 1 zieht eine brauchbare Linie, aber sie erzeugt Grenzfälle – Automatisierungen mit einem einzigen Modellaufruf, Assistenten mit genau einer schreibenden Funktion, Workflows, in denen ein Mensch formal freigibt, aber faktisch durchklickt. Diese Fälle sollte man bewusst entscheiden und die Entscheidung dokumentieren, statt sie jedes Mal neu zu diskutieren.

8 Fünf Maßnahmen

Die einzige Checkliste in diesem Papier.

Erstens: Die riskanteste Klasse zuerst erfassen, nicht die vollständige. Agenten mit Schreibrechten, mit Zugriff über mehrere Sicherheitsdomänen oder mit einem Kanal nach außen. Diese Auswahl ist in Wochen zu schaffen; die vollständige Erhebung ist es nicht.

Zweitens: Mindestens drei Signalquellen parallel nutzen. Selbstauskunft, SaaS-Administrationsebene und Abrechnung decken zusammen die häufigsten Entstehungswege ab. Jede einzelne Quelle für sich erzeugt ein falsches Sicherheitsgefühl.

Drittens: Den Registereintrag um Verantwortung und Reichweite erweitern. Name und Fachbereich reichen nicht. Notwendig sind verantwortlicher Owner, Werkzeugliste, Datenzugriff, Autonomiegrad und die daraus abgeleitete Trifecta-Prüfung.

Viertens: Bestand und Freigabe trennen – und die Differenz messen. Die Zahl der erkannten, aber nicht freigegebenen Agenten ist die aussagekräftigste Kennzahl, die dieses Thema hergibt. Sie gehört in die Berichterstattung an die Leitung.

Fünftens: Den Registereintrag zur technischen Voraussetzung machen. Kein Werkzeugzugriff ohne gültigen Eintrag. Erst damit wird aus der Inventur eine Kontrolle – und erst damit hört sie auf, zu veralten.

ABSCHLUSS

Ein Agent Inventory ist kein Verwaltungsprojekt. Es ist die Voraussetzung dafür, dass irgendeine der Kontrollen aus dieser Reihe überhaupt angewendet werden kann. Fähigkeiten lassen sich nicht begrenzen, Aktionen nicht autorisieren und Agenten nicht stoppen, solange unbekannt ist, welche es gibt.

Der Aufwand liegt dabei weniger im Erfassen als im Aktuellhalten, und die Lösung dafür ist eine Umkehrung der Beweislast.

Solange das Register den Agenten hinterherläuft, verliert es. Sobald der Agent das Register braucht, gewinnt es.

WEITERFÜHRENDE QUELLEN

OWASP GenAI Security Project – *Top 10 for Agentic Applications*. Insbesondere Identity and Privilege Abuse sowie Rogue Agents.

MITRE ATLAS – *Adversarial Threat Landscape for Artificial-Intelligence Systems*.

NIST – *AI Risk Management Framework (AI RMF 1.0)*, NIST AI 100-1, 2023. Die Funktion *Map* entspricht der hier beschriebenen Bestandsaufnahme.

Marcel Küppers – *Agentic AI Security Paper #01: Von Model Security zu Agentic Action Security*, August 2026.

Marcel Küppers – *NIS2 trifft Agentic AI*, August 2026.

Abbildungsverzeichnis

Die Serie

01	Von Model Security zu Action Security	ERSCHIENEN
02	Agent Inventory: Was läuft eigentlich bei uns?	DIESES PAPIER
03	Identität und Delegation für KI-Agenten	IN VORBEREITUNG
04	Least Capability	IN VORBEREITUNG
05	Runtime Authorization für autonome Systeme	IN VORBEREITUNG
06	MCP und Tool Security	IN VORBEREITUNG
07	Prompt Injection ist ein Aktionsproblem	IN VORBEREITUNG
08	Der Agent Action Ledger	IN VORBEREITUNG

Alle Papers erscheinen unter marcel-kueppers.de/agentic-ai-security-papers. Verwandt, außerhalb der Serie: *NIS2 trifft Agentic AI* (August 2026).

NÄCHSTER SCHRITT

Erst zählen, dann kontrollieren.

Im zweitägigen Training **„AI Agents für Security Management“** bauen Sie handelnde Agenten selbst – vollständig lokal, ohne eine Zeile Code. Sie sehen dabei aus nächster Nähe, woran ein Agent erkennbar ist, welche Spuren er hinterlässt und an welcher Stelle ein Registereintrag ihn tatsächlich aufhält. Sie gehen mit lauffähigen Werkzeugen und einem 90-Tage-Plan nach Hause.

MEHR ERFAHREN UND PLATZ SICHERN

marcel-kueppers.de/ai-agents-security-management

Marcel Küppers · Security.Strategy.Resilience. – Agentic AI Security Papers, Nr. 02 „Agent Inventory: Was läuft eigentlich bei uns?“, Version 1.0, Stand August 2026.

Quellen (Auswahl): OWASP GenAI Security Project, Top 10 for Agentic Applications · MITRE ATLAS · NIST AI RMF 1.0 (2023) · Marcel Küppers, Agentic AI Security Paper #01 (2026). Framework-Stände bitte zum Zeitpunkt der Anwendung prüfen.