عزل المستأجر

معمارية من 9 طبقات لبنية تحتية مشتركة

Hadda TIKIJJA

مختبر 0DATA، فرنسا

ملخص

لا يمكن لعزل المستأجر في بنية تحتية مشتركة أن يرتكز على طبقة واحدة. نوثّق معمارية العزل ذات **الطبقات التسع** أمان مستوى الصف في :DATAالمنشورة على البنية التحتية 0 PostgreSQL، لكل مستوى، وتطبيق ALFAلكل نطاق، و Cockpitلكل مستأجر، و NOVAلكل مستأجر، و SPINAو DNSالمُقسَّمة بمساحات أسماء، و NATS ويب تقدمي دون اتصال، وخطة المنافذ، وموضوعات wildcard. تعمل كل طبقة باستقلالية؛ فلا يُضعف تعطّل إحداها الأخريات. وتستلهم هذه المعمارية في الظهارة البيولوجية، حيث تحافظ كل خلية على غشائها الخاص مع مشاركتها **الوصلات المحكمة** في النسيج المشترك. نصف كل طبقة بالتفصيل، ونعرض خطة المنافذ الموثقة (لا تستمع أي خدمة على 0.0.0.0)، ونورد نتائج اختبار الاختراق عبر المستأجرين — إذ فشلت جميع محاولات القراءة بين المستأجرين.

في جملة واحدة

تضمن تسع طبقات عزل مستقلة ألا يتمكن مستأجر من رؤية بيانات مستأجر آخر أو الوصول إليها أو تعديلها — على أي مستوى من الحزمة.

1. لماذا تسع طبقات

1.1 الدفاع المتعمق ليس استعارة

في الأمن السيبراني الكلاسيكي، يُعد الدفاع المتعمق مبدأً نظريًا. أما في بنية تحتية مشتركة تستضيف عملاء ذوي مصالح متضاربة، فيغدو ضرورة حيوية. قد يستضيف مزوّد الخدمات المُدارة في آنٍ واحد مكتب MSP محاماة وعيادةً ومنافسًا مباشرًا للأول — على الخادم المادي نفسه.

يستند النهج التقليدي لتعدد المستأجرين إلى فصل على مستوى التطبيق: يصفّي التطبيق الاستعلامات وفقًا لـ `tenant_id` . ويفشل هذا النهج فشلاً ذريعًا عندما يتجاوز خطأ برمجي أو ثغرة حقن أو تصعيد امتيازات هذه الطبقة الوحيدة.

1.2 النموذج البيولوجي: الغشاء الخلوي والوصلات المحكمة

خاصًا بها — وهو الحاجز **غشاءً فوسفوليبيديًا** تجسّد الظهارة المعوية البشرية نموذجنا. تمتلك كل خلية حاجزًا ثانيًا، غير منفذ حتى للجزيئات (tight junctions) **الوصلات المحكمة** الأول. وبين الخلايا، تشكّل فيصفّي الجزيئات الكبيرة. لا يكفي أي من هذه الحواجز — **الغليكوكاليكس** الصغيرة. أما الحاجز الثالث وحده؛ فتراكبها يخلق إحكامًا يقارب المطلق.

النقل المحاكي للطبيعة

الغشاء الخلوي ← يمتلك كل مستأجر نسخه الخاصة من ALFA وNOVA وSPINA
الوصلات المحكمة ← يمنع RLS في PostgreSQL أي تسرب على مستوى قاعدة البيانات
الغليكوكاليكس ← تشكّل خطة المنافذ وDNS wildcard وNATS المُقسَّم بمساحات الأسماء طبقة ترشيح شبكية

1.3 الطبقات التسع

الطبقة	الاسم	المجال	الآلية
1	PostgreSQL RLS	قاعدة البيانات	سياسات أمان مستوى الصف
2	مخطط PostgreSQL	قاعدة البيانات	مخطط منطقي لكل مستأجر
3	سياسات التطبيق	قاعدة البيانات	قيود CHECK ومحفزات
4	SPINA v2	التوقيع	HMAC-SHA256 محلي + eIDAS RFC 3161

5	NOVA	المراقبة	نسخة لكل مستأجر، ربط 127.0.0.1
6	ALFA	القرار	محرك مشترك، ذاكرة معزولة
7	Cockpit	الواجهة	موقّع لكل مستأجر JWT، نطاق خاص
8	DNS / NATS	الاتصال	DNS wildcard وموضوعات مُقسَّمة
9	خطة المنافذ	الشبكة	لا خدمات على 0.0.0.0، منافذ موثقة

2. 3–1 الطبقات: PostgreSQL، الأساس

2.1 (1 الطبقة) أمان مستوى الصف

أعمق آليات العزل. يحمل كل صف في PostgreSQL (Row-Level Security) يُعد أمان مستوى الصف المتصلة PostgreSQL المطبقة على كل جدول ألا تتمكن جلسة RLS تضمن سياسة . `tenant_id` كل جدول للمستأجر أ من قراءة صفوف المستأجر ب ماديًا.

```
ALTER TABLE organisms ENABLE ROW LEVEL SECURITY;

CREATE POLICY tenant_isolation ON organisms
    USING (tenant_id = current_setting('app.current_tenant_id')::uuid);
```

الشكل 1: سياسة RLS في PostgreSQL تُقارن عمود tenant_id بمتغير الجلسة.

`current_setting` تقرأ الدالة .PostgreSQL متغير جلسة في PostgreSQL. يضبط هذا المتغير مجمّع الاتصالات لحظة مصادقة المستأجر. لا يمكن لأي استعلام تطبيقي تعديله — فهو للقراءة فقط من جانب التطبيق. وهذه الطبقة منيعة دون وصول مستخدم فائق (superuser) في PostgreSQL.

2.2 (2 الطبقة) مخطط لكل مستأجر

بمعنى PostgreSQL ليس مخطط — **مخططًا منطقيًا مخصصًا** يملك كل مستأجر، RLS إضافةً إلى بل إسقاط بيانات مقيد. تُصفّى العروض المُجسّدة والفهارس الجزئية والتسلسلات وفقًا لـ ، `CREATE SCHEMA` نفسه محمي بـ `tenants` RLS. لا يستطيع المستأجر تعداد المستأجرين الآخرين، لأن جدول . `tenant_id`

تمنع هذه الطبقة هجمات الاستدلال: حتى لو تمكّن مستأجر من قياس زمن تنفيذ استعلام (هجوم توقيت)، تضمن الفهارس الجزئية ألا تُمسح بيانات المستأجرين الآخرين أبدًا.

2.3 القيود والمحفزات (الطبقة 3)

لمنع كتابة بيانات تنتهك `BEFORE INSERT/UPDATE` ومحفزات `CHECK` الثالثة قيود PostgreSQL تستخدم طبقة العزل:

```
ALTER TABLE organisms ADD CONSTRAINT chk_tenant_owns
    CHECK (tenant_id IS NOT NULL
           AND tenant_id = current_setting('app.current_tenant_id')::uuid);
```

الشكل 2: قيد CHECK يمنع الكتابة عبر المستأجرين.

بعد الإدراج. وهذا التكرار مقصود: إذا عُطّلت سياسة `tenant_id` يقفل محفز إضافي أي محاولة لتعديل عمود RLS عن طريق الخطأ (أثناء صيانة مثلاً)، تحافظ القيود والمحفزات على العزل.

3. الطبقات 4–6: SPINA وNOVA وALFA

3.1 SPINA v2 — التوقيع لكل مستأجر (الطبقة 4)

SPINA (بروتوكول التوقيع الخاص بـ DATA. في الإصدار 0 (بروتوكول التوقيع لسلامة الأصول وعدم إنكارها) هو الثاني، يملك كل مستأجر:

- يُولَّد ويُخزَّن في مساحة المستأجر، **مفتاح HMAC-SHA256 محليًا**
- **شهادة eIDAS RFC 3161** للختم الزمني المؤهل
- مشفرًا بعبارة مرور مشتقة من سر المستأجر، **مخزن مفاتيح معزولاً**

الخاص به للخطر، تبقى HMAC لا يتشارك مستأجران أي مادة تشفيرية أبدًا. إذا عرّض المستأجر أ مفتاح فيُدار اتحاديًا على مستوى البنية التحتية، لكن رمز الطابع eIDAS تواقيع المستأجر ب صالحة. أما الختم الزمني الزمني يُخزَّن في مخطط المستأجر.

`spina_verify()` الخاص بالمستأجر. تستخدم الدالة PostgreSQL تُنفَّذ عملية التحقق من التوقيع في سياق `SET app.current_tenant_id` مع `SECURITY DEFINER` صفة صريح — ولا يمكنها التسرب خارج المستأجر.

3.2 NOVA — نسخة لكل مستأجر (الطبقة 5)

NOVA هو محرك مراقبة الشبكة. يملك كل مستأجر MSP **نسخة NOVA خاصة به**، تعمل كعملية مخصصة:

- ربط حصري على `127.0.0.1`، بمنفذ مميز لكل مستأجر

- قاعدة بيانات مسح محلية (SQLite أو مخطط PostgreSQL معزول)
- لا مشاركة في الذاكرة بين النسخ
- ملف إعدادات مميز (`/opt/nova/tenants/<tenant_id>/config.yaml`)

التأثير في نسخة (NOVA وبالتالي في الأهداف التي يمسحها) لا يستطيع مستأجر يتحكم في شبكته المحلية إلى `kill -9` الخاصة بمستأجر آخر. فالعمليات معزولة على مستوى نظام التشغيل — فإرسال NOVA معرّف عملية المستأجر أ لا يؤثر على المستأجر ب.

3.3 ALFA — (6 الطبقة) محرك مشترك، ذاكرة معزولة

محركًا ALFA يستخدم NOVA، هو محرك القرار المشترك. وخلافًا لـ (طبقة التحليلات للتقييم الجنائي) ALFA **عزل صارم للذاكرة** لكفاءة الموارد، لكن مع **مشتركًا**:

- يُقسَّم كل مستوى تحليل (مستأجر، موقع، جهاز) في حجيرات منفصلة
- تُحمَّل البيانات في أجزاء ذاكرة محكمة الإغلاق
- الخاص بالمستأجر PostgreSQL تُكتب نتائج التحليل في مخطط
- لا يُجرى أي ترابط بين المستأجرين دون موافقة صريحة

على المستوى الخوارزمي: لا يستقبل نموذج التحليل المنفَّذ للمستأجر أ **الحاجة إلى المعرفة** مبدأ ALFA يطبق سوى بيانات المستأجر أ. أما الإحصاءات المجمعة متعددة المستأجرين (المفيدة لقياس الأداء) فتستخدم بيانات مُجهَّلة الهوية مع خصوصية تفاضلية (ε = 1.0).

4. 9–7 الطبقات: Cockpit وDNS وNATS

4.1 Cockpit — لكل مستأجر (الطبقة 7) JWT نطاق خاص و

يصل كل مستأجر إلى واجهة Cockpit عبر **نطاق مخصص**: `client.odata.fr` . تُصدَر شهادة SSL (Let's Encrypt) للمصادقة بمفتاح سري خاص بالمستأجر، ويحوي JWT لهذا النطاق. يُوقَّع رمز `tenant_id` ضمن مطالباته (claims).

```
Authorization: Bearer eyJ...tenant_id:"a1b2c3d4-..."...
```

الشكل 3: ترويسة مصادقة Cockpit ǻ JWT يحوي tenant_id.

يتحقق وسيط nginx من JWT قبل توجيه الطلب. تُرفض محاولة الوصول إلى `client-b.odata.fr` أصلًا API للمستأجر أ على مستوى الوكيل العكسي — قبل أن يصل الطلب إلى واجهة JWT باستخدام. ويشكّل النطاق نفسه حدودًا للعزل: لا يستطيع مستأجر تخمين النطاق الفرعي لمستأجر آخر (فالأسماء هي UUIDs أو معرّفات مبهمة).

4.2 DNS wildcard وNATS (الطبقة 8) المُقسَّم بمساحات الأسماء

تعمل الطبقة 8 على مستوى الاتصال بين الخدمات:

DNS wildcard. يشير السجل `*.odata.fr` إلى البنية التحتية. يُحل كل نطاق فرعي، لكن خادم التطبيق خاضع DNS يصفّي النطاقات غير المسجلة. لا يستطيع مستأجر إنشاء نطاق فرعي اعتباطي — فسجل لسيطرة البنية التحتية.

NATS المُقسَّم بمساحات الأسماء. يستخدم NATS ناقل رسائل موضوعات مسبوقة ببادئة المستأجر:

```
tenant.a1b2c3d4.nova.scan.start
tenant.a1b2c3d4.spina.sign
tenant.e5f6g7h8.nova.scan.start
```

الشكل 4: موضوعات NATS تقيّد قوائم ACL ببادئة المستأجر. الوصول حسب البادئة.

بحيث لا تتمكن اعتمادات المستأجر أ من النشر أو الاشتراك سوى في NATS الخاصة بـ ACL تُضبط قوائم هذه القوائم على مستوى البروتوكول — فلا يستطيع NATS يطبّق خادم `tenant.a1b2c3d4.*` الموضوعات أي عميل تجاوز تقسيم مساحات الأسماء.

4.3 خطة المنافذ — لا خدمات على 0.0.0.0 (الطبقة 9)

ترتبط كل خدمة `0.0.0.0`. الطبقة 9 هي الأكثر مادية: خطة المنافذ الموثقة. لا تستمع أي خدمة داخلية على كل حركة المرور الواردة (سياسة الرفض الافتراضي، UFW) بمنفذ صريح. يمنع الجدار الناري `127.0.0.1` بـ غير المصرح بها صراحةً. وتُوثَّق هذه الطبقة بالتفصيل في القسم 5.

5. خطة المنافذ الموثقة

إلى واجهة **الربط (Bind)** يوثق الجدول التالي جميع المنافذ المستمعة في البنية التحتية. يشير عمود باستثناء المنافذ العامة المعرَّضة عمدًا، **لا تستمع أي خدمة على 0.0.0.0**. الاستماع (HTTP/S وSSH وSIP).

5.1 المنافذ العامة (المعرَّضة)

المنفذ	البروتوكول	الخدمة	المبرر
22	TCP	SSH	وصول إداري
80	TCP	HTTP (nginx)	إعادة توجيه HTTPS + ACME
443	TCP/UDP	HTTPS (nginx)	وكيل عكسي، HTTP/3 QUIC
5060	UDP	SIP (FreeSWITCH)	اتصالات VoIP
5080	UDP	SIP (FreeSWITCH)	اتصالات VoIP

5.2 المنافذ الداخلية (ربط 127.0.0.1)

المنفذ	الخدمة	المستأجر	الدور
5432	PostgreSQL	مشترك (RLS)	قاعدة البيانات الرئيسية
6379	Redis	مشترك (ببادئة)	الذاكرة المؤقتة والجلسات
4222	NATS	مشترك (بمساحات أسماء)	ناقل الرسائل
5090–5099	نسخ NOVA	لكل مستأجر	مسح الشبكة
5100–5109	واجهة NOVA API	لكل مستأجر	واجهة REST لـ NOVA
5110–5119	عمال SPINA	لكل مستأجر	التوقيع/الختم الزمني
8300	واجهة API الرئيسية	مشترك (RLS)	واجهة FastAPI
8400–8409	عمال Cockpit	لكل مستأجر	واجهة الويب Cockpit
9090	Cockpit النظام	البنية التحتية	إدارة النظام

5.3 قواعد UFW

```
ufw default deny incoming
ufw default allow outgoing
ufw allow 22/tcp
ufw allow 80/tcp
ufw allow 443
ufw allow 5060/udp
ufw allow 5080/udp
```

الشكل 5: قواعد UFW — 5 منافذ فقط معرَّضة للعموم.

حاجزًا ثانيًا: حتى لو عُطّل `127.0.0.1` تُحجب جميع المنافذ الأخرى على مستوى الجدار الناري. يضيف الربط الجدار الناري، لما أمكن الوصول إلى الخدمات الداخلية من الخارج.

6. التحقق — اختبار اختراق عبر المستأجرين

6.1 بروتوكول الاختبار

أُجري اختبار الاختراق عبر المستأجرين في 23 يوليو 2026. أُنشئ مستأجران للاختبار:

- **المستأجر أ** (`a1b2c3d4-...`): «بيانات اختبار عيادة بيطرية»
- **المستأجر ب** (`e5f6g7h8-...`): «بيانات اختبار مكتب محاسبة»

نُفِّذت الاختبارات من سياق المستأجر أ، في محاولة للوصول إلى بيانات المستأجر ب. استخدمت جميع الاختبارات اعتمادات صالحة للمستأجر أ.

6.2 النواقل المُختبَرة

#	الناقل	الطريقة	النتيجة
1	قراءة SQL مباشرة	`SELECT * FROM organisms` (دون مرشح tenant_id)	**فشل** — يعيد RLS صفر صفوف
2	تعديل tenant_id	`UPDATE organisms SET tenant_id = 'e5f6...'`	**فشل** — يرفض قيد CHECK
3	وصول API عبر المستأجرين	`GET /api/organisms?tenant_id=e5f6...`	**فشل** — المعامل API تتجاهل وتستخدم JWT

4	JWT مزوَّر	JWT مع `tenant_id: "e5f6..."` موقّع بمفتاح أ	**فشل** — توقيع غير صالح
5	وصول Cockpit عبر النطاق ب	الاتصال بـ `tenant-b.odata.fr` بJWT للمستأجر أ	**فشل** — عدم تطابق النطاق + JWT، 403
6	اشتراك NATS	`nats.sub("tenant.e5f6.*")` باعتمادات أ	**فشل** — ترفض قائمة ACL الاشتراك
7	عملية NOVA	الاتصال بمنفذ NOVA للمستأجر ب	**فشل** — ربط 127.0.0.1، والجدار الناري يمنع
8	مفتاح SPINA	محاولة تحقق بمفتاح المستأجر ب	**فشل** — عدم تطابق HMAC
9	تعداد DNS	محاولة اكتشاف النطاقات الفرعية	**فشل** — UUIDs مبهمة، ولا نقل منطقة

6.3 النتيجة

نتيجة اختبار الاختراق عبر المستأجرين

صفر نجاحات عبر تسعة نواقل. حجبت الطبقتان 1 (RLS) حجبت كل طبقة المحاولات التي تعنيها. وحجبت الطبقة 8 (SPINA) التزوير التشفيري (NATS). وحجبت الطبقة 4 SQL. و2 (القيود) محاولات اعتراض الرسائل. وحجبت الطبقة 9 (الجدار الناري + الربط) الوصول الشبكي المباشر.

لم تُتجاوز أي طبقة. ولم تمكّن أي تركيبة من النواقل من استدلال بيانات بين المستأجرين.

7. لماذا تعدد المستأجرين الكلاسيكي غير كافٍ

7.1 نموذج الطبقة الواحدة

تعتمد معمارية تعدد المستأجرين الكلاسيكية — تلك التي تعتمدها معظم تطبيقات SaaS — على فصل تطبيقي واحد. يضيف التطبيق `WHERE tenant_id = ?` إلى كل استعلام. يعاني هذا النهج من ثلاثة عيوب بنيوية:

1. **نقطة فشل واحدة.** إن نسيان عبارة WHERE في عملية ربط، أو ترحيل مخطط لم يُختبر جيدًا، أو خطأ في ORM يعرّض بيانات جميع المستأجرين.
2. **لا دفاع ضد تصعيد الامتيازات.** إن المهاجم الذي يحصل على وصول مدير تطبيقي (عبر حقن SQL أو حشو الاعتمادات أو ثغرة في التبعيات) يتجاوز العزل بالكامل.

3. **لا عزل على مستوى البنية التحتية.** قاعدة البيانات والذاكرة المؤقتة وناقل الرسائل — كلها مكونات مشتركة دون حاجز داخلي. فتسريب ذاكرة في Redis يعرّض جلسات جميع المستأجرين.

7.2 0 مقارنة مع نهج DATA

البُعد	تعدد المستأجرين الكلاسيكي	0 عزلDATA (9 طبقات)
قاعدة البيانات	`WHERE tenant_id = ?`	RLS + محفزات + قيود
التوقيع	مشترك	HMAC لكل مستأجر + eIDAS
المراقبة	مشتركة	نسخة NOVA لكل مستأجر
التحليل	مشترك	ALFA بذاكرة معزولة
الواجهة	النطاق نفسه	مخصص JWT، نطاق لكل مستأجر
الرسائل	موضوعات مشتركة	NATS مُقسَّم بمساحات أسماء مع ACL
الشبكة	خدمات على 0.0.0.0	ربط 127.0.0.1 وجدار ناري
العمق	طبقة واحدة	9 طبقات مستقلة

7.3 كلفة العزل

وصيانة، (لكل مستأجر NOVA نسخة) للعزل ذي الطبقات التسع كلفة: تعقيد النشر، واستهلاك ذاكرة إضافي حيث يمكن، MSP ففي سياق **استثمارًا بنيويًا**. تُقبل هذه الكلفة بوصفها NATS. الخاصة بـ ACL قوائم وخسارة العملاء، تكون كلفة العزل RGPD لتسريب بيانات عبر المستأجرين أن يجرّ ملاحقات قضائية وعقوبات أدنى من كلفة الانتهاك.

ومختبَرة (اختبار اختراق عبر المستأجرين عند كل نشر) (النشر عبر Ansible) المعمارية موثقة ومؤتمتة. ويُضبط التعقيد عبر قابلية التكرار.

8. أعمال ذات صلة وآفاق

يُعد عزل المستأجر في البنى التحتية المشتركة موضوعًا نشطًا في الأدبيات. تستخدم المعماريات من نوع المحاكاة الافتراضية العتادية لعزل المستأجرين — وهو (Google Borg وAWS Nitro Enclaves) «القفص» (كل مستأجر على مجموعته الخاصة) «shared-nothing» نهج متين لكنه مكلف الموارد. أما المعماريات فتقدم عزلًا مثاليًا مقابل عدم كفاءة اقتصادية.

يحتل نهج DATA 0 نقطة توازن: **مشاركة البنية التحتية مع عزل البيانات**. يوجه النموذج البيولوجي للوصلات المحكمة هذه المعمارية — فكل طبقة مستقلة ومتكررة وقابلة للتحقق.

تشمل الأعمال المستقبلية:

- **التدوير التلقائي لمفاتيح SPINA لكل مستأجر** (دورية قابلة للضبط)
- **خصوصية تفاضلية متعددة المستأجرين** للتجميعات الإحصائية
- **عزل عتادي** (SGX/TDX) للمستأجرين ذوي المتطلبات التنظيمية العالية
- **تدقيق مستمر عبر المستأجرين** — اختبار تلقائي للطبقات التسع عند كل نشر

المراجع

[1] TIKIJJA, Hadda. «0 مختبر DATA الانضباط». البحث 001، يوليو 2026.

[2] TIKIJJA, Hadda. «0 مختبر DATA الجهاز العصبي». البحث 003، يوليو 2026.

[3] TIKIJJA, Hadda. «0 مختبر DATA الطُعم الرقمي». البحث 004، يوليو 2026.

[4] TIKIJJA, Hadda. «0 مختبر DATA الجهاز المناعي». البحث 005، يوليو 2026.

[5] TIKIJJA, Hadda. «0 مختبر DATA تدقيق السطح». البحث 014، يوليو 2026.

[6] PostgreSQL Documentation. «Row Security Policies». PostgreSQL 16، 2024.

[7] NATS Documentation. «Subject-Based Messaging and ACLs». Synadia Communications، 2024.

[8] eIDAS Regulation (EU) No 910/2014. «Electronic Identification and Trust Services».

[9] RFC 3161. «Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)».

[10] Dwork, C., Roth, A. «The Algorithmic Foundations of Differential Privacy». Foundations and Trends in Theoretical Computer Science، 2014.

شكر وتقدير. إلى فريق PostgreSQL على تقسيم مساحات NATS على أمان مستوى الصف، وإلى مشرفي البنية التحتية 0 DATA، التي صمدت أمام اختبار الاختراق عبر المستأجرين دون أن تتزعزع. العزل ليس ميزة — إنه بنية الكائن الحي ذاتها.