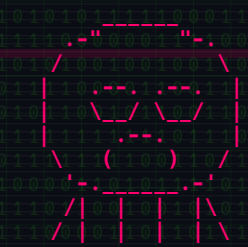
// 04/2026 // AIR-GAPPED

$ cat /dev/mag | hexdump -C

PLANET EXPRESS

FINDING PATTERNS IN THE NOIZE

EAT SLEEP HACK REPEAT



```
00004000  f1 d7 94 07 5b ed 01 15 22 2c c1 80 46 f1 ab a
00004010  99 8d f1 95 e7 04 4b 19 bc ab 42 f6 ce 65 f7 6
00004020  92 2e 81 f9 c2 33 22 6b 70 6b bb 0c 86 fd 67 7
00004030  a4 37 2a 6f 1c 60 d7 bc b9 66 a8 ed b2 04 d7 5
00004040  e6 d0 b4 3e ba 9d c5 bd f9 da dc ef ba 05 68 8
00004050  6d 8e d6 f1 db 6d 82 e1 0e 94 79 dd 9f 44 81 d
00004060  61 c0 de ca 6a 43 21 69 04 cd 83 8a 32 38 81 2
00004070  3e 77 26 2a bd de 61 6b 35 4f 27 7f 84 dd c7 1
00004080  ed 2b 2c 57 dd 09 70 e8 1b 8d b7 4b e2 77 2e e
00004090  90 a5 a8 90 2b 7c 53 2f 66 ef 20 f1 ea ef 94 c
000040a0  7f 08 09 04 56 09 04 72 85 12 ab 23 75 3c 2b 6
000040b0  62 24 ae 85 2d 99 29 c0 b9 d8 a6 23 b8 7b 6f 3
000040c0  4e 1b 1a b9 61 2b 0c b0 5f 1f cf a6 42 34 c7 0
000040d0  b4 28 c2 d8 6a e6 ba 9e a5 da f0 d7 88 fb 47 a
000040e0  af a2 93 6a 95 78 5c 7a 04 b1 87 fd f3 c8 dc c
000040f0  1f 45 52 7f c7 89 13 f0 54 17 3e 31 a5 8d 43 2
00004100  40 ab ce ef c9 a8 0d 95 64 44 b1 28 e5 fd ad d
00004110  2e bb 79 77 c5 ae 32 9c 07 b5 13 25 12 f3 40 b
00004120  a2 68 3b 23 0d bc 86 85 90 50 32 53 b2 3e 31 1
00004130  73 53 d0 42 ce 89 19 a0 b0 84 5b a3 1e 40 1b 6
00004140  61 57 00 0f 15 98 7f 2b ad 81 e3 cf ae bc aa 9
00004150  49 53 33 b4 22 5b 65 ba 21 79 1e 33 4f 8c c3 4
```

frei wie freibier

// FEATURE 01 // LIEFERKETTE

DIE AXIOS-AFFÄRE

Am 31. März 2026 war axios drei Stunden lang bösartig. 180 Millionen Downloads pro Woche, zwei Versionen betroffen — die Mechanik des Angriffs, Stufe für Stufe.

Es gibt eine bestimmte Sorte moderner Abhängigkeit: klein, unscheinbar, löst genau ein Problem — und sitzt inzwischen in so gut wie jedem JavaScript-Projekt dieses Planeten. axios ist so eine. Rund 180 Millionen npm-Downloads pro Woche, verteilt auf zwei Hauptzweige: 1.x und 0.30.x. Ein HTTP-Client, den die wenigsten Teams bewusst eingebaut haben und der trotzdem in jedem Build mitfährt.

Genau diese Allgegenwart ist das eigentliche Risiko. Was einen Supply-Chain-Angriff trägt, ist selten Cleverness, fast immer Reichweite.

Die Liste der Vorläufer ist lang. event-stream erreichte 2018 zwei Millionen wöchentliche Downloads, bevor sein Maintainer das Paket an einen Fremden übergab, der einen Wallet-Stealer für Copay-Nutzer ausrollte. ua-parser-js schleuste 2021 über einen gekaperten Maintainer-Account Krypto-Miner und Credential-Stealer aus. colors.js und faker.js wurden 2022 vom eigenen Autor sabotiert. node-ipc begann im selben Jahr, Dateien nach Geolokation zu löschen. xz-utils war 2025 eine dreijährige Geduldsarbeit — nur noch ein Beta-Release davon entfernt, sshd auf jeder Linux-Distribution mit einer Backdoor zu versehen.

Am 31. März 2026 um 00:21 UTC traf es axios selbst. Drei Stunden später, um 03:20, war das Fenster wieder zu. Was folgt, ist die Mechanik des Angriffs, Stufe für Stufe, so wie sie tatsächlich lief.

```
$ npm install --production

added 1847 packages in 23s

19 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

$ node server.js
server listening on :3000
POST packages.npm.org/product1  200
POST packages.npm.org/product1  200
POST packages.npm.org/product1  200
POST packages.npm.org/product1  200

# found 0 vulnerabilities
# found 1 Weltuntergang
```

```
[0] Maintainer-Account gekapert
      └─ neue E-Mail: ifstap@proton.me
            ▼
[1] plain-crypto-js @ 4.2.0 / 4.2.1
      └─ als axios-Abhängigkeit ergänzt
            ▼
[2] postinstall: node setup.js
            ▼
[3] SILKBELL (XOR + Base64)
      ├─ Win: %PROGRAMDATA%\wt.exe
      ├─ mac: /Library/Caches/...mond
      └─ lin: /tmp/ld.py
            ▼
[4] WAVESHAPER.V2 → sfrclak.com:8000
            ▼
[5] Persistenz: Run\MicrosoftUpdate

31.03.2026 · 00:21 - 03:20 UTC
```

// FEATURE 01 // LIEFERKETTE // CONT.

Stufe null: kompromittierter Maintainer-Account. Kein Typosquat, kein Nebenschauplatz – einem legitimen axios-Maintainer wurde sein npmjs-Konto aus der Hand genommen. Erstes sichtbares Artefakt: eine neu eingetragene Kontakt-E-Mail, ifstap@proton.me.

Stufe eins: versteckte Abhängigkeit. Der Angreifer hängte axios ein neues Paket an, plain-crypto-js, in den Versionen 4.2.0 und 4.2.1. Der Name klang harmlos; die eigentliche Angriffslogik saß vollständig in dieser Abhängigkeit, nicht im axios-Kern. Ein Diff auf axios selbst hätte nichts Verdächtiges gezeigt.

Stufe zwei: postinstall-Hook. plain-crypto-js trug in seiner package.json einen scripts-Eintrag, der bei jedem npm install automatisch `node setup.js` aufrief. Keine Rückfrage, kein Consent – Code direkt zur Installationszeit.

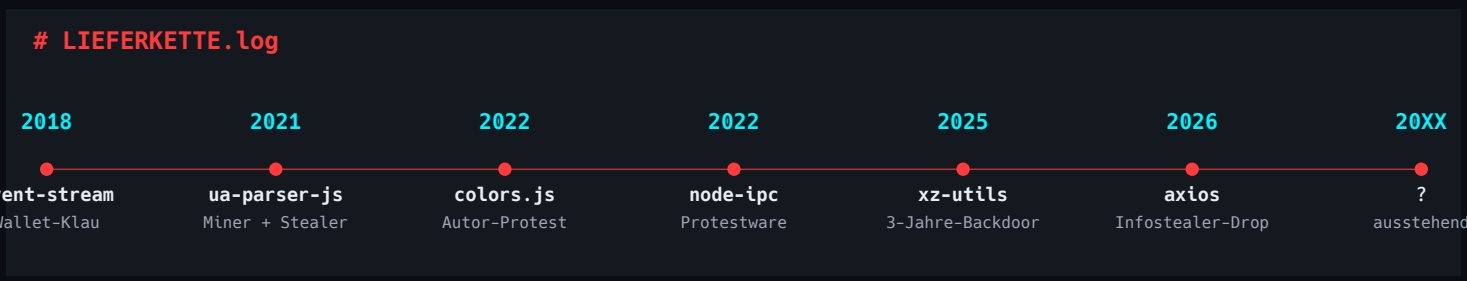
Stufe drei: SILKBELL, der Dropper. setup.js war XOR- und Base64-obfuskiert; C2-URLs und OS-spezifische Befehle entstanden erst zur Laufzeit. fs, os und execSync wurden dynamisch geladen, statische Analyse lief ins Leere. Nach dem Drop löschte sich setup.js selbst und benannte package.json in package.md um – damit die Forensik später länger sucht.

Stufe vier: OS-abhängige Payload. Unter Windows kopierte SILKBELL powershell.exe nach %PROGRAMDATA%\wt.exe – der Name eines legitimen Windows-Terminals, klassische Tarnung. Ein PowerShell-Skript wurde per curl von packages.npm.org nachgezogen – einer Lookalike-Domain, nicht der echten npm-Registry –, POST-Body `product1`. Unter macOS landete eine Mach-O-Binärdatei in /Library/Caches/com.apple.act.mond, Body `product0`. Unter Linux lief eine Python-Backdoor in /tmp/ld.py, Body `product2`.

Stufe fünf: WAVESHAPER.V2, die eigentliche Backdoor. Sie meldete sich im 60-Sekunden-Takt über Port 8000 bei sfrclak.com (142.11.206.73). Befehle: `kill`, `rundir` (Verzeichnis enumerieren), `runscript` (AppleScript), `peinject` (PE-Binary in einen Prozess injizieren). Ein fester User-Agent verriet die Familie – mozilla/4.0 (compatible; msie 8.0; windows nt 5.1; trident/4.0), eine IE8-Maskerade, die WAVESHAPER seit Jahren mitschleppt.

Stufe sechs: Persistenz, ausschließlich unter Windows. Eine versteckte %PROGRAMDATA%\system.bat plus ein Registry-Eintrag unter HKCU\...\Run mit dem Wert „MicrosoftUpdate" sorgten dafür, dass die Backdoor jeden Login neu anlief. Für macOS und Linux gab es keine Persistenz – der einmalige Durchlauf genügte.

Stufe sieben: Nachspiel. Hunderttausende gestohlene Secrets dürften seither kursieren. Die Gegenmittel sind keine Raketenwissenschaft – Lockfiles, `npm ci --ignore-scripts`, scoped Tokens, Egress-Allow-Lists im CI, ephemere Runner. Breit ausgerollt ist davon nichts, 2012 nicht und 2030 auch nicht. Offen bleibt nur, welches Paket als Nächstes drankommt.



// FEATURE 02 // FELDBERICHT

TRYHACKME IM ANY % SPEEDRUN

Ein Bot, der einen kompletten Cybersec-Lehrplan allein abarbeitet — und mit jedem Raum ein Stück souveräner wird.

TryHackMe ist eine Lernplattform für IT-Sicherheit. Dort bekommt man einen „Raum" zugewiesen — eine Aufgabe aus Erklärtexten, Kontrollfragen und meist einer virtuellen Zielmaschine. Richtige Antworten bringen XP, füllen den Fortschrittsbalken und schalten den nächsten Raum frei. Hunderte davon, sauber in Lernpfade einsortiert.

Die Ausgangsfrage dieses Versuchs: Was passiert, wenn man claude-code — einen LLM-Agenten, der in der Shell lebt, Dateien liest, schreibt und Befehle ausführt — auf diesen Lehrplan loslässt und einfach wartet?

Der erste Baustein ist unspektakulär. Ein Python-Skript fährt Playwright hoch, loggt sich einmalig von Hand ein (das Captcha will einen Menschen) und legt den Cookie-Jar ab. Von da an redet ein dünnes Tool direkt mit der internen API: GET /api/v2/rooms/tasks liefert jede Frage eines Raums, POST /api/v2/rooms/answer schickt eine Antwort zurück. Mehr ist die Plattform von innen nicht.

Der Agent bekommt einen Raum, liest die Aufgabe, überlegt, trägt die Antwort ein. Solange die Frage lautet „welches Wort steht im zweiten Absatz", ist das trivial. Sobald aber nmap gegen eine Ziel-VM laufen muss oder ein Login-Formular fällig wird, mutiert der API-Klicker zum kleinen Hacker: Shell auf, nmap 10.10.x.y, offene Ports gelesen, passendes Werkzeug selbst gewählt.

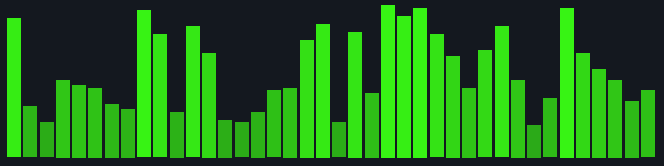
Ein Orchestrator fährt nicht einen, sondern mehrere Agenten gleichzeitig. Jeder bekommt genau einen Raum, ein festes Zeitbudget und ein eigenes Kontextfenster. Während der eine an einer SQL-Injection sitzt, räumt der zweite Linux-Forensik, der dritte ein Reverse-Engineering-Rätsel.

 # AGENT_POOL.dispatch

lernpfad: cybersecurity101

agent-001	**nmaplivehostdisc**	18:42	DONE	+400
agent-002	**linuxforensics**	—:—	RUN	
agent-003	**webfundamentals**	11:29	DONE	+300
agent-004	**introtodfir**	—:—	RUN	
agent-005	**burpsuitebasics**	03:11	FAIL	
agent-006	**wiresharkbasics**	—:—	RUN	
…	…	…	…	…

Pool-Größe	rollend
erledigt	34 / 43
Wall-Runtime	07:21:44
Tokens verbraten	2,8 M
Rezepte	wachsend



geräumte Räume · rollend 24h · y=Anzahl, x=Stunde

// FEATURE 02 // FELDBERICHT // CONT.

Der interessante Teil ist das, was zwischen den Räumen passiert.

Neben dem Code liegt ein Verzeichnis namens skills/ — das Gedächtnis des Bots. Darin: eine SKILL.md, die beschreibt, wie ein Raum grundsätzlich anzugehen ist (erst lesen, dann scannen, dann gezielt ausnutzen). Eine lessons.md mit Notizen aus früheren Solves („manche Static-Site-Flags stehen base64-kodiert im JS-Bundle — grep reicht"). Ein recipes/-Ordner, je eine Markdown-Datei pro Angriffstechnik — SMB-Recon, JWT-Manipulation, Buffer-Overflow — jeweils als fertiges Kochrezept.

Vor jedem Solve liest der Agent alles davon. Nach jedem Solve, wenn er unterwegs etwas Neues gesehen hat, schreibt er den Fund zurück: neue Datei, neuer Eintrag, kleines Diff.

Der Bot wächst mit dem Korpus. Räume, die in der ersten Woche an ihm gescheitert sind, fallen in der zweiten in Minuten, weil inzwischen das passende Rezept dafür existiert. Das Setup pflegt sich dabei selbst — jeder gelöste Raum hinterlässt ein Stück mehr Dokumentation für den nächsten.

Zwei Dinge laufen parallel, und sie lassen sich kaum sauber trennen. Lernen: über APIs, Captchas, Rate Limits, die Anatomie eines Angriffs. Schummeln: die Agenten klicken „Check" auf Fragen, die niemand gelesen hat. Der Fortschrittsbalken bewegt sich trotzdem.

Für die CTF-Szene ist das wenig überraschend. Ein Leaderboard war nie ein Zeugnis, sondern ein Anwesenheitssignal; Automatisierung würde es früher oder später einholen. Nur ist sie jetzt eben schneller da, als die Lehrpläne gerechnet haben.

Für IT-Sicherheit insgesamt wiegt es schwerer. Skripte gab es schon immer; neu ist, wie groß der Anteil der Arbeit ist, den ein Bot mit ein paar Markdown-Dateien als Gedächtnis übernimmt. Die Einstiegshürde „erfahrener Red Teamer" ist gerade erst auf „jemand mit einem freien Wochenende" gefallen. Die ehrliche Frage an die blaue Seite lautet: Was kann die Gegenseite heute schon automatisieren, was wir noch nicht können? Die Antwort fällt unangenehm aus.

// WERKZEUG

NMAP

Das Schweizer Taschenmesser, das sich weigert, in Rente zu gehen.

Gordon Lyon – Fyodor – hat nmap 1997 als Artikel im Phrack veröffentlicht. Neunundzwanzig Jahre später ist es immer noch das Erste, was jemand tippt, wenn ein neues Netz vor der Nase liegt. Ein größeres Kompliment kann man einem Stück Software nicht machen.

Was nmap langlebig macht: Es hat keine Meinung. Will nicht hübsch sein, will kein Agent sein, will nicht in deinem Browser wohnen. Es ist eine Taschenlampe, die du auf ein Netz richtest, und es sagt dir, was es gesehen hat.

Das am häufigsten Unterschätzte an nmap: Die nackte Form reicht. Einfach `nmap <Ziel>`. Keine Flags, keine Skripte, kein Feintuning. Der Scan nimmt die Top-1000-TCP-Ports, rät pro Port den dahinterliegenden Dienst und liefert eine erste Silhouette des Ziels. In den meisten Fällen genügt diese Silhouette, um zu entscheiden, wo als Nächstes hingeschaut wird. Der Spickzettel unten beantwortet die Fragen, die der erste Scan aufwirft.

SPICKZETTEL

`-sS`	stealth SYN-Scan
`-sV`	Version erkennen
`-O`	OS-Fingerabdruck
`-A`	aggressiv (alles davon)
`-p-`	alle 65535 Ports
`-T4`	schnelles Timing
`--open`	nur offene zeigen
`--script`	NSE-Script-Engine
`vuln`	Vuln-Detection-Skripte
`smb-enum-*`	SMB-Recon-Paket
`http-title`	HTTP-Titel greifen
`-iL <datei>`	Ziele aus Datei lesen

$ nmap ohne Flags:

```
nmap 10.10.10.42
```

genauso erlaubt: nmap example.com nmap 192.168.1.1-254 nmap 10.0.0.0/24
Top-1000-TCP-Ports, Dienst-Tipp pro Port.

// KOLUMNE // TERMINAL-TRICKS

SIEBEN KNIFFE, DIE DU WIRKLICH NUTZT

01 !!

Wiederholt den letzten Befehl. Mit sudo kombiniert: `sudo !!` — die meistgenutzte Zwei-Zeichen-Sequenz der Ops-Geschichte.

02 Strg-R

Rückwärts-Suche durch die Shell-History. Einfach lostippen, Strg-R nochmal für weiter zurück. Spart dir das Merken von Flags.

03 ssh root@segfault.net

Passwort: `segfault`. Eine frische Kali-VM mit Root, geschenkt von THC — neuer Rechner pro Login, Tor + VPN inklusive. Labor ohne Setup.

04 cd -

Springt ins vorherige Verzeichnis. Ping-Pong zwischen zwei Orten, ohne Pfade zu tippen. Zusammen mit `pushd` / `popd`, wenn es ordentlich sein soll.

05 python3 -m http.server 8000

Sofort-Fileserver im aktuellen Verzeichnis. Dateien zwischen VMs schieben, ein pcap an Kolleg:innen reichen, Payloads stagen. Null Abhängigkeiten.

06 ss -tulpn

Zeigt lauschende Sockets samt Prozess dahinter. Ersetzt netstat auf modernen Systemen. Jede:r sollte wissen, was auf der eigenen Kiste lauscht.

07 script -t timing.log session.log

Nimmt die komplette Terminal-Session auf — Tasten, Timing, Output — und spielt sie mit `scriptreplay` ab. Dein zukünftiges Ich dankt dir.

nerial.uk/

spiele · filme · serien · musik · bücher · leaks

# katalog	// 18 · 2026-04	
Cyberpunk 2077 + alle DLCs	[CPY]	ISO
GTA VI Cutscene-Dump (voll)	[???]	MP4
DAZN	[STREAM]	M3U8
Epstein Files, unzensiert	[LEAK]	PDF
Adobe 2025 Master Suite	[TRB]	ZIP
internes PRISM-Deck	[DOC]	PPT
Starcraft + Brood War (orig.)	[RAZ]	ISO
Windows 12 Pro (pre-RTM)	[FTCU]	ISO
Scrubs (2026)	[NERIAL]	MKV

https://nerial.uk/

# katalog	// cont.	
Rick & Morty S13 (pre-air)	[NTb]	MKV
Sky	[STREAM]	M3U8
The Last of Us Part III (Dev)	[LEAK]	PKG
Avatar 3 – Fire and Ash	[EVO]	MKV
Stranger Things S05 komplett	[NTb]	MKV
Zelda – Tears of the Kingdom II	[VENOM]	XCI
Half Life 3 (internal build)	[3DM]	ISO
Dune: Messiah (SCR.DVDRip)	[FGT]	MKV
Adobe Creative Cloud 2026	[XFORCE]	ZIP

// FEATURE 05 // KOMMENTAR

CTF NACH DEM AGENTEN

Der TryHackMe-Artikel auf Seite 4 war der einfache Teil: Der Agent funktioniert. Der schwerere Teil ist, was so ein Auto-Solver mit einer Szene macht, die darauf gebaut hat, dass Menschen die Aufgaben lösen.

Ein Agent, der Räume leerräumt, macht aus der Flag eine Quittung für etwas anderes — für die Fähigkeit, Tools zu verdrahten und laufen zu lassen. Das bleibt eine echte Fertigkeit, nur eben nicht die, die der Score je gemessen hat.

Der Schach-Vergleich trägt weiter, als es auf den ersten Blick aussieht. Deep Blue hat Schach nicht beendet; beendet wurde nur die Erzählung, dass oben der Mensch die Decke sei. Geblieben ist der Sport als menschliche Tätigkeit, im menschlichen Tempo gespielt und sauber von der Engine-Analyse getrennt. Online-Plattformen kennzeichnen Engine-Hilfe heute so, wie Dopingtests PEDs kennzeichnen. Centaur-Schach, die Mischform aus Mensch und Engine, hatte fünfzehn Jahre lang einen Moment — und verschwand wieder, weil die Engines allein ohnehin besser spielten.

Speedrunning hat denselben Druck anders aufgefangen. TAS — tool-assisted — ist eine eigene Kategorie, klar gelabelt und framegenau, und dorthin wird kein Mensch je kommen. Niemand tut so, als wären ein TAS-Lauf und ein RTA-Lauf vergleichbar; die Community hat die Trennlinie früh gezogen, und sie hält bis heute.

Bei CTFs ist diese Linie noch nicht gezogen. In den THM-Leitern, HTB-Rängen und monatlichen Scoreboards steckt längst ein unbekannter Anteil Agent-Arbeit, das Signal verrauscht von unten. Irgendwann bedeuten die Rankings nicht mehr, was die Leute in sie hineinlesen — dann müssen die Plattformen entweder eine eigene Division aufmachen (Agenten erlaubt, klar markiert) oder sie verkommen zur Dekoration.

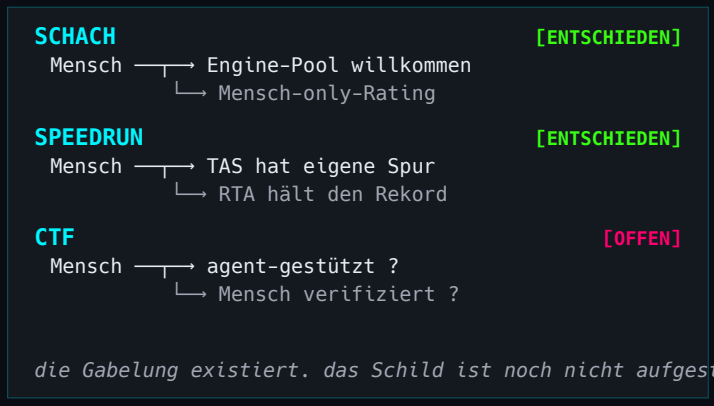
Der Verteidigung bleibt Spielraum. Puzzle-Designer stützen sich längst auf das, was Modelle schlecht können: Steganographie, die an menschlicher Bildwahrnehmung hängt; OSINT gegen unvorhersehbare Quellen; physische Artefakte, die man in der Hand halten muss; Out-of-Band-Interaktionen, die das Agent-Harness gar nicht mitbekommt. Das obere Ende der Kurve wird für alle härter, Agent hin oder her. Das untere Ende — Tutorial-Räume, CVE-Nachbauten, bekannte Web-Muster — ist bereits geräumtes Gelände.

Für die Spieler wandert die Fertigkeit eine Ebene nach oben: den Agenten schreiben, die Tools instrumentieren, erkennen, wann das Modell blufft und wann es recht hat, und die Form eines Problems sehen, an der er scheitern wird — bevor darin eine Stunde verbrennt. Das ist echte Kompetenz. Nur liegt sie näher an SRE als an Offense, und sie fühlt sich nicht so an wie eine Box, die man nachts um drei von Hand aufreißt. Manche werden es lieben. Viele, die CTFs geliebt haben, werden es nicht.

 # post_agent.diff

Leiter / XP	SIGNAL → RAUSCHEN
Tutorial-Räume	AUTO-GEKLÄRT
CVE-Nachbau-Challenges	AUTO-GEKLÄRT
Stego (visuell)	BLEIBT SCHWER
OSINT (Mensch-Quelle)	BLEIBT SCHWER
physisch / OOB	BLEIBT SCHWER
private Szene-CTFs	UNBERÜHRT
den Agenten schreiben	NEUE SKILL
wissen, wann man stoppt	NEUE SKILL

schach hat die kategorie geteilt. speedrunning auch. ctf noch nicht.



die Gabelung existiert. das Schild ist noch nicht aufgestellt.

Das ehrlichste Argument kommt aus der kleinen Szene. Private CTFs unter Freunden, ohne Scoreboard, ohne XP, ohne Plattform — die laufen weiter. Der Mechanismus waren ja nie die Punkte, sondern der Raum mit dem Whiteboard und den sechs Leuten, die über denselben PCAP streiten. Dort richten Agenten nichts an; was sie anrühren, ist die öffentliche Leiter, und die war schon vorher der schwächste Teil des Hobbys.

Nichts davon beendet CTFs; beendet wird nur die Rolle als Ranking-System. Das Lernen bleibt, die Scoreboards gehen oder teilen sich. Was danach kommt, ist entweder ehrlich gelabelt — Agenten willkommen, Menschen willkommen, gleiche Flag, getrennte Spalten — oder ein Kategorienkrieg mit Verifikationsverfahren, denen niemand so richtig traut. Schach hat den ersten Weg gewählt. Speedrunning auch. CTFs haben noch nicht gewählt.

// FEATURE 06 // GESUNDHEIT

GESÜNDER HACKEN: VITAMINE

Was bei Pizza um drei und Bildschirm statt Sonne leise auf der Strecke bleibt – und was fünfzig Cent aus der Drogerie dagegen tun.

Hacker-Kost ist keine Ernährungspyramide. Pizza nach Mitternacht, Club-Mate statt Wasser, Tiefkühl-Gemüse als Alibi – und Tageslicht überwiegend durchs Küchenfenster. Selbst wer diszipliniert kocht, bekommt nicht automatisch alles in ausreichender Menge: ein paar Spurenelemente, die fettlöslichen Vitamine, Magnesium in stressigen Phasen. Zwölf Stunden vor dem Monitor plus gelegentlicher Döner, und die Lücke wird zur Regel.

Die gute Nachricht: Sie lässt sich zum Preis eines Energy-Drinks schließen. Die schlechte: Jeder zweite Podcast verkauft dafür ein 40-Euro-Monatsabo – personalisiertes Pulver, proprietäre Mischung, Influencer-Code an der Kasse. Zwischen dieser Mischung und dem, was auf einer Drogerie-Brausetablette steht, liegt im Wesentlichen die Verpackung.

Die Empfehlung der Redaktion ist deshalb unspektakulär: eine Multivitamin- und eine Multimineral-Brausetablette pro Tag, zusammen rund ein Euro aus der Drogerie. Ein Glas Wasser, zehn Sekunden Plopp, fertig.

Dazu, aus einem spezifischen Grund: Vitamin D3. Von Oktober bis April steht die Sonne über weiten Teilen Mitteleuropas zu flach, als dass die Haut überhaupt noch D3 bilden könnte; wer ohnehin drinnen sitzt, startet den Februar mit fast leerem Speicher. Rund 1.000 IE täglich halten den Pegel stabil. D3 ist fettlöslich und legt sich als Depot ab – man darf also rechnen: 2.000 IE alle zwei Tage, 20.000 IE alle zwanzig. Nur der Jahresdurchschnitt zählt.

Und eine Empfehlung, die kein Präparat ist: täglich zwanzig Minuten Spaziergang. Tageslicht, Puls eine Stufe über Sitzruhe, Augen auf etwas, das weiter weg ist als der Monitor. Ersetzt kein D3. Ersetzt erstaunlich viel anderes.

rezept.txt

Multivitamin-Brause	50 ct
Multimineral-Brause	50 ct
Vitamin D3 (1.000 IE)	3 ct / Tag
Spaziergang (20 min)	gratis
Summe	**~ 1,05 € / Tag**

depot.plan

täglich	1 x 1.000 IE
alle 2 Tage	1 x 2.000 IE
alle 20 Tage	1 x 20.000 IE

D3 ist fettlöslich – der Jahresdurchschnitt zählt, nicht der einzelne Tag.

podcast_vs_drogerie.diff

```
- personalisierte Mischung, 40 €/Monat, Influencer-Code
+ Drogerie-Brause, 50 ct, kein Abo
```

EOF

// Verbindung zur Gegenstelle beendet

planet_express_2026_04 // lang=de