

 [LUISTER DIT VERSLAG](#)

-10%

1,0x

+10%

+25%

1,0x

↓ [Download naar telefoon](#)

▶  [Lees mee met de podcast](#)

Zestien agents lazen 320 bestanden in vijf repo's. Elke melding is daarna door een aparte scepticus nagetrokken, met de opdracht hem onderuit te halen. Wat je hieronder leest, is wat die controle overleefde: 96 van de 104. De 8 die afvielen staan onderaan.

96 echte problemen bevestigd, verdeeld over 5 repo's, na het lezen van 320 bestanden.

8 meldingen vielen af bij de controle, omdat ze bij natrekken niet klopten.

Het ergste: er staan twee geheimen in git (een sleutel voor het gedeelde brein en een GitHub-token), en de beveiligingsscaner doet echte inlogpogingen op je live websites.

Wat je nu moet doen

1. Sleutel voor het gedeelde brein staat in git

Wat gaat mis: in de backup-repo van de Lenovo-bot staat een toegangssleutel voor de gbrain-server gewoon in het bestand. Iedereen die die repo kan lezen, of een oude kopie heeft, kan het gedeelde brein uitlezen en beschrijven. Ook als je de regel nu weghaalt, blijft de sleutel in de git-geschiedenis staan.

Waar: `lenovo-bot/.mcp.json`, regel 11. Type geheim: een bearer-token voor de gbrain-API.

Fix: sleutel intrekken en vervangen. Bestand uit git halen en in `.gitignore` zetten. Sleutel via een omgevingsvariabele laden. Git-geschiedenis opschonen.

2. GitHub-token in platte tekst in de security-repo

Wat gaat mis: in `haarvisie-security-agent/.git/config` staat een GitHub personal access token gewoon leesbaar in de remote-URL. Dit is niet gemeld door de scanner zelf, het

kwam bij de controle boven water.

Waar: `.git/config` van de repo haarvisie-security-agent.

Fix: token intrekken op GitHub, opnieuw aanmaken, en de remote instellen via de credential helper of SSH in plaats van in de URL.

3. De scanner doet echte inlogpogingen op haarvisie.nl en haartips.nl

Wat gaat mis: bij elke scan worden er drie foute logins verstuurd naar `wp-login.php` van je live sites, met gebruikersnaam admin. Dat is een echte brute-force-poging, geen passieve check. Een beveiligingsplugin of fail2ban kan daardoor jouw eigen IP blokkeren of het admin-account op slot zetten. Het rapport van 28-7 bewijst dat dit echt is uitgevoerd op beide sites. Extra bezwaar: haarvisie.nl mag volgens het regelboek alleen gelezen worden.

Waar: `checkers/external/wordpress_checker.py`, regel 51 tot 57.

Fix: de actieve inlogpogingen weghalen. Alleen passief kijken, of dit uitsluitend op een testomgeving draaien.

4. Het rapport met al je zwakke plekken staat in git

Wat gaat mis: `laatste-rapport.md` staat wel in `.gitignore`, maar was al eerder toegevoegd aan git. Daardoor gaat hij bij elke commit gewoon mee. In dat rapport staat welke paden op je sites open staan, welke WordPress-versie draait en welke beveiligingsheaders ontbreken. Er staat al zo'n rapport in de geschiedenis van twee commits.

Waar: `laatste-rapport.md`, repo haarvisie-security-agent.

Fix: `git rm --cached laatste-rapport.md`, geschiedenis opschonen, en rapporten voortaan buiten de repo wegschrijven.

5. Klant tikt op het overzicht en het formulier wordt blanco

Wat gaat mis: op de laatste stap van het afspraakformulier kan de klant op een blok tikken om iets te wijzigen. Twee van die blokken verwijzen naar stap 2 en stap 3. Die stappen bestaan sinds de samenvoeging van 20-7 niet meer. De code verbergt dan alle stappen en vindt niets om te tonen. De klant ziet een lege pagina met "Stap 0 van 6" en geen enkele knop. Alleen verversen helpt, en de hele aanvraag is weg.

Waar: `app/public/afspraak-logica.js`, regel 801 en 810.

Fix: die twee blokken naar stap 1 laten verwijzen. En als vangnet: als het doelscherm niet bestaat, niets verbergen en blijven staan.

6. De opmaak van het afspraakformulier werkt niet

Wat gaat mis: alle kleuren, hoeken en lettertypes van het formulier hangen aan een naam die nergens in de pagina voorkomt. De code zoekt een id, de pagina heeft alleen een class. Gevolg: de keuzeknoppen krijgen geen witte achtergrond maar worden doorzichtig op de creme pagina. Hetzelfde geldt voor de grijze tekstkleur (24 keer gebruikt), de afgeronde hoeken (7 keer) en het lettertype (7 keer). Dit is nagemeten in de gebouwde versie: 17 treffers op de verkeerde naam.

Waar: `app/src/afspraak-form.css` , regel 10.

Fix: overal de id-verwijzing vervangen door de class, of de kleuren en fonts één keer bovenaan de site zetten.

7. De Lenovo-bot voert opdrachten uit de wachtrij blind uit

Wat gaat mis: tekst uit de wachtrij gaat letterlijk als opdracht naar Claude, die draait met alle toestemmingen vooraf goedgekeurd. Telegram is wel afgeschermd op jouw chat, de wachtrij niet. Er is geen lijst van wie er mag sturen, en de tekst wordt niet gemarkeerd als "dit is data, geen bevel". Stuurt een andere bot de tekst van een binnengekomen mail door, en staat daar toevallig "verwijder map X" of "stuur dit adres een mail" in, dan voert Lenovo dat gewoon uit.

Waar: `lenovo-bot/lenovo-bot.js` , regel 186 tot 190, met de instelling op regel 77.

Fix: alleen opdrachten van bekende bots accepteren, de opdrachttekst inpakken in tags met de uitleg dat het data is, en de "sla alle toestemmingen over"-vlag vervangen door een beperkte lijst toegestane acties.

8. Lenovo zet opdrachten van andere bots terug op de rij

Wat gaat mis: de opruimactie kijkt alleen of een opdracht langer dan 10 minuten in behandeling is. Er wordt niet gekeken van wie die opdracht is. Lenovo grijpt dus ook in de rijen van Sjakie, Coco, Paco en Conductor. Duurt een klus van Sjakie 25 minuten, dan zet Lenovo hem na 10 minuten terug en pakt een tweede bot dezelfde klus op. De klus wordt dan twee keer gedaan, bijvoorbeeld twee keer dezelfde concept-mail.

Waar: `lenovo-bot/lenovo-bot.js` , regel 158.

Fix: twee filters toevoegen zodat alleen de eigen claims worden teruggezet.

9. Opdrachten verdwijnen stil bij de queue-worker

Wat gaat mis: de rij wordt op "klaar" gezet vóórdát het werk begint. Loopt de opdracht daarna vast of tegen de tijdslimiet van 60 minuten aan, dan staat hij al afgevinkt. Niemand pakt hem nog op en er komt geen antwoord terug. De opdracht is gewoon weg.

Waar: `lenovo-bot/queue-worker.js` , regel 199 (het werk start pas op regel 223).

Fix: hetzelfde claim-patroon gebruiken als in `lenovo-bot.js`: eerst op "in behandeling", pas na een gelukt antwoord op "klaar".

10. Concept met "bounce" in het onderwerp krijgt stiekem een externe ontvanger

Wat gaat mis: het script zoekt op "oud adres in de aan-regel OF het woord bounce in het onderwerp". Bij een treffer op alleen het onderwerp wordt er niets verwijderd, maar wel een externe ontvanger toegevoegd, en het concept wordt opgeslagen. Ligt er een interne notitie met onderwerp "Bounce-analyse voor Jamal", dan krijgt die er een externe ontvanger bij. Verstuur je dat concept later, dan gaat interne informatie naar buiten.

Waar: `lenovo-bot/update-draft-gerrie.py` , regel 35.

Fix: alleen matchen op het oude adres in de ontvangers, en niets opslaan als er geen ontvanger verwijderd is.

11. De beatdetectie van de video-tool werkt bij geen enkel bestand

Wat gaat mis: de audio wordt door een pijp gelezen, en dan klopt de lengte in de bestandskop niet. De code vertrouwt op die lengte en botst daardoor altijd. Zelf nagedraaid op een gewoon geluidsbestand van 3 seconden: foutmelding. Dit is de kernfunctie voor muziekvideo-montage en hij faalt altijd.

Waar: `resolve_pilot/analyzers/beats.py` , regel 43.

Fix: het aantal samples uit de echte buffer rekenen, of eerst naar een tijdelijk bestand decoderen in plaats van naar een pijp.

12. Drie plekken in de video-tool waar vreemde tekst code kan worden

Wat gaat mis: menu-namen, bestandsnamen en notities worden zonder controle in scripts geplakt. Wie die tekst aanlevert, kan het script afsluiten en eigen regels toevoegen, die

vervolgens shell-commando's kunnen draaien. Getest en bevestigd: een geprepareerde menunaam levert een geldig script op met een shell-commando erin. Ook de Lua-fragmenten die jij zelf in de Resolve-console plakt, kunnen zo code bevatten die je niet ziet.

Waar: `resolve_pilot/ui_automation/applescript.py` regel 125,
`resolve_pilot/lua_snippets/library.py` regel 53, en
`resolve_pilot/lua_snippets/bridge.lua` (die voert 30 minuten lang elk bestand uit dat in /tmp verschijnt, zonder enige controle).

Fix: alle tekst netjes ontsnappen voor hij in een script gaat. De bridge naar een map in je eigen home met beperkte rechten verplaatsen, met een gedeeld wachtwoord erbij. Let op: de bridge wordt nergens door de tool aangeroepen, dus die kun je ook gewoon weggooien.

13. De kostenrem van gbrain werkt niet bij de gebruikelijke schrijfwijze

Wat gaat mis: schrijf je `--max-usd=5` met een isgelijktteken, dan wordt de vlag niet herkend en staat de uitgavenrem helemaal uit. Geen waarschuwing, geen foutmelding. Draait dat in een geplande taak, dan loopt hij onbeperkt door op betaalde API-calls. Hetzelfde geldt voor `--target-score=` en `--max-jobs=`. In 21 bestanden zit zo'n eigen vlaggenlezer die alleen de spatie-vorm kent.

Waar: `src/commands/doctor.ts`, regel 7934.

Fix: één gedeelde lezer die beide schrijfwijzen aankan, en die hard stopt bij een onbekende waarde in plaats van stil door te gaan.

14. De gezondheidscheck van gbrain liegt als er iets kapot is

Wat gaat mis: de check bestaat uit één functie van ruim 3000 regels met 193 losse controles en 73 plekken waar een fout stil wordt weggegooid. Struikelt een controle, bijvoorbeeld omdat een tabel ontbreekt, dan verdwijnt hij uit de lijst. Hij kan dan ook geen strafpunten meer opleveren. Hoe meer er kapot is, hoe gezonder het rapport eruitziet.

Waar: `src/commands/doctor.ts`, regel 4161.

Fix: elke controle door één runner laten lopen die bij een fout een waarschuwing toevoegt in plaats van niets.

15. De webhook van gbrain schrijft buiten de beveiligde poort om

Wat gaat mis: de route `POST /ingest` leest de bestemming uit de kopregels van het verzoek en schrijft rechtstreeks weg. Twee dingen gaan daardoor mis. Eén: markeringen die alleen door vertrouwde schrijvers gezet mogen worden (zoals "verberg deze pagina uit alle zoekresultaten") worden niet weggehaald, dus een externe partij kan pagina's onzichtbaar maken. Twee: de bron waaraan het toegangstoken gekoppeld is wordt genegeerd, alles belandt in de standaardbron. Een client die alleen bij "team-b" mag, schrijft dus overal. Het antwoord meldt intussen een bron die nooit gebruikt is.

Waar: `src/commands/serve-http.ts` regel 1888 tot 1897, en `src/core/minions/handlers/ingest-capture.ts` regel 116.

Fix: dezelfde vlag en dezelfde bron meegeven als de reguliere schrijfroute dat doet.

16. Bestands-functies van gbrain werken niet op een lokaal brein

Wat gaat mis: drie functies (bestanden opsommen, uploaden, link opvragen) gebruiken de oude vaste databaseverbinding in plaats van de meegegeven engine. Op een brein dat met de aanbevolen lokale variant is aangemaakt, geven ze allemaal een fout die zegt dat de database niet is ingericht. Nagemeten en bevestigd. Een subagent die die melding krijgt, concludeert dat het brein stuk is terwijl het gewoon werkt.

Waar: `src/core/operations.ts` , regel 2675, 2726 en 2777.

Fix: dezelfde engine-aanpak gebruiken als de commandoregel-versie al doet.

17. Twee database-motoren van 6000 regels die uit de pas lopen

Wat gaat mis: er zijn twee bijna identieke bestanden met dezelfde 143 functies. Elke fix moet twee keer, en niets waarschuwt als dat vergeten wordt. Dat is al gebeurd: één filter (zoeken op paginatype) staat wel in de ene motor en niet in de andere. Op een lokaal brein komen er dan ook andere paginasoorten terug dan gevraagd.

Waar: `src/core/pglite-engine.ts` regel 228 en 1749, tegenover `src/core/postgres-engine.ts` .

Fix: de gedeelde SQL in één basisklasse zetten, en een test toevoegen die dezelfde vragen aan beide motoren stelt.

18. Een timeout in gbrain doodt taken die nog maar net gestart zijn

Wat gaat mis: de bewaking rekent de looptijd vanaf de allereerste poging, inclusief alle wachttijd tussen herhalingen. Na een paar mislukte pogingen wordt de nieuwe, net gestarte poging meteen als dood gemarkeerd. De taak krijgt zo nooit zijn volle looptijd. Kanttekening: de standaarddrempel is ruimer dan gemeld, dus de ernst valt in de praktijk mee.

Waar: `src/core/minions/queue.ts` , regel 742.

Fix: bij elke nieuwe poging de starttijd opnieuw zetten, of een aparte kolom per poging gebruiken.

19. Drie checks van de beveiligingsscanter kunnen nooit iets vinden

Wat gaat mis: dit is het gevaarlijkste soort fout, want alles ziet er groen uit.

- De check op gelekte gebruikersnamen zoekt naar een veldnaam die WordPress nooit teruggeeft. Op haartips.nl staan die gebruikersnamen wel degelijk open, maar de melding komt niet.
- Mislukt het downloaden van een scriptbestand, dan wordt dat stil genegeerd en telt het toch mee als gescand. Zit de gelekte sleutel net in dat bestand, dan zegt het rapport "niets gevonden".
- Er worden maar tien scripts per site bekeken, zonder te melden dat er is afgekapt.

Waar: `checkers/external/wordpress_checker.py` regel 85,
`checkers/external/bundle_scanner.py` regel 54 en 94.

Fix: zoeken op de velden die WordPress echt teruggeeft. Downloadfouten als eigen bevinding melden. En in het rapport zetten: zoveel gevonden, zoveel echt gescand.

20. De scanner slaat vals alarm bij een nette weigering

Wat gaat mis: de scanner test of een chatbot te misleiden is. Hij ziet het als geslaagde hack zodra het antwoord de gewone woorden "instructions" of "system prompt" bevat. Antwoordt de bot netjes met "ik kan mijn instructies niet negeren", dan komt er een kritiek alarm terwijl er niets mis is. Precies de valse melding waar het commentaar in een ander bestand al voor waarschuwt.

Waar: `checkers/external/injection_tester.py` , regel 21.

Fix: alleen alarm slaan op unieke controlewoorden, en eerst kijken of het antwoord de vraag niet gewoon terugkaatst.

Per repo, de rest

haarvisie-nieuw (nieuwe website met afspraakformulier)

93 bestanden gelezen, 25 bevindingen, allemaal bevestigd. Naast de twee zware hierboven:

- **Gewiste voorkeursdatum wordt toch verstuurd.** Haalt de klant datum 2 weg, dan blijft de oude datum in het geheugen staan en gaat hij alsnog mee naar de salon. Jullie plannen dan een dag in die de klant juist had geschrapt. (`afspraak-logica.js` regel 741)
- **"Geen voorkeur" kent de woensdagsluiting van Wateringen niet.** Kiest een klant "geen voorkeur" plus twee stylisten uit Wateringen, dan wordt een woensdag gewoon goedgekeurd. Jullie moeten terugbellen om te verzetten. (regel 602)
- **De openingsdagen staan op twee plekken los van elkaar.** Zet je Wateringen op woensdag open in de homepage-tabel, dan blijft het formulier die dag weigeren. De site belooft dan iets wat het formulier blokkeert. (`routes/index.tsx` regel 78)
- **De spamcheck van het formulier is leeg.** De code zoekt naar een verborgen lokveld, maar dat veld staat niet in de pagina. De bescherming staat op papier aan en is in werkelijkheid uit. (regel 925)
- **De beveiligingsheaders zijn geschreven maar nooit aangezet.** Er ligt een keurig bestand met alle instellingen klaar, maar niets roept het aan. Wie de code leest denkt dat het aanstaat. (`lib/security-headers.server.ts` regel 5, drie keer apart gemeld)
- **Contactgegevens blijven 30 dagen op de computer staan en komen automatisch terug.** Vult iemand het formulier in op een gedeelde tablet, dan ziet de volgende bezoeker naam, e-mail, telefoonnummer en opmerkingen van de vorige persoon terug in beeld. (`afspraak-hulp.js` regel 265)
- **De scripts draaien opnieuw bij elke terugkeer naar /afspraak.** Navigeert een bezoeker weg en terug, dan zijn er twee kopieën actief en klappen er secties open die nog niet aan de beurt zijn. (`routes/afspraak.tsx` regel 22)
- **De sitemap bevat alleen de homepage.** Je twintig blogpagina's en de afspraakpagina staan er niet in, dus Google pakt ze trager of niet op. (`routes/sitemap[.].xml.ts` regel 8)
- **De site stuurt nog "Higgsfield" mee als auteur en X-account.** Deelt iemand de site, dan staat er een vreemd account onder. (`routes/___root.tsx` regel 81)
- **99 bestanden templatecode die de site nooit gebruikt** (ruim 11.000 regels). Wie "de knop van de site" zoekt, vindt drie kandidaten en past de verkeerde aan. (`src/layouts/` ,

src/components/ui/)

- Verder: dubbele controles op e-mail en telefoon in twee bestanden, dubbele datumberekening, dubbele Google Fonts-blokken, dode auto-doorschakelcode die een nieuwe knop ongevraagd kan laten doorspringen, en een klein probleem waarbij een fout na een geslaagde verzending toch als mislukt wordt getoond (risico op dubbele aanvraag).

gbrain (het gedeelde geheugen)

158 bestanden gelezen, 28 meldingen, 26 bevestigd, 2 afgevallen. Naast de zware hierboven:

- **Tijdzones met een half uur verschil worden verkeerd doorgegeven** aan de agent. India, Adelaide en Nepal krijgen een tijd die 30 of 45 minuten fout is. (`context-engine.ts` regel 345)
- **Een handtekeningcontrole kan crashen** in plaats van netjes te weigeren. De beller krijgt dan een serverfout en er komt een foutspoor in het log. (`timing-safe.ts` regel 33)
- **Het wachtwoord in een database-adres wordt niet altijd verborgen**. Bij de ene schrijfwijze wel, bij de andere niet, en dan staat het wachtwoord gewoon in je terminal en in de logs. (`commands/config.ts` regel 13)
- **De verkeerde standaard-embedmodel-naam** zorgt dat de tool je adviseert over te stappen op een model waar je al op zit. (`core/embedding.ts` regel 118)
- **De snelheidsrem op de tokenroute telt dubbel**, dus je limiet is in de praktijk de helft. (`serve-http.ts` regel 621)
- **Een oude, uitgeschakelde netwerkroute controleert helemaal geen rechten**. Zet iemand hem ooit weer aan, dan kan elk token gevoelige acties uitvoeren. Nu slapend en bewust op een uitzonderingslijst gezet. (`mcp/http-transport.ts` regel 151 en 382)
- **Een voortgangsmelding zonder foutopvang kan het hele werkproces afsluiten** midden in een lange klus, terwijl het verliezen van één voortgangsregel onschadelijk was geweest. (`handlers/embed-backfill.ts` regel 180)
- **De configuratie wordt 176 keer opnieuw van schijf gelezen** en niet veilig weggeschreven. Wordt er net geschreven terwijl er gelezen wordt, dan kan een upload wel als "gelukt" gemeld worden terwijl het bestand alleen als naam bestaat, zonder inhoud. (`mcp/dispatch.ts` regel 202)
- Verder veel dubbele code: vijf eigen versies van dezelfde weekschrijver (met zeven varianten van hetzelfde mappad, waarvan sommige een spatie niet weghalen, dus je krijgt twee mappen), twee bijna identieke zoekfuncties waarvan de instellingen uit elkaar

kunnen lopen en dan de cache verkeerd bedienen, twee identieke commandobestanden, een dubbel en onbereikbaar codeblok, een ongebruikte variabele, en vier functies die alleen maar een foutmelding gooien.

davinci-resolve-mcp-free (video-montage)

27 bestanden gelezen, 13 meldingen, 11 bevestigd, 2 afgevallen. Naast de zware hierboven:

- **Een kapot videobestand geeft "geen scènewisselingen"** in plaats van een foutmelding. Zelf getest met een onleesbaar bestand: nul resultaten, geen waarschuwing.
(`analyzers/scenes.py` regel 51)
- **Een ampersand in de bestandsnaam laat de export crashen.** Getest met "Ben & Jerry.mp4": technische foutmelding in plaats van je tijdlijn. (`exporters/fcpxml.py` regel 137)
- **De verbinding met Resolve wordt nooit verversd.** Start je Resolve opnieuw, dan blijft de tool met de oude, dode verbinding werken. (`studio_api/bridge.py` regel 56)
- **De echte reden waarom Resolve niet bereikbaar is, wordt weggegooid.** Je krijgt "bridge load failed" terwijl de code de nuttige hint ("zet externe scripting aan") al had klaarstaan.
(`mcp_server.py` regel 415)
- **De keuze bij magic mask doet niets.** Vraag je om een auto te maskeren, dan krijg je stappen voor een persoon terug. (`mcp_server.py` regel 347)
- Verder: een ongebruikte klasse, een ongebruikte functie, en twee zware installatiepakketten die nergens in de code gebruikt worden en de installatie kunnen laten mislukken.

haarvisie-security-agent (de beveiligingsscanter)

14 bestanden gelezen, 17 meldingen, 15 bevestigd, 2 afgevallen. Naast de zware hierboven:

- **Een fout in de TLS-check wordt gelezen als "veilig".** Ontbreekt het hulpprogramma of loopt het in een timeout, dan komt er geen melding en lijkt het in orde. Dat is precies het valse "nee" waar je al eerder mee bent gepakt. (`wordpress_checker.py` regel 30)
- **Het filter op valse meldingen verbergt echte sleutels.** Staat er op één regel een verwijzing naar een omgevingsvariabele en daarachter een echte hardgecodeerde sleutel, dan wordt de hele regel overgeslagen. Nagespeeld: nul bevindingen bij een echte sleutel.
(`code_scanner.py` regel 39)

- Een **.env-bestand met een vreemd teken** laat de hele scan crashen, zonder rapport en zonder melding. (`env_checker.py` regel 80)
- Een mislukte WhatsApp-melding zegt niet waarom en probeert het niet opnieuw. Is het token verlopen, dan krijgt niemand bericht en weet niemand waarom. (`notifier.py` regel 23)
- Is het token leeg, dan wordt de melding stil overgeslagen en eindigt het script zonder fout. Jij denkt dan dat er geen kritieke bevindingen waren. (`main.py` regel 115)
- De injectietest heeft geen rem op het aantal aanroepen. Elke geslaagde test kost een betaalde AI-aanroep en vult je chatlogboeken met nepvragen. Dit is een overtreding van je eigen vlootregel dat elke automatisering een rem hoort te hebben. (`injection_tester.py` regel 29)
- Verder: het rapport vervangt het precieze pad door alleen het domein, een variabele die alleen als commentaar in `.env` staat telt als "aanwezig", en WordPress-paden worden dubbel gecontroleerd zodat het aantal bevindingen wordt opgeblazen.

fleet-backup-lenovo (de Lenovo-bot)

28 bestanden gelezen, 21 meldingen, 19 bevestigd, 2 afgevallen. Naast de zware hierboven:

- **fix.ps1 doodt elk node-proces op de pc**, niet alleen de bot. Draait er nog een tool, dan valt die zonder waarschuwing om en komt niet terug. Dit raakt ook de eigen bot, wat volgens het regelboek nooit mag. (`fix.ps1` regel 9)
- De bot draait als geplande taak met de hoogste rechten. Samen met het punt hierboven over blind uitvoeren betekent dat: een verkeerde opruimopdracht kan bestanden buiten je gebruikersmap raken. (`register.ps1` regel 6)
- Een leeg antwoord laat dezelfde opdracht twee keer draaien, ook als het werk al gedaan was. Resultaat: twee identieke concepten in je Outlook. (`lenovo-bot.js` regel 97)
- Een download zonder foutopvang kan de hele bot laten omvallen midden in een gesprek. Een foutpagina van Telegram wordt gewoon als .jpg opgeslagen en aan Claude gegeven als "afbeelding". (regel 48)
- Een foutmelding gebruikt een naam die er nog niet is, waardoor het script kan stoppen voordat er een rapport wordt geschreven. (`zoek-afspraakverzoeken.py` regel 65)
- Namen en onderwerpen uit e-mail gaan ongefilterd in het HTML-rapport. Een afzender met een geprepareerde weergavenaam kan code laten draaien in jouw browser zodra je het rapport opent. Een naam met een `&` erin sloop sowieso de tabel. (regel 161)

- **Mislukt één map, dan wordt de hele onderliggende boom overgeslagen**, stil. Het rapport meldt "Totaal gevonden: 0" terwijl er niet gemeten is. Twee andere scripts in dezelfde map doen dit wel goed. (regel 50)
 - **Twee lezers op dezelfde wachtrij**. De oude worker is uitgezet maar staat er nog. Gaat hij bij een herstel weer aan, dan vinkt hij berichten af die de bot nooit ziet. (`queue-worker.js`)
 - Verder: bijlagen met dezelfde naam overschrijven elkaar zonder melding, twee bijna identieke Telegram-verzendscripts, twee identieke zoekscripts, een openingsbericht zonder foutopvang dat de hele verzending kan afbreken, een ongebruikte variabele, en een verplaatsactie in Outlook waarvan de nieuwe verwijzing wordt weggegooid.
-

Wat er afviel bij de controle

8 van de 104 meldingen zijn afgevallen. Ze klonken plausibel, maar hielden geen stand toen ze werden nagemeten. Dat is precies de bedoeling: liever streng dan een lijst met verzinselfs.

Twee voorbeelden:

- **"Elke https-verbinding via een IP-adres faalt" (gbrain)**. De code klopte, maar het was op de verkeerde omgeving gemeten. Onder node faalt het inderdaad, onder bun (waar gbrain echt op draait) geeft precies dezelfde aanroep gewoon een geslaagd antwoord. Nagemeten in beide.
- **"De Resolve-check geeft altijd vals nee" (video-tool)**. De redenering klopte, de aanname niet. macOS noemt een proces naar de interne naam, niet naar de naam van de app. Gemeten met drie draaiende apps op deze machine, en de interne naam van Resolve is precies wat de code zoekt. Bevinding onterecht.

Andere afvallers: een crash bij het wegschrijven van het rapport (bleek in geen enkele realistische situatie te gebeuren, met lege omgeving nagemeten), een telefoonnummer dat "gelekt" zou zijn maar gewoon op je website staat, een lek dat door een andere regel al werd afgevangen, en een geheugenlek dat door een bestaande opruimfunctie wordt gedekt.

Eén afvaller leverde iets beters op: bij het natrekken van de "telefoonnummer in de repo"-melding kwam de echte vondst boven water, namelijk het GitHub-token in `.git/config` (punt 2 hierboven).

Tabel per repo

REPO	BESTANDEN GELEZEN	GEMELD	BEVESTIGD
haarvisie-nieuw	93	25	25
gbrain	158	28	26
davinci-resolve-mcp-free	27	13	11
haarvisie-security-agent	14	17	15
fleet-backup-lenovo	28	21	19
Totaal	320	104	96