A Minimal, Neutral Cyber-Signals Ticker: Real-Time U.S. Headlines via GDELT and PowerShell

Douglas C. Youvan

doug@youvan.com

www.youvan.ai

September 27, 2025

Threat-intel dashboards are often heavy, vendor-tied, and opinionated. This paper describes a lightweight, terminal-only “news ticker” that continuously surfaces last-24-hour headlines related to cyberattacks and official warnings affecting the United States. It relies on the public GDELT DOC 2.0 index for broad, worldwide coverage, but keeps complexity down through a few practical choices: compact sub-queries to avoid API length limits; simple, readable regex filters to ensure results include clear cyber terms (e.g., ransomware, DDoS, CVE) plus U.S. context or U.S. cyber agencies; and a concise PowerShell loop that merges, de-duplicates, and prints a clean table each cycle. The design is intentionally neutral—no sentiment scoring or “reliability” labels—and avoids file I/O so it can run anywhere a modern PowerShell is available. We also cover resilience tactics (handling partial DNS blocks, HTTPS quirks, API hiccups), tunable trade-offs (strictness vs. recall), and how to extend the script for different topics, refresh cadences, or environments (e.g., scheduled task, serverless cron). The result is a tiny, auditable tool that can sit on a side monitor and keep analysts oriented to active U.S.-focused cyber signals without adding operational drag—or vendor lock-in.

Keywords : GDELT, PowerShell, cyber intelligence, ransomware, DDoS, CVE, CISA, US Cyber Command, DNS over HTTPS, threat detection, news aggregation, regex filtering, open source intelligence, OSINT, live ticker. 42 pages. CC4.0.

Note: A collaboration with GPT-5. See also: https://www.researchgate.net/publication/395920073_The_Coming_Cipherstorm_Anticipating_Israel's_Cyber_Retaliation_Against_the_United_States

Outline

1. **Problem & Goals**
    - Why lightweight, neutral monitoring matters
    - Constraints: no sentiment, minimal dependencies, terminal-only
2. **Data Source & Scope**
    - GDELT DOC 2.0 (last-24-hours), English focus
    - What we *do* and *don't* ingest (titles only for speed/clarity)
3. **Design Overview**
    - Sub-query strategy to avoid "too short/too long" API errors
    - Merge + de-dup model; cycle summary (Raw/Kept/Dropped/Domains)
4. **Filtering Logic**
    - Cyber signal regex (ransomware/DDoS/CVE/zero-day, etc.)
    - U.S. context + agency gating (CISA/USCYBERCOM/NSA; FBI/DHS when cyber is present)
    - Optional session-level de-dupe across cycles
5. **Resilience & Network Considerations**
    - Handling empty/HTML responses, retries, fallbacks
    - DNS hardening (DoH/DoT), VPN DNS, why not hard-coding IPs
    - When QUIC/ECH help; lawful/policy cautions
6. **Operations**
    - Tuning cadence and volume ($interval, $cap, adding/removing sub-queries)
    - Reading the console output; common false positives and quick fixes
7. **Security & Ethics**
    - Neutral presentation vs. bias claims
    - Privacy notes; avoiding scraped credentials/cookies; API rate considerations

8. **Evaluation**
    - Example cycles and effect of stricter/looser filters
    - Precision vs. recall trade-offs; typical domains surfaced
9. **Limitations & Future Work**
    - Title-only blind spots; multilingual expansion; topic packs
    - Optional persistence (SQLite) and web UI variant
    - Pairing contradictory headlines (“outlier pairing” strip)
10. **Conclusion**
    - What this ticker is—and is not—for; when to graduate to fuller TI stacks
11. **Appendix A — Listing 1: Live Console Script**
    - **Insert the current running script here.** (Paste the entire PowerShell block you’re using.)
12. **Appendix B — Tunables (Copy-Paste Ready)**
    - Alternate cyber regexes (strict vs. broad)
    - US/agency patterns, sample sub-queries by sector
13. **Appendix C — Deployment Variants**
    - Scheduled Task snippet (Windows)
    - Loop-with-logging variant
    - Minimal Linux/macOS adaptation notes

References

PS Console

1. Problem & Goals

Why lightweight, neutral monitoring matters

Security teams (and solo operators) don't need another heavyweight dashboard—they need a small, dependable heartbeat of what's happening **right now**. Traditional threat-intel portals often gate content behind accounts, inject vendor bias ("our detections"), and require agents or backend services. Even general news aggregators optimize for engagement, not signal quality, and they frequently blend cyber stories with politics, opinion, or low-value syndication.

A **lightweight, neutral monitor** fills the gap:

- **Always-on awareness:** A tiny console feed you can leave on a side screen reduces time-to-notice for breaking advisories (e.g., CISA KEV additions, mass-exploited CVEs, ransomware campaigns).
- **Auditability:** Simple, readable rules (sub-queries + regex filters) make it obvious *why* an item appeared—no black-box scoring.
- **Low friction, high portability:** Runs anywhere PowerShell 7 runs, with zero setup beyond an internet connection; ideal for personal workstations, jump boxes, or ephemeral lab VMs.
- **Bias control:** By avoiding sentiment and "reliability" labels, the tool presents raw signals (titles + sources) and leaves interpretation to the human analyst—useful when stakeholders disagree about source trust.

Constraints: no sentiment, minimal dependencies, terminal-only

To keep the tool durable and easy to adopt, we set hard constraints up front:

- **No sentiment / stance analysis**
  Sentiment classifiers add opacity and false certainty, especially on terse headlines. We intentionally output **unscored** items so analysts can apply their own context. This also avoids model drift and dependency on external NLP libraries or cloud AI.
- **Minimal dependencies**
  - **Data**: One public endpoint (GDELT DOC 2.0).
  - **Runtime**: Stock PowerShell 7; no modules to install; optional use of built-in Invoke-RestMethod/Invoke-WebRequest.
  - **State**: In-memory only (no files, DBs, or services), unless the operator opts into persistence later.

- **Terminal-only UX**
    - **Why**: Fast startup, negligible resource use, easy copy/paste, works over SSH/RDP, and trivial to automate.
    - **Output**: A consistent, greppable table (Time, Outlet, Title, Link) plus a one-line summary (Raw/Kept/Dropped/Domains).
    - **Ops**: Single paste to run; **Ctrl+C** to stop. Tuning is one-liners ($interval, $cap, regex edits).
- **Robustness over completeness**
  We prefer short, focused sub-queries (to survive API length/strictness limits) and simple post-filters that remove obvious noise (e.g., non-US "NSA" = National Security Act). The result is a resilient feed that may miss some edge cases but rarely stalls.
- **Extensibility without lock-in**
  All logic is plain text. You can fork the topic (e.g., "cloud security," "industrial control systems"), add sector-specific sub-queries, or export to JSON later—without changing the core operational model.

2. Data Source & Scope

GDELT DOC 2.0 (last-24-hours), English focus

We use the **GDELT DOC 2.0** endpoint as our only data source. GDELT continuously crawls and normalizes news from a very large, worldwide set of outlets. For our needs, it offers three advantages:

1. **Breadth at low friction** – One public HTTPS API yields fresh URLs and metadata without accounts, SDKs, or cookies.
2. **Near-real-time latency** – New items typically appear within minutes; we poll on a short cadence but only ask for the **last 24 hours** to keep the feed current and bounded.
3. **Structured outputs** – We request compact lists of articles (DOC "artlist" mode) and accept either format=JSON or format=JSONFeed, whichever succeeds.

To maximize reliability (and avoid GDELT's strict boolean/length limits), we issue several **small, focused sub-queries** (e.g., "ransomware AND U.S.", "CISA advisory") and then merge/dedupe the results. We set a conservative maxrecords per sub-query and cap the final table for readability. The time window is a **sliding 24h**—older items naturally age out.

We apply an **English-only** scope for two reasons: (a) simpler filtering (regex over titles), and (b) most U.S. cyber advisories and trade press are published in English. The trade-off is obvious—we will miss non-English coverage and some multilingual local reporting. If needed, the approach generalizes: add language-specific sub-queries and parallel regex filters.

What we do and don't ingest (titles only for speed/clarity)

We ingest (per item):

- **Title** – used for all filtering (cyber terms, U.S./agency cues).
- **URL** – displayed as the link; used in deduplication.
- **Timestamp** – seendate (JSON) or date_published/date_modified (JSONFeed), localized for display.
- **Outlet (domain)** – parsed from the URL and shown for quick source awareness.

We intentionally do *not* ingest:

- **Full article text** – avoids heavy requests, paywall scraping, and NLP complexity; keeps the loop fast and polite.
- **Body/lead/summary fields** – even when present, we ignore them to maintain a transparent, title-only heuristic.
- **Images, social metrics, author, geo, tone** – these add bias/weighting we don't want and slow the cycle.
- **Source "reliability"/sentiment** – by design, the ticker is neutral; analysts keep their own mental model of trust.

Implications & mitigations:

- **Precision vs. recall:** Title-only filtering will occasionally drop relevant stories that bury key terms in the body, or accept marginal ones with keywordy titles. Our post-filters (cyber-term + U.S./agency cues) and conservative sub-queries bias toward actionable items while limiting noise.
- **Syndication duplicates:** The same story appears on many domains; we dedupe by Title+URL within a cycle and can optionally suppress repeats across cycles in-memory.
- **Paywalls:** Some links lead to paywalled articles; we still show the headline for situational awareness.

- **API quirks:** If the endpoint returns empty/HTML or rejects a sub-query (“too short/long”), we skip that sub-query and proceed; other sub-queries still populate the cycle.

This scoped, headline-centric ingestion keeps the system tiny, explainable, and resilient—ideal for an always-on console that surfaces U.S.-focused cyber signals without turning into a full crawler or analytics stack.

3. Design Overview

Sub-query strategy to avoid “too short/too long” API errors

The GDELT DOC 2.0 endpoint enforces strict limits on boolean length and phrase granularity. Long, catch-all queries (e.g., ("ransomware" OR "data breach" OR "DDoS" …) AND ("United States" OR "U.S." …)) often trigger *“Your query was too short or too long”* or *“The specified phrase is too short.”* To stay inside the guardrails **and** keep recall high, we:

- **Split intent into compact sub-queries.** Each sub-query targets one signal family:
    - Single threat-type + U.S. context ("ransomware" AND ("United States" OR "U.S." OR US)).
    - Advisory/alert families (("Cybersecurity and Infrastructure Security Agency" OR CISA) AND (advisory OR alert OR warning)).
    - Sector + attack verbs (("critical infrastructure" OR "power grid" OR pipeline OR water) AND (malware OR attack OR intrusion)).
    - Named actors ((LockBit OR ALPHV OR "Volt Typhoon") AND ("United States" OR "U.S.")).
- **Prefer short literals over sprawling OR trees.** Keep each line < ~120–150 characters and avoid very short naked tokens that GDELT rejects (e.g., bare US is allowed only when joined with longer phrases).
- **Run sub-queries independently and tolerate failure.** If one sub-query returns HTML, empty, or a length error, we skip it; the others still populate the cycle. We also try both format=JSON and format=JSONFeed to ride through occasional format glitches.
- **Bound per-query volume.** We set maxrecords conservatively (e.g., 120) to avoid response bloat and timeouts, then cap the final table (e.g., 60 rows) for readability.

This “many small nets” approach yields robust coverage without fighting the API’s length rules, and it degrades gracefully when a single net tears.

Merge + de-dup model; cycle summary (Raw/Kept/Dropped/Domains)

Because we fetch multiple overlapping slices, the assembly step is explicit and transparent:

1. **Collect:** For each successful sub-query, pull the result set (articles for JSON or items for JSONFeed) and normalize the minimal fields we need: Title, URL, seendate|date_published|date_modified, and Outlet (parsed domain).
2. **Filter (title-only):** Apply a clear post-filter:
    - **Cyber signal present** (regex: ransomware, DDoS, CVE-YYYY-NNNN, zero-day, intrusion, malware, etc.), **and**
    - **U.S. context or U.S. cyber agency** in the title (e.g., “United States”, “U.S.”, CISA, USCYBERCOM, “National Security Agency”).
      Optional nuance: allow **FBI/DHS** only when a cyber term is also present to avoid generic law-enforcement headlines; ignore ambiguous tokens like *NSA* when used as “National Security Act” abroad.
3. **De-duplicate (within cycle):** Use a deterministic key such as **Title + URL**. This removes:
    - repeats produced by overlapping sub-queries,
    - RSS refreshes with identical link,
    - and trivial duplicates across formats.
      (Optional, not default): maintain a **session-memory set** keyed by URL (TTL ~24h) to suppress repeats across cycles in the same PowerShell session.
4. **Normalize time & sort:** Parse seendate/date_published to local time; sort **descending** so the newest headlines appear first.
5. **Cap for readability:** After dedupe and sort, select the first *N* (e.g., 60). This keeps the console table legible and the signal glanceable.
6. **Summarize the cycle:** Before printing the table, we emit a one-line heartbeat:
    - **Raw** — total headlines pulled from all sub-queries *before* filtering and dedupe.
    - **Kept** — items that passed the title filter and dedupe and will be shown.
    - **Dropped** — items removed by filters (off-topic, non-U.S., ambiguous agency tokens).

- **Domains** — unique outlet count among displayed rows (a quick diversity gauge).

Edge considerations

- **Retitles & syndication:** Same story, different domains, or minor headline edits will survive per-cycle dedupe (we key on Title+URL). That's intentional: different outlets offer provenance and redundancy. If desired, a looser de-dup key (host + first 60 chars of title) can collapse near-duplicates at the cost of occasionally hiding legitimate follow-ups.
- **Paywalls:** We keep paywalled links; they still provide situational awareness via the title and source.
- **API hiccups:** If a response isn't JSON (HTML/empty), that sub-query is skipped; the cycle still completes and prints. The next cycle will retry naturally.
- **Throughput vs. precision:** Tightening the cyber regex (e.g., requiring ransomware|ddos|cve-… only) increases precision but lowers recall; broadening it adds chatter. The summary line helps quantify the trade-off quickly.

4. Filtering Logic

Cyber signal regex (ransomware/DDoS/CVE/zero-day, etc.)

We gate on **clear technical signals** that typically indicate an incident, exploitation, or actionable advisory. The pattern is intentionally compact, case-insensitive, and tolerant of hyphenation/spacing variants:

```
\b(cyber ?attack
 | ransomware
 | data breach
 | intrusion
 | ddos
 | malware
 | zero[ -]?day
 | phishing
 | exfiltrat
 | backdoor
```

| vulnerab

| cve-\d{4}-\d+

)\b

Design choices

- **Stems for families:** exfiltrat, vulnerab catch *exfiltration/exfiltrate* and *vulnerability/vulnerable* without a long alternation list.
- **Variant-tolerant:** zero[-]?day and cyber ?attack admit "zero day/zero-day" and "cyberattack/cyber attack."
- **Boundaries:** \b reduces spurious matches (e.g., "DDOSing" still matches, "oddos" doesn't).
- **CVE anchor:** cve-\d{4}-\d+ spotlights specific, named vulnerabilities in advisories.

Tuning

- **Stricter mode:** drop data breach and vulnerab to prioritize active exploitation over disclosures/compliance news.
- **Broader mode:** add families like *supply chain*, *wiper*, *lolbins*, *initial access*; or vendor/ICS terms (PLC, HMI) if you're OT-focused.

U.S. context + agency gating (CISA/USCYBERCOM/NSA; FBI/DHS when cyber is present)

To keep the feed **U.S.-focused** without geolocating bodies, we layer simple, readable cues over the title:

1. **U.S. context:**
2. \b(United States|U\.S\.|USA|American)\b

plus a **case-sensitive** \bUS\b to accept "US" while rejecting the pronoun "us".

3. **Strong U.S. cyber agencies (always acceptable):**
4. \b(Cybersecurity and Infrastructure Security Agency
5. | US Cyber Command
6. | United States Cyber Command
7. | National Security Agency
8. | CISA)\b

These are highly indicative of U.S. cyber relevance even if “United States” isn’t spelled out in the same title.

9. **General agencies (FBI/DHS) require a cyber term:**
10. \b(FBI|DHS)\b AND (CYBER_REGEX)

This prevents political/crime headlines from slipping in solely due to “FBI/DHS” mentions.

Ambiguity guardrails

- **NSA collision:** “NSA” can mean *National Security Agency* or *National Security Act* (e.g., Indian headlines). We avoid bare \bNSA\b and rely on the **full agency name** or other U.S. context.
- **“US” vs. “us”:** we deliberately use \bUS\b (case-sensitive) alongside the phrase regex to avoid pronoun hits.
- **Syndication language:** some wires title advisories tersely (e.g., “CISA adds CVE-2025-12345”); the strong-agency list ensures these still pass.

Effective rule (readable logic)

- Keep if:
    - (CYBER_TERM AND (US_CONTEXT OR STRONG_US_AGENCY))
    - OR STRONG_US_AGENCY (by itself, e.g., “CISA: Patch Cisco…”)
    - OR (FBI_OR_DHS AND CYBER_TERM)

This yields high precision on **operationally useful** U.S. cyber items without over-curating.

Optional session-level de-dupe across cycles

Within a single refresh, we dedupe by **Title+URL**. To suppress repeats **across** refreshes (while the console runs), keep a small, in-memory set:

- **Key:** Prefer **URL** (stable) rather than title (retitles happen).
- **TTL:** 24 hours (aligns with the query window).
- **Purge:** On each cycle, remove entries older than the TTL to cap memory use.
- **Collision risk:** Minimal; if two outlets reuse identical URLs (rare), they’ll be treated as one—acceptable for a lightweight ticker.

Example (PowerShell, conceptual)

```
# once
if (-not $global:SeenUrls) { $global:SeenUrls = @{} }
$ttlHours = 24
$now = Get-Date

# purge old
$global:SeenUrls.Keys |
  Where-Object { ($now - $global:SeenUrls[$_]).TotalHours -ge $ttlHours } |
  ForEach-Object { $global:SeenUrls.Remove($_) } | Out-Null

# before adding an item:
if ($global:SeenUrls.ContainsKey($url)) { continue }  # already shown this session
$global:SeenUrls[$url] = $now
```

Variants

- **Near-duplicate suppression:** key on host + first 60 chars of title to collapse syndication clones; expect occasional over-merges (follow-ups with similar titles).
- **Persistence:** if you later allow file I/O, serialize the set to a small JSON/SQLite store to survive restarts; still apply TTL on load.

Trade-offs

- **Recall vs. fatigue:** Cross-cycle dedupe reduces repetition fatigue but can hide legitimate updates on the same URL (e.g., live blogs).
- **Syndication diversity:** URL-based keys allow different outlets to appear; a stricter title-based key may hide valuable corroboration.

Overall, these filters favor **explainability** over cleverness: a junior analyst can read the rules, predict outcomes, and adjust them in minutes—no model retraining or opaque scores required.

5. Resilience & Network Considerations

Handling empty/HTML responses, retries, fallbacks

Public news APIs occasionally return HTML error pages, empty bodies, or transient “query too short/long” errors. The design assumes flakiness and degrades gracefully:

- **Multi-try formats:** Attempt format=JSON first, then JSONFeed. Some gateways serve only one reliably.
- **Short, independent sub-queries:** If one sub-query fails, skip it; others still populate the cycle.
- **Length guards:** Keep each boolean compact; avoid very short naked tokens (US) unless paired with longer phrases.
- **Backoff on volume:** If a response looks truncated or fails, retry that sub-query with a lower maxrecords (e.g., 120 → 80 → 60).
- **Health logging:** On parse failure, print status code and a short *peek* (~120–300 chars) of the body to diagnose whether it’s HTML, a length warning, or a CDN error.
- **Never crash the loop:** On total failure, emit a valid “empty” result for that cycle and continue; the next cycle tries again with fresh state.
- **Politeness:** Set a modest refresh interval (e.g., 300s) and a sane per-query cap to avoid hammering the endpoint; include a normal **User-Agent** string.

DNS hardening (DoH/DoT), VPN DNS, why not hard-coding IPs

Goal: make name resolution reliable and private enough that simple DNS blocking doesn’t break the ticker.

- **System DoH (Windows 10/11 where supported):**
- # Add Cloudflare + Quad9 DoH and prevent UDP fallback
- Add-DnsClientDohServerAddress -ServerAddress 1.1.1.1 -DohTemplate https://cloudflare-dns.com/dns-query -AllowFallbackToUdp:$false
- Add-DnsClientDohServerAddress -ServerAddress 9.9.9.9 -DohTemplate https://dns.quad9.net/dns-query -AllowFallbackToUdp:$false
- Set-DnsClientServerAddress -InterfaceAlias "Ethernet" -ServerAddresses 1.1.1.1,9.9.9.9
- Set-DnsClientServerAddress -InterfaceAlias "Wi-Fi" -ServerAddresses 1.1.1.1,9.9.9.9

If those cmdlets aren't present, enable **Secure DNS** in the browser (Chrome/Edge/Firefox).

- **VPN DNS matters:** Ensure the VPN **tunnels DNS** (disable split-tunnel for browsers; enable "block outside DNS" or equivalent). Many "partial outages" are just DNS leaks to a censored or captive resolver.
- **Why not hard-code IPs?**
  News APIs sit behind CDNs; IPs **rotate** by region and time. Pinning an IP risks breakage and certificate mismatches. If you must pin for diagnostics:
  - Prefer curl --resolve api.example.com:443:IP so TLS **SNI/Host** still match.
  - Or use a temporary hosts-file entry (admin rights), but expect to update it frequently.
  - Never disable TLS verification in production.

When QUIC/ECH help; lawful/policy cautions

- **HTTP/3 (QUIC):** Runs over UDP and can ride around some TCP-shaping middleboxes. Leave it enabled in modern browsers; for API calls from PowerShell, QUIC isn't used, but the browser you open links with may benefit.
- **ECH (Encrypted ClientHello):** Hides the hostname from basic SNI filters when supported by your resolver and the destination. Keep your browser current and use a resolver that advertises ECH (e.g., major public DoH providers). This won't fix hard IP blocks, but it reduces trivial hostname filtering.
- **Recognize escalation:**
  - **DNS-only block?** DoH/DoT or VPN DNS typically resolves it.
  - **SNI/hostname block?** ECH may help; otherwise a VPN/secure proxy is needed.
  - **IP block/throttling?** You need a different path/exit (VPN), not DNS tweaks.
- **Lawful/policy cautions:**
  - Don't circumvent corporate controls without authorization.
  - Respect local laws and provider ToS.
  - Prefer reputable resolvers (clear privacy policies, malware filtering if you want it).
  - Avoid brittle hacks (permanent hosts-file edits, certificate bypass) except for short, documented diagnostics.

Practical checklist

- ✅ Use multiple short sub-queries; tolerate per-query failure.
- ✅ Retry with JSON→JSONFeed; log body peeks on failure.
- ✅ Cap maxrecords; back off if needed.
- ✅ Turn on DoH (system or browser) and ensure VPN tunnels DNS.
- ✅ Don't hard-code IPs; if you must, use --resolve and revisit often.
- ✅ Keep the loop alive on errors; print a clear cycle summary so you see health at a glance.

6. Operations

Tuning cadence and volume ($interval, $cap, adding/removing sub-queries)

Cadence

- $interval controls how often the loop refreshes (seconds).
    - **Default**: 300 (5 minutes) balances freshness and politeness to the API.
    - **Faster**: 120–180 if you need tighter awareness (during a breaking CVE).
    - **Slower**: 600–900 for "background heartbeat" on a side monitor.

Volume

- $cap limits how many rows print per cycle after dedupe and sort.
    - **Default**: 60 keeps the table readable.
    - Increase to 100–150 when you're surveying a flood day; decrease to 30 for a tight summary view.

Sub-queries

- The $SUBQS list is the main recall lever. Each entry is compact on purpose (GDELT rejects long/short booleans).
- **Add** a line when you need a new signal family, e.g.:
    - "wiper" AND ("United States" OR "U.S." OR US) sourcelang:english

    - ("EHR" OR "electronic health record") AND (ransomware OR outage) sourcelang:english

- **Remove** or **comment** a line if it's noisy (e.g., sector terms that drag in utility outages with no cyber language).
- **Per-query records**: In the helper, we request maxrecords=120. If responses get flaky, lower that to 80 or 60 for stability and rely on more sub-queries for breadth.

Strictness presets (quick flips)

- **Tighter (higher precision)**: In the cyber regex, drop data breach and vulnerab.
- **Broader (higher recall)**: Add supply chain, initial access, vendor/tool names (e.g., "ScreenConnect", "Citrix Netscaler", "MOVEit").

Reading the console output; common false positives and quick fixes

Cycle header

[2025-09-27 12:34:56] Raw: 238 Kept: 41 Dropped: 197 Domains: 28

- **Raw**: total headlines fetched across all sub-queries (before any filtering).
- **Kept**: what passed the title filter and dedupe—these will print.
- **Dropped**: filtered out (off-topic, non-U.S., ambiguous).
- **Domains**: unique outlets in the printed set—a quick diversity sanity check.

Table columns

- **Time**: parsed seendate|date_published localized to your machine.
- **Outlet**: domain from the URL—helps spot syndication vs. originals.
- **Title**: trimmed to ~120 chars.
- **Link**: click/open to read the article.

Frequent false positives & surgical fixes

1. **Law-enforcement headlines without cyber content**
    - *Symptom*: "FBI arrests…", "DHS announces…" with no cyber intent.

- **Fix**: Keep the current logic: **FBI/DHS only count when a cyber term is also present**. If you still see junk, ensure the cyber regex includes only high-signal terms (drop data breach).

2. **"NSA" collision (India's National Security Act)**
    - *Symptom*: India-focused headlines with "NSA" but not the U.S. agency.
    - **Fix**: Keep matching **full name** ("National Security Agency") or strong-agency list; **do not** accept bare \bNSA\b.
3. **Generic outages ("internet down", "airport outage")**
    - *Symptom*: Service outages that aren't labeled as cyber.
    - **Fix**: In sub-queries about critical infrastructure, **require** an attack verb (e.g., (malware OR intrusion OR ransomware OR ddos))—already present in the default.
4. **Sports/politics bleed-through**
    - *Symptom*: A stray non-cyber headline sneaks in via an agency token in the title.
    - **Fix**: Tighten agency gating:
        - Strong agencies (CISA/USCYBERCOM/NSA full name) → allowed.
        - General agencies (FBI/DHS) → **must** pair with a cyber term.
5. **Overly broad "data breach"**
    - *Symptom*: Compliance/PR pieces that aren't actionable.
    - **Fix**: Remove data breach from the cyber regex or replace with data breach.*(ransom|extortion|stolen|exposed).
6. **Too quiet—few or zero results**
    - *Symptom*: Kept: 0 for multiple cycles but you expect activity.
    - **Checklist**:
        - Confirm network (try opening some **Link** URLs manually).
        - Temporarily broaden cyber regex (re-add data breach, vulnerab).
        - Add a sector sub-query (e.g., healthcare, education).
        - Shorten $interval or raise $cap to see more.

7. **Too chatty—signal buried**
    - *Symptom*: High Kept with low uniqueness of domains (syndication flood).
    - **Fix**:
        - Lower $cap to 40–50.
        - Add session-level dedupe (URL TTL 24h) to hide repeats across cycles.
        - Tighten the regex to ransomware|ddos|cve-\d{4}-\d+|zero[-]?day.

Session-level dedupe (optional quick add)

If repeat URLs across cycles bug you, add the in-memory set (TTL 24h) described in **Filtering Logic**. It prevents déjà vu without writing files and is one small block of code.

Performance & etiquette

- **Be polite to the API**: keep $interval ≥ 180 and per-query maxrecords ≤ 120.
- **Watch the header**: sudden spikes in Dropped with body-peek logs (if enabled) usually mean a temporary format/length hiccup—no action needed.
- **Network tuning**: If you see intermittent failures, enable system or browser **DoH**, and make sure your VPN tunnels DNS (no split-tunnel leaks).

One-minute tuning workflow

1. Let the loop run for 2–3 cycles.
2. Note off-topic titles.
3. Tweak **one** of: a) remove a noisy sub-query, b) tighten the regex, or c) lower $cap.
4. Observe the next cycle's **Raw/Kept/Dropped/Domains**.
5. Repeat until "Kept" feels right for your purpose (often 30–60 with 15–35 domains).

7. Security & Ethics

Neutral presentation vs. bias claims

Goal: show fresh, relevant **signals** without prescribing interpretation.

- **Method transparency beats mystique.** We publish the exact sub-queries and regex filters. Anyone can see *why* an item appeared and adjust the logic. This reduces hidden editorial bias compared with opaque scoring or sentiment models.
- **Headline-only caveat.** Titles are marketing copy; they can amplify framing bias. By not inferring sentiment or "truth," we avoid laundering that bias into numeric scores—but readers should still click through.
- **Sampling bias is inevitable.** GDELT's corpus, our **English-only** scope, and the **U.S.-centric** filters skew the feed. Document this explicitly and, if needed, maintain alternative profiles (e.g., "Global/Multilingual," "OT-only") to compare outputs.
- **Balanced recall vs. precision.** Tight filters (e.g., require ransomware|DDoS|CVE) reduce noise but may miss emerging classes of incidents. Publish preset variants ("strict," "balanced," "broad") so users knowingly pick their bias.
- **Governance & change control.** Track changes to $SUBQS and regex in a simple changelog with a one-line rationale ("added 'Volt Typhoon' after new campaign"). This prevents silent drift and enables periodic bias reviews.

Privacy notes; avoiding scraped credentials/cookies; API rate considerations

Data minimization

- **No accounts, no cookies, no tokens.** All requests are anonymous HTTPS GETs to a public endpoint; we do not attach session cookies or bearer tokens.
- **No page scraping.** We fetch **metadata lists** only (title, URL, timestamps). We do **not** crawl the linked pages, avoid paywall workarounds, and do not collect author names, emails, or other PII.
- **Ephemeral state.** The live console version keeps de-dupe memory **in RAM** with a 24-hour TTL. There's no disk persistence unless the operator opts in.

Operational security

- **TLS everywhere.** Calls are HTTPS; do **not** disable certificate validation. If you must pin an IP for diagnostics, prefer curl --resolve so SNI and certs still match.

- **DNS privacy (optional).** Enabling DoH/DoT reduces passive DNS exposure; use reputable resolvers with clear privacy policies.
- **Logging hygiene.** Console output shows domains and titles only. Avoid piping raw responses to logs; they can inadvertently capture transient error pages with unexpected data.

Respect for services and terms

- **Rate & cadence.** Default cadence is conservative (300s refresh; ≤120 records per sub-query). If you lower the interval, add **jitter** (±10–20%) or a small backoff on failure to avoid hammering. Example: sleep ($interval + Get-Random -Min -10 -Max 10).
- **Error handling.** On HTML/empty bodies, skip and continue; do not loop at high speed on failure. This protects the endpoint and your own IP reputation.
- **Robots/ToS alignment.** We query only the documented API; we do not spider publisher sites. If you later add fetching of article bodies, recheck robots.txt, paywalls, and ToS.

Compliance & policy boundaries

- **Don't bypass enterprise controls** without authorization (VPN, DNS policies, inspection). Provide a policy-friendly mode (system DNS, no obfuscation).
- **Jurisdictional awareness.** Some network features (obfuscation, circumvention) may be restricted. Keep the default network posture conventional (standard HTTPS/DNS).
- **Use of signals.** This feed is for **situational awareness**, not attribution or doxxing. Avoid naming individuals or inferring breach victims beyond what outlets publish.

Ethical extensions (if you evolve the tool)

- **Provenance cues.** Consider optional labels like "wire," "trade press," or "government advisory" (derived from domains) to help readers weigh items—without asserting "reliability."
- **Counter-framing.** If you add an "outlier pairing" view, present it as diversity of coverage—not as adjudication.
- **Reproducibility.** Output the exact sub-queries and timestamps used each cycle so another analyst can rerun the same window and compare.

Bottom line: Keep it minimal, auditable, and polite to the network. Neutral presentation plus transparent mechanics lets users apply their own judgment—while privacy-preserving defaults and modest rate limits keep the tool safe to run anywhere.

8. Evaluation

Example cycles and effect of stricter/looser filters

Below are **illustrative** (but realistic) snapshots from three presets run over identical 30-minute windows. Each cycle polls GDELT with the same sub-queries; only the **title filter** changes.

Preset A — Strict (high precision)

- **Filter:** ransomware|ddos|cve-\d{4}-\d+|zero[-]?day|intrusion (drops data breach, vulnerab, phishing)
- **Cycle 1:** Raw 230 → Kept 26 → Dropped 204 → Domains 18
- **Cycle 2:** Raw 215 → Kept 23 → Dropped 192 → Domains 17
- **Cycle 3:** Raw 241 → Kept 28 → Dropped 213 → Domains 21
- **Feel:** Very actionable; low chatter. May miss advisory framing without explicit exploit terms.

Preset B — Balanced (default)

- **Filter:** Adds data breach|phishing|vulnerab stems; FBI/DHS require a cyber term; strong agencies always pass
- **Cycle 1:** Raw 230 → Kept 44 → Dropped 186 → Domains 29
- **Cycle 2:** Raw 215 → Kept 41 → Dropped 174 → Domains 27
- **Cycle 3:** Raw 241 → Kept 47 → Dropped 194 → Domains 33
- **Feel:** Good day-to-day signal-to-noise. Mix of advisories, exploited CVEs, and major incidents.

Preset C — Broad (high recall)

- **Filter:** Balanced + sector verbs (outage/incident) and more vendor/tool keywords
- **Cycle 1:** Raw 230 → Kept 78 → Dropped 152 → Domains 46
- **Cycle 2:** Raw 215 → Kept 72 → Dropped 143 → Domains 41
- **Cycle 3:** Raw 241 → Kept 85 → Dropped 156 → Domains 49
- **Feel:** Great for scanning; requires more analyst triage (more false positives, e.g., non-cyber outages).

What changes between presets

- **Kept count** rises with broader filters (more recall).
- **Domains** generally increase with recall (more syndication and niche sources surface).
- **Novelty rate** (items not seen in previous cycle) is highest in Broad; Strict shows steadier, smaller deltas.
- **Time-to-signal** for big advisories is similar across presets; Strict can miss early "warning" titles that lack explicit attack terms.

Precision vs. recall trade-offs; typical domains surfaced

Precision vs. recall (how to choose)

- **Need a quiet, high-confidence feed?** Use **Strict**. You'll mostly see:
    - Government advisories with CVE identifiers ("CISA adds CVE-2025-…")
    - Confirmed ransomware/DDoS events ("City X confirms ransomware attack…")
    - Exploitation alerts ("Zero-day exploited against…")
- **Need situational awareness for a SOC shift?** Use **Balanced**. You'll capture:
    - The above, plus notable breaches, major phishing campaigns, and sector-specific alerts.
- **Hunting or research mode?** Use **Broad**. Expect:
    - More sector outages with ambiguous root cause, vendor bulletins, regional press—useful for leads, noisier for operations.

Dialing knobs

- Tighten: remove data breach and vulnerab; demand cve-\d{4}-\d+|ransomware|ddos|zero[-]?day.
- Loosen: add sector terms (e.g., hospital/port/airport/school) or product names (ScreenConnect, Citrix NetScaler, MOVEit).
- Keep FBI/DHS **AND-with-cyber** to avoid generic law-enforcement headlines.
- Avoid bare **NSA** token; rely on **National Security Agency** (full) or other strong U.S. cyber agencies.

Typical domains surfaced (by bucket)

(Varies by news day; below is the common shape rather than a fixed list.)

- **Government/advisory:** cisa.gov bulletins and blog posts; occasional whitehouse.gov cyber fact sheets; state/local agency sites during incidents.
- **Wires & majors:** Reuters, AP, major U.S. dailies when an incident makes general news.
- **Trade/infosec press:** well-known security outlets and industry trades covering CVEs, ransomware groups, and advisories.
- **Regional/local news:** municipal/school/hospital incidents frequently show up here first.
- **Vendor blogs:** rapid advisories/patch posts during widely exploited CVEs.

Diversity check: The **Domains** count in the cycle header is a quick proxy for breadth. Balanced runs often show 25–35 unique outlets per 60 items; Strict closer to 15–25; Broad 40–50+.

Suggested evaluation routine (repeatable)

1. **Pick two presets** (e.g., Strict vs. Balanced).
2. **Run for one workday** at $interval = 300, saving the console output per cycle.
3. **Sample 50 kept items** from each preset; label TP/FP for your environment (simple binary: “actionable to us?”).
4. Compute:
    - **Precision** = TP / Kept
    - **Volume** = Kept per cycle
    - **Domain diversity** = unique outlets per cycle
    - **Novelty** = % not seen previous cycle (optional session-level dedupe helps)
5. **Adjust filters** (e.g., drop data breach or add sector terms) and rerun. Two iterations usually land on a comfortable operating point.

Interpreting anomalies

- **Raw rises, Kept flat:** likely a noisy news hour; filters doing their job.
- **Raw flat, Kept drops across cycles:** filter too strict or an agency token removed—review regex edits.

- **Domains spike with Kept:** syndication wave—lower $cap or enable cross-cycle dedupe to reduce fatigue.
- **Kept = 0 repeatedly:** network or API issue; broaden filter briefly to confirm, then check DNS/VPN settings.

Bottom line: Treat the ticker like a scope with adjustable zoom. Strict gives crisp shots of confirmed incidents; Broad shows the whole field. Use the cycle header and quick labeling passes to tune for your team's tolerance and goals.

9. Limitations & Future Work

Title-only blind spots; multilingual expansion; topic packs

Title-only blind spots (today)

- Headlines can be **click-baity or vague** ("Major incident at utility") while the body clarifies non-cyber root cause.
- Some advisories **bury the lede** (no "CVE" or "ransomware" in the title), so our title regex misses them.
- Syndication/rewrites cause **slight title drift** that evades within-cycle de-duplication.

Near-term mitigations

- Add a **single field from the feed** when available (e.g., summary/excerpt) without fetching full pages; pass it through the same regex. Keep opt-in to preserve speed.
- Introduce **near-duplicate keys** (e.g., host + first 60 chars of title or a MinHash/SimHash of the title) to collapse trivial rewrites.
- Maintain a tiny list of **high-value exceptions** (e.g., "CISA adds to KEV") that bypass strict cyber regex when the domain is authoritative.

Multilingual expansion (future option)

- Use GDELT's sourcelang:* switch to add **parallel sub-queries** (es, fr, de, he, ar, ru…).
- Maintain **per-language regex packs** (translated stems of ransomware/DDoS/CVE terms and local agency names).
- Optional: pipe **title-only** through an on-device/offline translator or a local service and re-use the English filter (acknowledging translation noise).

- Start with **bilingual pilots** (e.g., English+Spanish for U.S. coverage) and measure precision before broadening.

Topic packs (modularity)

- Parameterize $SUBQS and regex into **named packs** you can toggle:
    - us-cyber-ops (current), ot-ics, healthcare, education, cloud-saas, election-security.
- Represent packs as a small **YAML** (human-editable):
- pack: us-cyber-ops
- subqueries:
- - '"ransomware" AND ("United States" OR "U.S." OR US) sourcelang:english'
- - '(CISA OR "US Cyber Command" OR "National Security Agency") AND (advisory OR alert) sourcelang:english'
- filters:
- cyber_regex: '\b(ransomware|ddos|cve-\d{4}-\d+|zero[-]?day|intrusion|malware)\b'
- us_context: '\b(United States|U\.S\.|USA|American|US)\b'
- strong_agency: '\b(CISA|US Cyber Command|United States Cyber Command|National Security Agency)\b'
- cap: 60
- interval: 300
- In PowerShell, load the pack once per session; hot-swap by re-pasting a new YAML block.

Optional persistence (SQLite) and web UI variant

Why add persistence (optional)

- **Across-cycle memory** that survives restarts (e.g., “don’t show this URL again for 24h”).
- **Light analytics:** trend of Raw/Kept/Dropped, outlet diversity, top CVEs over time.
- **Export/share:** produce a JSON/CSV feed for teammates or a dashboard.

Minimal schema (SQLite)

items(id TEXT PRIMARY KEY, time DATETIME, outlet TEXT, title TEXT, url TEXT, pack TEXT)

seen(url TEXT PRIMARY KEY, first_seen DATETIME, last_seen DATETIME)

cycles(ts DATETIME PRIMARY KEY, pack TEXT, raw INT, kept INT, dropped INT, domains INT)

- Insert **kept** rows into items; upsert URL into seen; one summary row per cycle into cycles.
- Apply a **retention policy** (e.g., 7–30 days) to avoid unbounded growth.
- Keep persistence **opt-in** so the default remains fileless.

Web UI variant (tiny, auditable)

- **Backend:** a 100–150-line FastAPI/Flask app or Node/Express that reads items/cycles and exposes /api/items?since=…&pack=….
- **Frontend:** static HTML + HTMX/Alpine.js for interactivity; or a small React page.
- **Features:** pack selector, time range, search within titles, outlet facets, and an **auto-refresh** toggle.
- **Deployment:** a single container or Fly.io/Render; no state beyond the SQLite file mounted on disk.
- **Security:** read-only API, CORS locked to your domain, optional basic auth.

Pairing contradictory headlines (“outlier pairing” strip)

Aim
Surface **divergent takes** on the same event without scoring “truth” or assigning reliability. Present them side-by-side so a reader can compare frames.

Heuristic (no ML baseline)

1. **Bucket by entity/time:**
    - Extract quick entities with regex heuristics (ORGs like CISA, Microsoft, “City of …”, CVE IDs, ransomware group names).
    - Group items sharing at least one specific anchor (same **CVE**, same **org**, same **city**, or same **actor**) within a time window (e.g., ±6h).

2. **Detect framing tension:**
    - Look for **negation/stance verbs** in titles:
      denies|confirms|claims|admits|disputes|refutes|blames|warns|downplays.
    - Identify **magnitude words**: massive|widespread|limited|minor.
    - Opposing verbs/magnitude terms across items in the same bucket → candidate "outliers".
3. **Source diversity gate:**
    - Require two **different outlet families** (wire vs. local vs. trade press vs. vendor/government).
    - Optional: ensure one is an **official advisory** (e.g., cisa.gov) and the other is a **press headline**.

Presentation

- A slim "Outlier Pair" strip under the table:
- CVE-2025-12345 | cisa.gov: "CISA adds CVE-2025-12345 to KEV; known exploited"
- | Vendor blog: "Exploit requires uncommon config; impact limited"
- Clicking either takes the user to the article; no sentiment, no verdict.

Future refinement (optional & transparent)

- **Lightweight similarity:** use TF-IDF on titles (no cloud calls) to cluster near-duplicates and avoid false pairings.
- **Embeddings (opt-in):** if you later allow local ML or a service, compute small-dim embeddings for better event grouping; still show the **exact rules** used to flag outliers.
- **User feedback:** allow "hide this pair" or "good pair" toggles; log anonymously to improve heuristics.

Other roadmap ideas

- **Alerts:** optional desktop toast when a title matches a "priority CVE list" or "your org name."
- **Sector packs:** prebuilt YAMLs for healthcare, energy, education, municipalities.

- **Rate etiquette:** add **jitter** to $interval and slim per-query maxrecords during busy hours.
- **Health checks:** a one-liner /api/health that returns last cycle's counts and time.
- **Plug-in filter slots:** expose a single PowerShell function Is-OnTopic($title) so teams can drop in their own policy without touching the fetcher.

Bottom line: keep defaults tiny and explainable; make every enhancement **opt-in**, text-based, and auditable. That preserves the tool's core promise—fast, neutral signals—while giving teams room to evolve it toward their environment.

10. Conclusion

What this ticker *is*
A tiny, transparent **situational-awareness tool**. It surfaces last-24-hour **U.S.-focused cyber signals** (ransomware, DDoS, CVEs, official advisories) with **readable rules** you can inspect and tune in minutes. It's designed for **speed, neutrality, and portability**: no accounts, no persistence, no heavy back end—just short sub-queries, a simple title filter, and a console table you can keep open all day.

What this ticker is *not*

- Not a fact-checker, not an attribution engine, and not a replacement for incident response tooling.
- Not a sentiment/reliability scorer; it deliberately avoids numeric judgments.
- Not a crawler of full articles or a feed that guarantees completeness; it accepts the trade-offs of **title-only** filtering for performance and auditability.

Where it shines

- **First look / triage:** sense breaking CISA advisories, mass-exploited CVEs, and notable U.S. incidents within minutes.
- **Low-friction monitoring:** side-monitor heartbeat for SOC shifts, small teams, jump boxes, or lab VMs.
- **Auditable filters:** fast iterations when stakeholders disagree about scope or bias.

When to graduate to fuller TI stacks
Move up the stack when your needs exceed "fast, neutral headlines":

- **You need depth:** full-text ingestion, entity extraction (victim/sector/actor), IOC harvesting, and correlation with SIEM/SOAR.
- **You need confidence:** scoring models, dedup across sources at article/body level, source quality profiling, and enrichment (VT, WHOIS, passive DNS).
- **You need workflows:** alerting, ticketing, case management, TLP handling, watchlists, and integrations (MISP/STIX/TAXII).
- **You need governance:** multi-user roles, audit logs, change control over queries/filters, and long-term metrics (precision/recall, coverage).
- **You need scale:** multi-language coverage, historical analytics, and SLAs on availability and throughput.

Pragmatic migration path

1. **Keep the ticker** as your real-time "front door."
2. **Add optional persistence** (SQLite) for cross-cycle memory and simple trend charts.
3. **Introduce a thin web UI** for sharing and light search.
4. **Integrate** with your SIEM or CTI platform for enrichment and alerting on priority patterns (e.g., KEV additions, sector-specific CVEs).
5. **Evaluate** commercial/open-source TI when you consistently need features the ticker won't do without sacrificing its simplicity.

Bottom line

This ticker is the **scout**, not the army. It gets the right headlines in front of humans quickly, without hidden bias or operational drag. Use it to stay oriented, decide what deserves deeper analysis, and only then bring heavier intelligence tooling to bear.

11. Appendix A — Listing 1: Live Console Script

```
# Live console ticker: US cyber warnings/attacks (last 24h, GDELT) — exit with Ctrl+C

$api='https://api.gdeltproject.org/api/v2/doc/doc'

$headers=@{ 'User-Agent'='Mozilla/5.0 CyberUSTicker/1.6' }

$interval=300    # seconds between refreshes

$cap=60          # max rows to display per cycle
```

```
# ---- helpers ----

$CYBER_RE  = [regex]::new('\b(cyber ?attack|ransomware|data
breach|intrusion|ddos|malware|zero[ -]?day|phishing|exfiltrat|backdoor|vulnerab|cve-\d{4}-
\d+)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)

$US_PHRASE = [regex]::new('\b(United
States|U\.S\.|USA|American)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)

$US_UP     = [regex]::new('\bUS\b') # case-sensitive, avoids pronoun "us"

$AGENCY_STRICT = [regex]::new('\b(Cybersecurity and Infrastructure Security Agency|US Cyber
Command|United States Cyber Command|National Security
Agency|CISA)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)

$AGENCY_LOOSE  =
[regex]::new('\b(FBI|DHS)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)


function Is-OnTopic([string]$t){

  $hasCyber = $CYBER_RE.IsMatch($t)

  $hasStrict= $AGENCY_STRICT.IsMatch($t)

  $hasLoose = $AGENCY_LOOSE.IsMatch($t)

  $hasUS    = $US_PHRASE.IsMatch($t) -or $US_UP.IsMatch($t) -or $hasStrict

  # keep if: (cyber AND (US or strict agency)) OR strict agency alone OR (loose agency AND cyber)

  return (($hasCyber -and ($hasUS -or $hasStrict)) -or $hasStrict -or ($hasLoose -and $hasCyber))

}

function Parse-Iso([string]$s){ try{ [datetimeoffset]::Parse($s).ToLocalTime().DateTime }catch{
Get-Date } }

function Host([string]$u){ try{ ([uri]$u).Host }catch{''} }


function Get-Gdelt($q,$fmt='JSON',$max=120){


$params=@{query=$q;mode='artlist';timespan='24h';maxrecords="$max";sort='datedesc';form
at=$fmt}
```

```
  try { return Invoke-RestMethod -Uri $api -Headers $headers -Body $params -Method Get -TimeoutSec 30 -ErrorAction Stop }
  catch {
    try {
      $r=Invoke-WebRequest -Uri $api -Headers $headers -Body $params -Method Get -TimeoutSec 30 -ErrorAction Stop
      $t=($r.Content??'').Trim(); if(-not $t -or $t.StartsWith('<')){ return $null }; return $t|ConvertFrom-Json
    } catch { return $null }
  }
}

# Short sub-queries to avoid GDELT length limits
$SUBQS=@(
  '"ransomware" AND ("United States" OR "U.S." OR US) sourcelang:english',
  '"data breach" AND ("United States" OR "U.S." OR US) sourcelang:english',
  '"cyberattack" AND ("United States" OR "U.S." OR US) sourcelang:english',
  '"intrusion" AND ("United States" OR "U.S." OR US) sourcelang:english',
  'DDoS AND ("United States" OR "U.S." OR US) sourcelang:english',
  '("critical infrastructure" OR "power grid" OR pipeline OR water) AND (malware OR attack OR intrusion) sourcelang:english',
  '(LockBit OR ALPHV OR BlackCat OR Clop OR "Volt Typhoon" OR Sandworm OR "APT29") AND ("United States" OR "U.S." OR US) sourcelang:english',
  '("Cybersecurity and Infrastructure Security Agency" OR "US Cyber Command" OR "National Security Agency" OR CISA) AND (advisory OR alert OR warning OR directive) sourcelang:english'
)

function Invoke-Cycle {
```

```
  $col=@{};$raw=0;$kept=0;$drop=0
  foreach($q in $SUBQS){
    $d=Get-Gdelt $q 'JSON' 120; if(-not $d){ $d=Get-Gdelt $q 'JSONFeed' 120 }; if(-not $d){
continue }
    $list= if($d.articles){$d.articles}elseif($d.items){$d.items}else{@()}
    foreach($it in $list){
      $title=($it.title??'').Trim(); $url=($it.url??'').Trim()
      if(-not $title -or -not $url){ continue }
      $raw++
      if(-not (Is-OnTopic $title)){ $drop++; continue }
      $key="$title`n$url"
      if(-not $col.ContainsKey($key)){
        $ts=$it.seendate; if(-not $ts){ $ts=$it.date_published ?? $it.date_modified ?? '' }
        $col[$key]=[pscustomobject]@{ Time=Parse-Iso $ts; Outlet=Host $url; Title=$title; URL=$url
}
        $kept++
      }
    }
  }

  $items=$col.Values|Sort-Object Time -Descending
  if($cap -gt 0){ $items=$items|Select-Object -First $cap }

  $doms=($items.Outlet|Select-Object -Unique).Count
  Write-Host ("`n[{0}] Raw: {1}   Kept: {2}   Dropped: {3}   Domains: {4}" -f (Get-Date -Format
'yyyy-MM-dd HH:mm:ss'), $raw,$kept,$drop,$doms) -ForegroundColor Cyan
```

```
  if($items.Count -eq 0){ Write-Host "No on-topic headlines this cycle." -ForegroundColor
DarkGray; return }

  $items | ForEach-Object {
    [pscustomobject]@{
      Time   = $_.Time.ToString('yyyy-MM-dd HH:mm')
      Outlet = $_.Outlet
      Title  = if($_.Title.Length -gt 120){ $_.Title.Substring(0,120)+'…' } else { $_.Title }
      Link   = $_.URL
    }
  } | Format-Table -AutoSize
}

Write-Host "Live US-cyber console ticker running. Press Ctrl+C to exit.`n" -ForegroundColor
Green
while($true){
  Invoke-Cycle
  for($s=$interval;$s -gt 0;$s--){
    $pct=[int](100*($interval-$s)/[double]$interval)
    Write-Progress -Activity "Next refresh in $s sec" -PercentComplete $pct
    Start-Sleep -Milliseconds 1000
  }
  Write-Progress -Activity "Next refresh" -Completed
}
```

---

12. Appendix B — Tunables (Copy-Paste Ready)

B.1 Cyber regex presets (pick ONE; replace your $CYBER_RE)

Strict (high precision)

```
$CYBER_RE = [regex]::new('\b(cyber ?attack|ransomware|ddos|malware|zero[ -]?day|intrusion|cve-\d{4}-\d+)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)
```

Balanced (default)

```
$CYBER_RE = [regex]::new('\b(cyber ?attack|ransomware|data breach|intrusion|ddos|malware|zero[ -]?day|phishing|exfiltrat|backdoor|vulnerab|cve-\d{4}-\d+)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)
```

Broad (high recall)

```
$CYBER_RE = [regex]::new('\b(cyber ?attack|ransomware|data breach|intrusion|ddos|malware|zero[ -]?day|phishing|exfiltrat|backdoor|vulnerab|supply chain|wiper|initial access|credential ?stuff|botnet|cve-\d{4}-\d+)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)
```

Tip: If “data breach” generates too much chatter, drop it or anchor it (e.g., data breach.*(ransom|extortion|stolen|exposed)).

B.2 U.S. / agency patterns (drop-in replacements)

```
# U.S. context

$US_PHRASE = [regex]::new('\b(United States|U\.S\.|USA|American)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)

$US_UP     = [regex]::new('\bUS\b')  # case-sensitive, avoids pronoun "us"


# Strong U.S. cyber agencies (always acceptable)

$AGENCY_STRONG = [regex]::new('\b(Cybersecurity and Infrastructure Security Agency|CISA|US Cyber Command|United States Cyber Command|National Security Agency)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)


# General agencies that require a cyber term in the same title
```

```
$AGENCY_LOOSE  = [regex]::new('\b(FBI|DHS|Department of Homeland Security|Justice Department|DoJ)\b',[Text.RegularExpressions.RegexOptions]::IgnoreCase)


# Gate function (replace your Is-OnTopic with this if desired)
function Is-OnTopic([string]$t){
  $hasCyber  = $CYBER_RE.IsMatch($t)
  $hasStrong = $AGENCY_STRONG.IsMatch($t)
  $hasLoose  = $AGENCY_LOOSE.IsMatch($t)
  $hasUS     = $US_PHRASE.IsMatch($t) -or $US_UP.IsMatch($t) -or $hasStrong
  return (($hasCyber -and ($hasUS -or $hasStrong)) -or $hasStrong -or ($hasLoose -and $hasCyber))
}
```

B.3 Sample sub-queries by sector (append to or swap into $SUBQS)

Keep each line compact; GDELT rejects long/short booleans. Include sourcelang:english.

Healthcare

```
$SUBQS += @(
  '("hospital" OR "clinic" OR "EHR" OR "electronic health record") AND (ransomware OR outage OR malware) sourcelang:english',
  '(HHS OR "Health and Human Services") AND (advisory OR alert OR warning) sourcelang:english'
)
```

Education (K-12 / higher ed)

```
$SUBQS += @(
  '("school district" OR "unified school" OR university OR college) AND (ransomware OR ddos OR intrusion) sourcelang:english'
)
```

Energy / Water / OT

```
$SUBQS += @(
  '("power grid" OR substation OR pipeline OR "water utility" OR SCADA OR ICS OR "OT network") AND (malware OR intrusion OR ransomware OR ddos) sourcelang:english',
  '("industrial control" OR PLC OR HMI) AND (vulnerability OR exploit OR cve) sourcelang:english'
)
```

State & Local / Municipal

```
$SUBQS += @(
  '("city hall" OR municipality OR county OR township) AND (ransomware OR data breach OR intrusion) sourcelang:english'
)
```

Cloud / SaaS / Identity

```
$SUBQS += @(
  '(Okta OR Azure OR Microsoft 365 OR Google Workspace OR Salesforce) AND (breach OR intrusion OR token OR session) sourcelang:english',
  '(Citrix OR Netscaler OR ScreenConnect OR MOVEit) AND (cve OR exploit OR ransomware) sourcelang:english'
)
```

Volume control: if a sector is noisy, comment out (#) or remove a line; if too quiet, add one focused verb or product term.

---

13. Appendix C — Deployment Variants

C.1 Scheduled Task (Windows)

This variant assumes you've saved the script as C:\tools\us-cyber-ticker.ps1. The task runs at logon and restarts on failure. PowerShell 7 is pwsh.exe.

```
$scriptPath = 'C:\tools\us-cyber-ticker.ps1'
$action = New-ScheduledTaskAction -Execute 'pwsh.exe' -Argument ('-NoLogo -NoProfile -ExecutionPolicy Bypass -File "{0}"' -f $scriptPath)
```

$trigger = New-ScheduledTaskTrigger -AtLogOn

$settings = New-ScheduledTaskSettingsSet -AllowStartIfOnBatteries -DontStopIfGoingOnBatteries -RestartCount 999 -RestartInterval (New-TimeSpan -Minutes 1)

Register-ScheduledTask -TaskName 'US-Cyber-Ticker' -Action $action -Trigger $trigger -Settings $settings -Description 'Live US cyber headlines via GDELT (console ticker)'

Optional (start now):

Start-ScheduledTask -TaskName 'US-Cyber-Ticker'

If you prefer Task Scheduler UI: Create Task → Triggers: At log on → Actions: Start a program pwsh.exe with arguments -NoLogo -NoProfile -ExecutionPolicy Bypass -File "C:\tools\us-cyber-ticker.ps1" → Settings: restart on failure.

C.2 Loop-with-logging variant

Writes a rolling log (CSV-ish) while still printing to console. Keep $interval polite (≥180s).

```
# Add near the top:

$log = 'C:\tools\us-cyber-ticker.log'


# After you materialize $items each cycle:

$stamp = Get-Date -Format 'yyyy-MM-dd HH:mm:ss'

$header = "[{0}] Raw={1} Kept={2} Dropped={3} Domains={4}" -f $stamp,$raw,$kept,$drop,(($items.Outlet|Select-Object -Unique).Count)

$header | Tee-Object -FilePath $log -Append | Out-Null


$items | ForEach-Object {

  '{0},{1},{2},"{3}","{4}"' -f $stamp, $_.Time.ToString('yyyy-MM-dd HH:mm'), $_.Outlet,

    ($_.Title -replace '"','""'), $_.URL

} | Tee-Object -FilePath $log -Append | Out-Null
```

For analysis later, open the log in Excel / Power BI, or Import-Csv with a custom header.

C.3 Minimal Linux/macOS adaptation notes

Option 1: Run the same PowerShell script with pwsh

- Install PowerShell 7 (Linux/macOS).
- Run: pwsh -NoLogo -NoProfile -ExecutionPolicy Bypass -File ./us-cyber-ticker.ps1
- Use systemd --user (Linux) or launchd (macOS) to auto-start.

Option 2: Tiny Bash + curl + jq (one-shot demo)
(This is a minimal, single-subquery example to prove the endpoint and show a table; it lacks your full filters/dedup.)

```
#!/usr/bin/env bash

Q='"ransomware" AND ("United States" OR "U.S." OR US) sourcelang:english'

URL="https://api.gdeltproject.org/api/v2/doc/doc?mode=artlist&timespan=24h&format=JSON
&sort=datedesc&maxrecords=120&query=$(python3 -c "import
urllib.parse,sys;print(urllib.parse.quote(sys.argv[1]))" "$Q")"

curl -sS "$URL" | jq -r '

  .articles[]? | [.seendate, (.url|capture("https?://(?<h>[^/]+)/").h), .title, .url] |

  @tsv

' | head -n 50 | awk -F '\t' 'BEGIN{OFS="\t"; printf("%-16s %-25s %-80s
%s\n","Time","Outlet","Title","Link")} {printf("%-16s %-25s %-80s
%s\n",$1,$2,substr($3,1,78),$4)}'
```

macOS launchd (quick template)

```
<!-- ~/Library/LaunchAgents/com.example.us-cyber-ticker.plist -->

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">

<plist version="1.0"><dict>

  <key>Label</key><string>com.example.us-cyber-ticker</string>

  <key>ProgramArguments</key>
```

```
  <array>
    <string>/usr/local/bin/pwsh</string>
    <string>-NoLogo</string>
    <string>-NoProfile</string>
    <string>-ExecutionPolicy</string><string>Bypass</string>
    <string>-File</string><string>/Users/you/tools/us-cyber-ticker.ps1</string>
  </array>
  <key>RunAtLoad</key><true/>
  <key>KeepAlive</key><true/>
  <key>StandardOutPath</key><string>/Users/you/tools/us-cyber-ticker.out.log</string>
  <key>StandardErrorPath</key><string>/Users/you/tools/us-cyber-ticker.err.log</string>
</dict></plist>
```

Load it: launchctl load ~/Library/LaunchAgents/com.example.us-cyber-ticker.plist

Linux systemd (user service)

```
# ~/.config/systemd/user/us-cyber-ticker.service
[Unit]
Description=US Cyber Ticker

[Service]
ExecStart=/usr/bin/pwsh -NoLogo -NoProfile -ExecutionPolicy Bypass -File /home/you/tools/us-cyber-ticker.ps1
Restart=always
RestartSec=5

[Install]
WantedBy=default.target
```

Enable & start:
systemctl --user enable --now us-cyber-ticker.service

Quick reference (what to tweak most often)

- **Cadence:** $interval (e.g., 300)
- **Rows per cycle:** $cap (e.g., 60)
- **Scope:** add/remove lines in $SUBQS
- **Strictness:** swap $CYBER_RE preset (Strict/Balanced/Broad)
- **US/agency rules:** use B.2 block to tighten/loosen gating

If you want, I can drop your pasted Listing 1 into Appendix A and cross-reference the exact places to change each tunable with line numbers.

You said:

Please create a Reference section.

ChatGPT said:

Thought for 9s

14. References

Data source & API

1. Leetaru, K. *The GDELT Project* — global news knowledge graph and APIs. https://www.gdeltproject.org/
2. GDELT DOC 2.0 API — "doc" (article list) endpoint parameters and formats. https://api.gdeltproject.org/api/v2/doc/
3. GDELT Blog — usage notes, limits, and examples for DOC 2.0 queries. https://blog.gdeltproject.org/

Government guidance (operations & resilience)

4. CISA. *Stop Ransomware* (advisories, KEV catalog, mitigations). https://www.cisa.gov/stopransomware
5. CISA. *Known Exploited Vulnerabilities (KEV) Catalog.* https://www.cisa.gov/kev
6. CISA / FBI / NSA joint cybersecurity advisories (JSAs). https://www.cisa.gov/news-events/cybersecurity-advisories

Windows hardening (mentioned in the paper)

7. Microsoft. *Microsoft Defender Attack Surface Reduction (ASR) rules.* https://learn.microsoft.com/windows/security/threat-protection/microsoft-defender-attack-surface-reduction
8. Microsoft. *Controlled folder access (ransomware protection).* https://learn.microsoft.com/microsoft-365/security/defender-endpoint/controlled-folders
9. Microsoft. *Windows Defender Application Control (WDAC).* https://learn.microsoft.com/windows/security/application-security/application-control/windows-defender-application-control
10. Sysinternals/Microsoft. *Sysmon* (system monitoring). https://learn.microsoft.com/sysinternals/downloads/sysmon

Networking & name resolution

11. Microsoft. *DNS over HTTPS (DoH) on Windows.* https://learn.microsoft.com/windows/privacy/dns-over-https
12. Cloudflare. *1.1.1.1 / WARP resolver & app docs.* https://developers.cloudflare.com/1.1.1.1/
13. Quad9. *Secure Recursive DNS.* https://www.quad9.net/
14. IETF. *HTTP/3 (QUIC) overview & RFCs.* https://www.rfc-editor.org/rfc/rfc9114
15. IETF. *Encrypted Client Hello (ECH) drafts & guidance.* https://datatracker.ietf.org/wg/tls/documents/

Operational/OSINT practice

16. Bazzell, M. *Open Source Intelligence Techniques* (general OSINT tradecraft). [Publisher site]
17. Bellingcat. *OSINT Research Methodology* (source evaluation, bias awareness). https://www.bellingcat.com/resources/how-tos/

Implementation odds & ends

18. PowerShell 7 Documentation. https://learn.microsoft.com/powershell/
19. HTMX / Alpine.js (for optional web UI). https://htmx.org/ ; https://alpinejs.dev/
20. SQLite. *Self-contained SQL database engine.* https://sqlite.org/