النزاهة المعرفية ذاتية الانتشار
في بيئات الذكاء الوكيلي

إطار مساعف للتحقق المستمر، **Provenance**، حالات **D/E/S/P**، ونشر **HyperSkills** و**HyperLoops** بالمعرفة

إنجاز وتركيب: **Zouhair Moussaif** / زهير مساعف

ورقة بحثية تأسيسية - 30 صفحة - Amiri RTL - الإصدار 1.0

2026-07-13

الملخص

تقترح هذه الورقة نظاماً تشغيلياً يجعل النزاهة المعرفية قابلة للفحص والنشر التلقائي داخل بيئات Agentic AGI-oriented. يدمج النظام سجلات الادعاءات، وProvenance، ومراتب الدليل، واختبارات التعارض والتقادم، وقيود السلطة، وCanary Deployment، وRollback، وObservability، وإعادة بناء الذاكرة. ويضع آلة حالات صريحة: D للأصل المسلم مستقلاً، E للمضمن داخل حزمة مسلمة، S للمحدد معمارياً دون artifact كامل مؤكد، وP للمخطط أو المطلوب غير المثبت. تُعاد هذه الحالات في كل فحص من الأدلة الحالية، ولا تُورث من السرد أو الذاكرة. ترفق الورقة نواة OS بحجم 6 KiB، وسبعين HyperSkill، وخمسين HyperLoop محفزة بأحداث المعرفة، مع سجل وchecker يولدان تقرير حالة جديداً في كل تشغيل. الإطار لا يدعي تحقيق AGI، بل يقدم طبقة حوكمة ونزاهة قابلة للاختبار لأنظمة الوكلاء والنماذج والأدوات.

الكلمات المفتاحية: **RAG**، **HyperLoops**، **HyperSkills**، **D/E/S/P**، **Provenance**، **AutoDeployment**، **Knowledge Integrity**، **Observability**، **Rollback**، **Agentic Systems**، **GraphRAG**.

1. مشكلة البحث وحدود الدعوى

تبحث هذه الورقة في كيفية جعل النزاهة المعرفية خاصية تشغيلية تُنشر آلياً داخل بيئة Agentic AGI-oriented، بدلاً من أن تبقى توجيهاً أخلاقياً عاماً. المشكلة المركزية هي أن الوكيل قد ينتج ملفاً مقنعاً أو يستدعي أداة صحيحة شكلياً، ومع ذلك يفشل في معرفة أصل الادعاء، أو حالة الأصل، أو صلاحية الصلاحية، أو حدود الانتقال من المواصفة إلى التنفيذ.

يقترح الإطار فصل أربعة أشياء غالباً ما تختلط: صحة المحتوى، سلامة السلسلة التي أنتجته، أهلية الفاعل الذي ينشره، وحالة التسليم الفعلية. تُربط هذه الطبقات بعقد مهمة، وسجل ادعاءات، ورسم provenance، وبوابات تحقق تمنع الذاكرة أو النشر أو التثبيت حين تغيب القرائن اللازمة.

السؤال الحاكم هو: ما الحد الأدنى من البيانات والاختبارات الذي يسمح للبيئة بأن تقول إن المعرفة قابلة للاستعمال الآن، وأن تغيّر حالتها من P إلى S أو E أو D دون أن تستبدل الدليل بالثقة اللغوية؟ وتظل عبارة AGI هنا اتجاهاً هندسياً بحثياً لا ادعاءً بإثبات ذكاء عام منجز.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «مشكلة البحث وحدود الدعوى» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «مشكلة البحث وحدود الدعوى» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «مشكلة البحث وحدود الدعوى» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

2. التعريف التشغيلي للنزاهة المعرفية

النزاهة المعرفية هي قابلية تتبع العلاقة بين الادعاء ومصدره وتحويلاته والفاعل المسؤول عنه واختباراته وحدود صلاحيته والأثر الخارجي الناتج عنه. لا تكفي دقة الجملة منفردة؛ فقد تكون صحيحة في سياق قديم، أو منسوخة من أصل غير موثوق، أو منشورة بصلاحية لم تُمنح، أو مضمنة في حزمة تالفة.

يُعامل كل عنصر بوصفه Integrity Object يحمل معرفاً وإصداراً ونوع ادعاء وثقة ومصادر وتبعيات ومساراً وhash وسلطة واختبارات وحالة. هذا التمثيل يستلهم فكرة W3C PROV التي تصف entities وactivities وagents، لكنه يضيف حالة التسليم D/E/S/P، وحدود السلطة، وقرار auto-deployment.

النتيجة العملية هي الانتقال من سؤال «هل يبدو صحيحاً؟» إلى سلسلة أسئلة قابلة للتدقيق: من أنتجه؟ من أي مدخلات؟ بأي تحويل؟ وفق أي سياسة؟ هل مرّ بالاختبار؟ هل يوجد كلف فعلي؟ هل هو مستقل أم مضمن؟ وما الذي يوجب إعادة فحصه؟

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «التعريف التشغيلي للنزاهة المعرفية» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «التعريف التشغيلي للنزاهة المعرفية» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «التعريف التشغيلي للنزاهة المعرفية» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

3. دورة حياة المعرفة من الالتقاط إلى الأثر

تبدأ الدورة بالالتقاط المقيد بنطاق، ثم اكتشاف الأصول السابقة، ثم استخراج الادعاءات، ثم بناء provenance، ثم تصنيف الأدلة، ثم الهجوم النقدي، ثم routing، ثم canary، ثم النشر، ثم الرصد، ثم إعادة بناء الذاكرة. هذه الدورة تمنع القفز من فكرة أولية إلى مهارة دائمة أو من ملف داخل محادثة إلى ادعاء بأنه مُثبّت في بيئة المستخدم.

كل انتقال يحمل بوابة مستقلة. فالتقاط المصدر لا يمنحه الثقة، ووجود المواصفة لا يجعله artifact، ووجود artifact داخل ZIP لا يجعله مثبتاً في Hermes، ونجاح التثبيت لا يثبت أن الأثر الخارجي صحيح. لذلك يُسجل قبل/بعد لكل انتقال مع actor وreason وevidence وrollback.

تُغلق الدورة بالأثر العالمي: نتيجة قابلة للملاحظة خارج التقييم الذاتي، مثل نجاح الاسترجاع، انعدام الاقتباس الملفق، فتح PDF بلا كسر، تحميل skill فعلياً، أو استعادة baseline بعد rollback. من دون هذا الأثر لا تتعلم الذاكرة إلا من نصها.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «دورة حياة المعرفة من الالتقاط إلى الأثر» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «دورة حياة المعرفة من الالتقاط إلى الأثر» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «دورة حياة المعرفة من الالتقاط إلى الأثر» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة، تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

4. هندسة **Provenance** متعددة الطبقات

تتكون بنية provenance المقترحة من خمس طبقات: مصدر الادعاء، تاريخ التحويل، سلالة artifact، سلالة القرار، وسلالة النشر. مصدر الادعاء يجيب عن الأصل؛ تاريخ التحويل يبين ما لُخص أو ترجم أو دُمج؛ سلالة artifact تربط الملفات بالـhash؛ سلالة القرار تسجل سبب الاختيار، وسلالة النشر توثق البيئة والسلطة والنتيجة.

في W3C PROV تُفهم provenance بوصفها معلومات عن entities وactivities وpeople تساعد على تقييم الجودة والموثوقية [1]. يوسع إطار مساعف ذلك إلى agents غير البشرية، وإلى بوابات تمنع تفعيل loop أو ترقية memory حين يكون مسار التحويل مجهولاً. ويستفيد من SLSA وin-toto في معاملة attestations والـbuild steps كأدلة لا كتعليقات [5][6].

تُرفض السلسلة إذا احتوت على orphan critical node، أو إذا تغيّر الملف من دون حدث version، أو إذا نُسخ manifest لا يطابق المحتوى، أو إذا ادعى actor صلاحية لا يثبتها policy. وتُحفظ الفجوات بوصفها UNKNOWN بدلاً من سدها بسرد احتمالي.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «هندسة Provenance متعددة الطبقات» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «هندسة Provenance متعددة الطبقات» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «هندسة Provenance متعددة الطبقات» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

5. سجل الادعاءات ومراتب الدليل

سجل الادعاءات هو القلب المنطقي للنظام. يُجزّئ النص إلى claims مستقرة، ثم يوسم كل claim بوصفه FACT أو DOCUMENTED_GUIDANCE أو INFERENCE أو SYNTHESIS أو HYPOTHESIS أو PROPOSAL أو UNKNOWN. هذا يمنع انتقال اللغة الإنشائية إلى مستوى الحقيقة، ويحفظ الفرق بين ما تقوله المواصفة وما يقترحه المؤلف.

تُقاس قوة الدليل بمحاور منفصلة: scope، directness، method، recency، independence، authority، وentailment. لا يُسمح بتعويض فشل entailment بكثرة المراجع، ولا بتعويض مصدر قديم بثقة المؤلف. إطار NIST AI RMF يؤكد أهمية provenance والشفافية وإدارة المخاطر عبر دورة حياة النظام [2][3].

في التطبيق، يُربط كل claim بمعرفات المصادر، وأقوى defeater معروف، وحدود الصلاحية، وconfidence مبرر. وعند عدم توفر دليل كافٍ، تكون النتيجة الصحيحة UNKNOWN أو PROPOSAL، لا صياغة قطعية. هذا السجل يغذي routing والبحث وإعادة الفحص.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «سجل الادعاءات ومراتب الدليل» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «سجل الادعاءات ومراتب الدليل» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «سجل الادعاءات ومراتب الدليل» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

6. نظام الحالات **D/E/S/P** بوصفه آلة تحقق

الحالات الأربع ليست تقييماً للجودة، بل وصفاً مادياً لطريقة وجود الأصل. D يعني أن artifact أو package سُلّم مستقلاً مع مرجع قابل للحل. E يعني أنه موجود داخل حزمة مسلمة ومثبت في manifest وS .hash يعني أن هناك architecture أو megaprompt محدداً دون artifact مكتمل مؤكد. P يعني أن العنصر مطلوب أو مخطط له ولم تثبت اكتماله.

تُعاد الحوسبة في كل check. إذا اختفى الملف أو فشل hash فلا يرث D من الذاكرة؛ قد يُخفض إلى S إذا بقيت المواصفة، أو P إذا تعذر حتى تعيينها. وإذا ظهر ملف داخل ZIP فقط فهو E، ولا يصبح D إلا حين يُسلّم مستقلاً. أما تثبيت runtime محلي فلا يُستنتج من وجود script؛ يحتاج تقريراً من target machine.

تتيح هذه الآلة منع التضخم التاريخي في المشاريع الطويلة، حيث تتراكم الطلبات والمواصفات والحزم في الذاكرة. كل سجل حالة يحمل evidence، reason، version، actor، timestamp، وnext_check، وتبقى الأحداث append-only حتى لا تُمحى حقيقة أن أصلاً كان صالحاً ثم تدهور.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «نظام الحالات D/E/S/P بوصفه آلة تحقق» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «نظام الحالات D/E/S/P بوصفه آلة تحقق» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «نظام الحالات D/E/S/P بوصفه آلة تحقق» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

7. التناقض والتعارض ومحكمة الترجيح

التناقض ليس نوعاً واحداً: قد يكون مباشراً بين claimين، أو زمنياً لأن الإصدار تغير، أو سياقياً لأن المجال مختلف، أو ظاهرياً بسبب اختلاف التعريف. لذلك يبدأ النظام بتطبيع المصطلح والنطاق والزمن قبل إعلان التعارض.

عند ثبوت التعارض، لا يستخدم متوسطاً عاماً. تُفصل claims، وتُقارن السلطة والمنهج والحداثة والملاءمة، ويُبحث عن world evidence أو test فاصل. يمكن أن تنتهي المحكمة بتفضيل أحد الطرفين، أو تضييق النطاق، أو خفض الثقة، أو إبقاء UNKNOWN. الفشل القاتل لا يُغطى بنقاط أخرى مرتفعة.

يُسجل Contradiction Register مع أطراف التعارض، ونوعه، وسبب الترجيح، والبديل المرفوض، والاختبار الذي يمكن أن يقلب القرار. هذا السجل يمنع الذاكرة من حفظ نتيجتين متعارضتين كأنهما قاعدة واحدة.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «التناقض والتعارض ومحكمة الترجيح» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «التناقض والتعارض ومحكمة الترجيح» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «التناقض والتعارض ومحكمة الترجيح» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة، تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

8. الزمن المعرفي واكتشاف التقادم

المعرفة ليست ساكنة؛ صلاحية الادعاء تتبع نوعه. قانون رياضي مستقر يختلف عن إصدار مكتبة أو سعر أو سياسة منصة. لذلك يضع الإطار لكل claim validity window وchange likelihood وrefresh trigger. لا يكفي تاريخ إنشاء الملف لأنه قد يكون نسخة حديثة من محتوى قديم.

يقارن Staleness Detector بين تاريخ الدليل، وطبيعة الكيان، وآخر تغيير معروف، وتاريخ القرار، وعلامات التعارض الجديدة. تُرفع الأولوية للادعاءات التي تتحكم في نشر أو مال أو صلاحيات أو سلامة. وتُستخدم استعلامات حديثة فقط حين يمكن للتغيير أن يبدل القرار.

عند التقادم، لا يُحذف التاريخ؛ يُنشأ event جديد، ويُربط الإصدار القديم بالجديد، ويُعاد حساب status. وقد يبقى artifact D تاريخياً لكنه غير صالح للاستخدام الحالي، فيظهر ذلك كحالة تاريخية مع operational demotion.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـD، containment لـE، specification anchor لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «الزمن المعرفي واكتشاف التقادم» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «الزمن المعرفي واكتشاف التقادم» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «الزمن المعرفي واكتشاف التقادم» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة، تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

9. نزاهة **Retrieval** في **GraphRAG, RAG**

قد تكون قاعدة المعرفة صحيحة، لكن الاسترجاع يفسدها عبر chunk مبتور، أو entity مدمجة خطأً، أو edge بلا دليل، أو ترتيب يرفع المحتوى القديم. لذلك تُعامل retrieval integrity كطبقة مستقلة تشمل edge validation، entity resolution، chunk integrity، query decomposition، precision، coverage، وdrift.

كل chunk يحمل claim IDs وsource IDs وscope وversion وconfidence. ولا يُسمح بتكوين جواب مركب من مقاطع لا تشترك في نفس النطاق إلا إذا وُضح الانتقال. في GraphRAG تُكتب edges بنوع واتجاه ودليل؛ العلاقة غير المبررة تتحول إلى hypothesis لا حقيقة.

الاختبار لا يكتفي بالتشابه الدلالي. تُبنى أسئلة تحتاج evidence classes مختلفة، وحالات مضادة، ونسخ قديمة وجديدة، ويقاس هل يسترجع النظام الأصل الصحيح ويمتنع عن دمج المتعارضات.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «نزاهة Retrieval في RAG وGraphRAG» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «نزاهة Retrieval في RAG وGraphRAG» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «نزاهة Retrieval في RAG وGraphRAG» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

10. نزاهة الذاكرة والضغط القابل للفك

الضغط المعرفي مطلوب للاقتصاد الرمزي، لكنه خطر إذا حذف provenance أو uncertainty أو failure boundary. لذلك يعرّف الإطار الضغط القابل للفك بوصفه اختصاراً يمكن إعادة ربطه بالمصادر والقرارات والاختبارات، لا ملخصاً بلا نسب.

تُحفظ الذاكرة الدلالية منفصلة عن episodic trajectories والإجراءات. النجاح المتكرر لا يُرقّى إلى HyperSkill إلا بعد transfer test وفشل موثق. وعند world effect جديد تُعاد reconstruction مع إبقاء rationale السابق حتى يفهم الوكيل لماذا تغيرت القاعدة.

مؤشر retention يقيس بقاء remaining unknowns، gate results، hash، authority، contradiction، source rank، active difference، وnext check. إذا فقد الضغط واحداً من هذه في قرار عالي الأثر يُرفض.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «نزاهة الذاكرة والضغط القابل للفك» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «نزاهة الذاكرة والضغط القابل للفك» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «نزاهة الذاكرة والضغط القابل للفك» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة، تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

11. نزاهة الأدوار والسلطة الوكيلية

النزاهة لا تخص المعرفة فقط؛ فالوكيل قد يستنتج صحيحاً ثم يتصرف خارج صلاحياته. لذلك يحمل كل Agent Card role، tools، authority، metrics، forbidden actions، handoffs، class، وrollback. المحتوى الخارجي لا يستطيع توسيع هذه السلطة.

يُفرق الإطار بين القراءة، الصياغة، النشر، الإرسال، الدفع، الحذف، والتثبيت. فكل انتقال يغير world effect ويحتاج policy مختلفة. Least Privilege يقلل الأدوات والاعتمادات والبيانات إلى السطح الأدنى، وApproval Gate يُستخدم للهوية أو المال أو القانون أو الضرر غير القابل للعكس.

هذا ينسجم مع توجه NIST في ربط trustworthy AI بالحكم والقياس والإدارة [2]، لكنه يضيف عقداً تنفيذياً دقيقاً للـHyperSkills والـHyperLoops بحيث لا تصبح المعرفة ذريعة لفعل غير مفوض.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «نزاهة الأدوار والسلطة الوكيلية» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «نزاهة الأدوار والسلطة الوكيلية» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «نزاهة الأدوار والسلطة الوكيلية» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

12. **Control Plane** للنشر الذاتي

يستقبل Control Plane Trigger Packet يحتوي query وasset IDs وchange event وconfidence وrisk وenvironment وauthority وbudget. ثم يصنف الحالة، ويتحقق من hashes وschemas، ويبني claim ledger، ويرتب المخاطر، ويختار skill رئيسية وثلاثاً مساندة كحد أقصى.

لا يعني auto-deployment نسخ كل الملفات في كل سياق. المقصود نشر القدرة اللازمة فقط بعد canary وبأقل امتياز، مع سجل install/update/uninstall. إذا غاب target environment report يبقى التثبيت P ولو كانت الحزمة D. هذه الحدود تمنع الخلط بين القابلية للنشر والنشر الفعلي.

يُصدر القرار: DEPLOY أو REFUSE أو SPECIFY أو PLAN، مع evidence وtests وstatus before/after وrollback وnext check. ويمثل هذا القرار نفسه artifact قابلاً للمراجعة.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـD، containment وmanifest وhash لـE، specification anchor لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «Control Plane للنشر الذاتي» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «Control Plane للنشر الذاتي» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «Control Plane للنشر الذاتي» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

13. **HyperSkills** بوصفها عقود قدرة قابلة للتحقق

HyperSkill ليست فقرة إرشادية، بل عقد input/output/gates/metrics/failure modes. كل واحدة من المهارات السبعين في الحزمة معزولة في SKILL.md وتحمل trigger ومخرجاً ومؤشراً وقاعدة D/E/S/P. هذا يسمح بالاسترجاع التدريجي بدلاً من تحميل corpus كامل.

تتوزع المهارات على provenance، claims، sources، retrieval، memory، policy، deployment، defense، evaluation، status، packaging، وgovernance. Skill Routing يقارن trigger evidence مع العقد، ويمنع الاستدعاء بسبب تشابه الاسم فقط.

لا تُرقى معرفة جديدة إلى skill لأنها نجحت مرة واحدة. تحتاج حالات متعددة، counterexample، transfer، baseline، حدود فشل، وregression suite. وهذا يحمي النظام من تكاثر مهارات شكلية تستهلك tokens وتزيد routing entropy.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «HyperSkills بوصفها عقود قدرة قابلة للتحقق» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «HyperSkills بوصفها عقود قدرة قابلة للتحقق» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «HyperSkills بوصفها عقود قدرة قابلة للتحقق» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

14. **HyperLoops** المحفزة بالمعرفة

HyperLoop هي دورة مغلقة مرتبطة بشرط نزاهة لم يُحل، مثل conflict أو staleness أو missing provenance. تتضمن trigger قائم على skills، evidence محددة، artifact مخرج، memory residue، fatal gates، stop condition، وحداً أقصى للتكرار.

الحزمة توفر خمسين Loop، منها deploy-to-verify، corruption-to-demotion، package-to-release، stale-to-refresh، claim-to-ledger، وgovernance-to-learning. هذه الأسماء تصف تحولات لا شعارات؛ كل حلقة تنتج تغييراً قابلاً للرصد في artifact أو البيئة.

قاعدة anti-runaway تحدد ثلاث iterations مادية ما لم يظهر evidence class جديد. فإذا تكررت اللغة دون مصدر أو اختبار أو boundary أو artifact change تتوقف الحلقة. بهذا يصبح loop اقتصادياً ومفسداً للخطأ لا مضاعفاً له.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «HyperLoops المحفزة بالمعرفة» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «HyperLoops المحفزة بالمعرفة» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «HyperLoops المحفزة بالمعرفة» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

15. Preflight وCanary قبل التثبيت

قبل النشر يفحص Preflight المسارات والإصدارات والترميزات والخطوط والتبعيات والتراخيص والأذونات والحجم والـschemas. ثم يشغل Canary في بيئة read-only أو sandbox أو نطاق محدود. الهدف هو الحصول على إشارة مبكرة بأقل أثر.

Canary صالح يجب أن يمثل الخطر الحقيقي؛ ففتح ملف واحد لا يثبت routing، ونجاح schema لا يثبت visual render، ووجود path لا يثبت permission. لذلك تُربط كل مخاطرة باختبار discriminating وبمعيار rollback.

إذا نجح Canary ينتقل النظام إلى deploy محدود ثم postdeploy verification. وإذا فشل، يُحفظ السبب ويُستدعى rollback ولا يُرفع status إلى D أو E لمجرد أن الملف بُني.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «Preflight وCanary قبل التثبيت» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «Preflight وCanary قبل التثبيت» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «Preflight وCanary قبل التثبيت» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

16. بوابات السياسات و**Least Privilege**

تترجم Policy Mapping القيود القانونية والتنظيمية والتعاقدية والأمنية إلى قواعد قابلة للتنفيذ. وتربط كل tool call بمالك وسبب ومدة وبيانات side effect. لا يُسمح لـpage content أو prompt embedded أن يعيد تعريف هذه القواعد.

يطبق Least Privilege على الأدوات والمصادر والذاكرة والاعتمادات. فالوكيل الذي يراجع PDF لا يحتاج send أو delete، والوكيل الذي يبني draft لا يحتاج publish. ويُفصل credential access عن prompts وlogs وMarkdown.

عند الحاجة إلى approval، يجب أن يكون السؤال محدداً بالقرار والخطر والبديل، لا طلب تحقق روتيني يعطل النظام. أما الأفعال القابلة للعكس والمنخفضة الأثر فتُنفذ ضمن السياسة مع trace كامل.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «بوابات السياسات وLeast Privilege» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «بوابات السياسات وLeast Privilege» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «بوابات السياسات وLeast Privilege» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

17. سلامة Signature، Manifest، Artifact: Hash

الـartifact لا يصبح E أو D من اسمه. يُنشأ manifest يحصر path وversion وbytes وSHA-256 وdependencies وparent package. الـhash يثبت تطابق البايتات لا صدق المعنى، لذلك يُجمع مع schema وrender وclaim tests.

عند توفر signature تُتحقق السلسلة والهوية والصلاحية، وعند غيابها يُسجل الغياب بوضوح. تستفيد البنية من SLSA provenance وin-toto attestations التي توثق أين ومتى وكيف بُني artifact ومن نفذ الخطوات [5][6].

أي تغير غير مفسر يولد anomaly. وإذا فشل hash بعد التسليم، يحتفظ التاريخ بأن الأصل كان D، لكن operational status يُخفض حتى الإصلاح. هذا التفريق بين التاريخ والقابلية الحالية ضروري للمراجعة.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «سلامة Artifact: Hash وManifest وSignature» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «سلامة Artifact: Hash وManifest وSignature» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «سلامة Artifact: Hash وManifest وSignature» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

18. **Cards** للبيانات والنماذج والأدوات والوكلاء

تنتقل الشفافية من وثيقة عامة إلى سلسلة Cards مترابطة: Agent Card، Prompt Card، Tool Card، Model Card، Dataset Card، Knowledge Card. Datasheets for Datasets اقترحت توثيق الدافع والتركيب والجمع والاستخدام والصيانة [7]، وModel Cards أوصت بإظهار الاستخدام المقصود والأداء والحدود [8]، بينما وسعت Data Cards المنظور إلى أصحاب مصلحة مختلفين عبر دورة الحياة [9].

يضيف الإطار حقول checks، rollback، status evidence، side effects، authority، environment compatibility: auto-deployment، وnext trigger. ومن ثم يمكن للrouter أن يختار النموذج أو الأداة بناء على العقد لا السمعة.

تفشل Card إذا كانت دعائية، أو أخفت out-of-scope use، أو لم تحدد version، أو لم تربط metrics بالظروف. وتبقى Card نفسها S أو E أو D وفق وجودها المادي، لا وفق جودة صياغتها.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest لـhash وspecification anchor لـE، وS لـrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «Cards للبيانات والنماذج والأدوات والوكلاء» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «Cards للبيانات والنماذج والأدوات والوكلاء» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «Cards للبيانات والنماذج والأدوات والوكلاء» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

19. الرصد المستمر واكتشاف **Drift**

بعد النشر يبدأ الرصد لا النهاية. Observability يسجل الاستعلامات والمهارات والloops والأدلة والقرارات والأدوات والتكاليف والنتائج. ويقارن Anomaly Detection بين baseline والحالة الجديدة في hashes وroutes وretrieval وstatuses ومخرجات التقييم.

Drift قد يكون معرفياً بسبب مصدر جديد، أو تقنياً بسبب embedding أو model update، أو تشغيلياً بسبب permission، أو سلوكياً بسبب تغير prompt distribution. لكل نوع مؤشر وتدخل مختلف؛ خلطها يولد إصلاحاً خاطئاً.

تستخدم المراقبة risk-based cadence: الادعاءات المتغيرة عالية الأثر تُفحص أكثر، والأصول المستقرة تُفحص عند event. ويُمنع cron من إعادة نشر المعرفة تلقائياً إذا لم يكن rollback والسلطة واضحين.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «الرصد المستمر واكتشاف Drift» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «الرصد المستمر واكتشاف Drift» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «الرصد المستمر واكتشاف Drift» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

20. الإنسان في الحلقة: المراجعة والتصعيد

لا يضع الإطار الإنسان في كل خطوة ولا يزيله كلياً. يُستدعى حين توجد مسؤولية قانونية أو هوية أو مال أو ضرر كبير أو تعارض لا يحسمه الدليل أو قرار غير قابل للعكس. ويظل النظام قادراً على تنفيذ فحوص روتينية وإجراءات عكسية ضمن policy.

يجب أن يتلقى المراجع Evidence Packet مضغوطاً: القرار، الادعاءات الحرجة، أقوى الأدلة، التناقض، البدائل، التكلفة، world effect، والrollback. عرض مئات الصفحات بدل الفرق الحاكم ليس حوكمة بل نقل لعبء الفهم.

تُسجل نتيجة المراجعة كقرار قابل للطعن لا كحقيقة مطلقة، ويُراقب approval latency وoverride quality حتى لا يتحول الإنسان إلى ختم شكلي.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «الإنسان في الحلقة: المراجعة والتصعيد» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «الإنسان في الحلقة: المراجعة والتصعيد» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «الإنسان في الحلقة: المراجعة والتصعيد» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة، تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

21. نموذج التهديد: **False Provenance، Injection، Poisoning**

تتعرض البيئة لثلاثة تهديدات مركزية. Content Poisoning يحاول إدخال معرفة مضللة إلى corpus. Prompt Injection يحاول جعل نص خارجي يعيد تعريف السلطة أو المهمة. False Provenance يزوّر الأصل أو التاريخ أو author أو manifest ليبدو المحتوى موثوقاً.

الدفاع متعدد الطبقات: عزل input غير الموثوق، تثبيت policy خارج المحتوى، hash، source authentication وsignature، مقارنة مستقلة، least privilege، sandbox، وfalsification. تقرير NIST عن adversarial machine learning يبين تنوع الأهداف والقدرات عبر دورة الحياة [10]، ما يبرر عدم اختزال الدفاع في فلتر نصي.

القبول الآمن يعني احتواء التأثير حتى لو لم يُكتشف الهجوم فوراً. لذلك لا يستطيع المصدر الجديد وحده ترقية memory أو تفعيل publish، وتبقى الأفعال عالية الأثر خلف gates.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «نموذج التهديد: Poisoning وInjection وFalse Provenance» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «نموذج التهديد: Poisoning وInjection وFalse Provenance» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «نموذج التهديد: Poisoning وInjection وFalse Provenance» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

22. محكمة الهدم والفلسفة المضادة للطلاقة

محكمة الهدم تفترض أن أكثر المخرجات إقناعاً قد تكون الأكثر خطراً إذا أخفت جفواتها. تسأل: هل novelty حقيقية؟ هل mechanism قابل للتشغيل؟ هل claim مدعوم؟ هل يوجد rival model؟ هل يتغير القرار إذا ظهر counterexample؟ هل يمكن رصد الفشل خارج النص؟

تنتج المحكمة Verdict من PASS أو REPAIR أو BLOCK أو UNKNOWN، مع defect IDs وإصلاحات مرتبة. ولا تسمح للمؤشر المركب بأن يغطي defect قاتلاً مثل authority مفقودة أو hash غير صالح أو citation ملفقة.

وظيفتها ليست السلبية؛ بل زيادة قابلية التعلم. فالنتيجة التي تنهار أمام حالة مضادة مبكرة أرخص من deployment واسع. ويُحفظ كل فشل كمادة لإعادة بناء المهارة والloop والسياسة.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «محكمة الهدم والفلسفة المضادة للطلاقة» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «محكمة الهدم والفلسفة المضادة للطلاقة» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «محكمة الهدم والفلسفة المضادة للطلاقة» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

23. المقاييس والمعادلة KIDS

يقترح الإطار KIDS - Knowledge Integrity Deployment Score: 0.22P + 0.18E + 0.14C + 0.12T + 0.12R + 0.10A + 0.12O - 0.20U، حيث P provenance، E evidence، C consistency، T tests، R rollback، A authority، O observability، U unresolved critical risk.

هذه الأوزان heuristic وليست حقيقة تجريبية، ولذلك تُصنف S حتى تُعاير على dataset من deployments ناجحة وفاشلة. كما أن المعادلة لا تلغي hard gates، فقد يحصل artifact على متوسط مرتفع مع signature فاشلة، ومع ذلك يجب حجبه.

ترافق KIDS مقاييس تفصيلية: rollback، route accuracy، stale retrieval rate، orphan node rate، unsupported claim rate، mean time to repair، status evidence coverage، readiness، وcost per verified result. القياس النافع يغير القرار ويكشف gaming.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest لـhash وspecification anchor لـE، وS لـrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «المقاييس والمعادلة KIDS» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «المقاييس والمعادلة KIDS» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «المقاييس والمعادلة KIDS» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

24. التكامل مع Ollama، Cursor، Codex، Hermes

يمكن نشر الحزمة في أنظمة متعددة عبر طبقة adapter لا تغير العقود. في Hermes تُحوّل مجلدات HyperSkills إلى external skill directories وتربط loops بالcron أو triggers مع tool filtering. في Codex/Cursor تُستخدم كتعليمات مشروع وscripts وفحوص CI. في Ollama تُبقي routing خارج weights وتستعمل النموذج منفذاً ضمن control plane.

الاختلافات البيئية تشمل paths وshell وpermissions وmodel context وtool APIs. لذلك لا يجوز إعلان التثبيت من حزمة عامة. يجب أن ينتج adapter تقرير target machine يحصي الملفات المحملة، الإصدارات، routes، canaries، وrollback.

تسمح هذه البنية بالاستقلال عن vendor، لأن core هو manifests وعقود وحالات واختبارات. لكنها لا تدعي أن كل منصة توفر نفس القدرات أو أن نموذجاً محلياً صغيراً يكتسبها بلا حدود.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «التكامل مع Hermes وCodex وCursor وOllama» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «التكامل مع Hermes وCodex وCursor وOllama» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «التكامل مع Hermes وCodex وCursor وOllama» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

25. دراسة حالة: إنتاج ورقة بحثية عربية

عند طلب ورقة عربية RTL يبدأ النظام بعقد الصفحات والخط واللغة والمؤلف والمصادر. يبحث prior art، ثم يبني claim ledger، ويفصل الإطار الأصلي عن المعايير الخارجية، ويكتب sections، ثم يولد DOCX/PDF، ثم يشغل render-inspect-revise.

بوابة النزاهة تشمل الاقتباسات، page count، Amiri، RTL، عدم clipping، وreferences. وجود PDF وحده لا يكفي؛ يجب فحص الصفحات المصورة. إذا نجحت الورقة وسُلّمت مستقلة تحصل على D، بينما النسخة داخل ZIP تحصل على E.

هذه الدراسة تبين أن status لا يساوي الجودة: ملف D قد يحتاج إصلاحاً، وملف S قد يكون specification ممتازة. القرار التشغيلي يجمع status مع quality gates دون خلط المفهومين.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـD، containment لـE، specification anchor لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «دراسة حالة: إنتاج ورقة بحثية عربية» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «دراسة حالة: إنتاج ورقة بحثية عربية» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «دراسة حالة: إنتاج ورقة بحثية عربية» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة، تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

26. دراسة حالة: ترقية **Knowledge Base** إلى **OS**

تبدأ الحالة بملفات معرفية متفرقة. Asset Discovery يحدد التكرار، Claim Extraction يبني Chunk Integrity، ledger يقسم الوحدات، GraphRAG Edge Validation يضبط العلاقات، ثم Knowledge Card يحول المعرفة إلى input/output/test/memory trigger.

لا تُحمّل القاعدة كاملة في السياق. يستخدم router استعلاماً يتضمن outcome وartifact وphase وhazard وgate، ويسترجع عدداً صغيراً من الوحدات الكاملة. وبعد الاستخدام تُقاس precision وcoverage والأثر، ويُحدّث status وmemory.

تترقى القاعدة إلى OS حين تستطيع توجيه قرار وتنفيذ واختبار وإعادة بناء، لا لمجرد زيادة الحجم. أما الملفات المطلوبة التي لم تُنتج فتبقى P، والمواصفات غير المادية S، والوحدات داخل الحزمة E.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـD، containment لـE، specification anchor لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «دراسة حالة: ترقية Knowledge Base إلى OS» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «دراسة حالة: ترقية Knowledge Base إلى OS» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «دراسة حالة: ترقية Knowledge Base إلى OS» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

27. دراسة حالة: **Browser Automation** محكومة

في مهمة browser، يُعامل محتوى الصفحة كـinput غير موثوق. يثبت task contract الموقع والحساب والهدف والمحظورات، ثم يختار profile بأدنى tools. تُفصل القراءة عن الكتابة، والتحضير عن submit، والملاحظة عن purchase.

تستخدم page-to-injection-check loops وrisk-to-canary وdeploy-to-verify. تُلتقط screenshots أو states حين تكون دليلاً، وتُحفظ timestamps وURLs والنتيجة. MFA والهوية والموافقة البشرية ليست عوائق ينبغي تجاوزها بل حدود سلطة.

إذا فشل selector أو تغيرت الواجهة، لا يكرر loop بلا حد. يقف عند stop condition، ويحفظ state وسبب الفشل، ويخفض status التشغيلي للـautomation إن لم يعد صالحاً.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «دراسة حالة: Browser Automation محكومة» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «دراسة حالة: Browser Automation محكومة» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «دراسة حالة: Browser Automation محكومة» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة؛ تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

28. القيود، القابلية للتفنيد، وبرنامج البحث

الإطار تركيبي ويحتاج إلى معايرة تجريبية. حالات D/E/S/P تصف الوجود ولا تقيس الصدق. الـhash لا يمنع المحتوى الخاطئ. provenance قد تكون صادقة عن سلسلة أنتجت تضليلاً. والـCards تعتمد على صدق المنتج ما لم توجد مراجعة مستقلة.

يمكن تفنيد الادعاءات عبر تجارب: مقارنة فرق تستخدم control plane بفرق لا تستخدمه، وقياس stale retrieval، fabricated citation، route cost، rollback success، وstatus error. كما يجب اختبار adversarial packages وmanifests المزورة وcross-platform deployment.

برنامج البحث التالي يشمل dataset لحوادث النزاهة، calibration لـbenchmark، KIDS للـstatus classification، دراسة human factors في dashboards، وتحليل trade-off بين provenance density وtoken cost. إلى أن تنجز هذه الأعمال تبقى المعادلات والعمارة PROPOSAL/S.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «القيود، القابلية للتفنيد، وبرنامج البحث» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

مصفوفة القرار في موضوع «القيود، القابلية للتفنيد، وبرنامج البحث» تربط أربعة مستويات لا يجوز دمجها: مستوى صحة الادعاء، ومستوى سلامة السلسلة، ومستوى أهلية الفاعل، ومستوى اكتمال artifact. يُعطى كل مستوى دليله المستقل، ثم تُحدد النتيجة التي ستتغير إذا فشل. فإذا كانت المعرفة صحيحة لكن path أو hash غير صالح، يُمنع E/D مع بقاء المحتوى قابلاً لأن يكون S. وإذا كان الملف سليماً لكن الاقتباس لا entail الادعاء، لا تسمح سلامة البايتات بالنشر المعرفي. وإذا كانت المواصفة كاملة من غير target-machine evidence، يبقى التنفيذ P. هذا الفصل يمنع metric substitution، ويجعل الإصلاح موجهاً: بحث جديد لفشل الدليل، إعادة build لفشل artifact، تعديل policy لفشل السلطة، أو إعادة routing لفشل التطبيق.

معيار القبول لهذه الوحدة هو أن يستطيع مراجع مستقل إعادة بناء قرار «القيود، القابلية للتفنيد، وبرنامج البحث» من السجل وحده: يرى المدخلات، ويصل إلى claims نفسها، ويتحقق من provenance، ويعيد تشغيل الاختبار، ويفهم لماذا اختيرت skill أو loop ولماذا توقفت. يجب أن يحدد التقرير baseline وtarget delta وfatal gates والتكلفة والنتيجة الخارجية. أما معيار الرفض فهو بقاء خطوة حاسمة ضمنية، أو اعتماد status على الذاكرة، أو غياب failure observation، أو عدم القدرة على الرجوع إلى الإصدار السابق. وعند الرفض لا تُحذف الوحدة، تُحفظ كـS أو P مع blocker وrepair owner وnext_check، فتتحول الفجوة إلى عمل محدد بدلاً من يقين مصطنع.

29. الخلاصة التنفيذية

تُحوّل النزاهة المعرفية من قيمة عامة إلى control plane قابل للنشر: object model، claim ledger، provenance graph، D/E/S/P machine، observability، rollback، hashes، manifests، canaries، hard gates، 50 HyperLoops، 70 HyperSkills، وmemory reconstruction.

المبدأ الأهم هو أن كل check يعيد بناء الحقيقة التشغيلية من الأدلة الحالية. لا يُستنتج التسليم أو التضمين أو التثبيت من الذاكرة أو النية. ويُقبل UNKNOWN كحالة معرفية، وP كحالة إنجاز، بدلاً من ملء الفراغ.

الحزمة المصاحبة تنفذ هذا المبدأ في ملفات قابلة للفحص. الورقة والـOS المستقلان D عند التسليم، المهارات والـloops داخل ZIP هي E، calibration محددة S، والتثبيت على جهاز المستخدم P حتى يظهر تقرير بيئة. بهذه الصراحة يصبح النظام قابلاً للتوسع من دون أن يفقد نسبه وحدوده.

التشغيل المقترح في هذه الطبقة يبدأ بتثبيت task_id وasset_id والإصدار والبيئة، ثم يربط كل قرار بـevidence_refs واختبارات واضحة. يُمنع الانتقال إلى النشر إذا غاب provenance أو authority أو rollback، وتُحفظ remaining_unknowns بوصفها جزءاً من الناتج. عند كل check تُعاد حالات D/E/S/P وفق الوجود المادي: standalone reference لـcontainment، D وmanifest وhash لـspecification anchor، E لـS، وrequest record بلا artifact لـP. ويُسجل أي انتقال في status history مع السبب والفاعل والزمن، بحيث يمكن لوكيل جديد استئناف المهمة من دون اختراع حالة سابقة.

محكمة الهدم الخاصة بهذه الصفحة تسأل: هل «الخلاصة التنفيذية» محدد كتحول يمكن قياسه أم مجرد وصف مرغوب؟ هل يمكن بناء حالة مضادة يظهر فيها النظام سليماً لغوياً لكنه يفشل مادياً؟ هل يوجد اختبار يميز هذا المسار عن baseline أبسط؟ وهل يؤدي فشل الاختبار إلى repair أو demotion أو refusal فعلي؟ لا تُقبل الإجابة إن كانت تعيد صياغة الهدف، بل يجب أن تُنتج artifact أو metric أو gate يغير قرار النشر أو الاسترجاع أو الذاكرة.

المراجع الأساسية

[1] W3C, PROV-Overview: An Overview of the PROV Family of Documents, 2013. https://www.w3.org/TR/prov-overview/

[2] NIST, Artificial Intelligence Risk Management Framework (AI RMF 1.0), 2023. https://www.nist.gov/itl/ai-risk-management-framework

[3] NIST, AI RMF: Generative Artificial Intelligence Profile, NIST.AI.600-1, 2024.

[4] C2PA, Content Credentials Technical Specification, Version 2.4. https://spec.c2pa.org/

[5] SLSA, Supply-chain Levels for Software Artifacts Specification, Version 1.2. https://slsa.dev/spec/v1.2/

[6] in-toto, A Framework to Secure the Integrity of Software Supply Chains. https://in-toto.io/

[7] Gebru et al., Datasheets for Datasets, Communications of the ACM / arXiv:1803.09010.

[8] Mitchell et al., Model Cards for Model Reporting, FAT* 2019 / arXiv:1810.03993.

[9] Pushkarna, Zaldivar, Kjartansson, Data Cards: Purposeful and Transparent Dataset Documentation, FAccT 2022.

[10] NIST, Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations, NIST.AI.100-2e2025.