Salty2FA & Tycoon2FA: Hybrid Phishing Threat

By raptur3

Archived: 2026-04-05 18:19:56 UTC

Phishing kits usually have distinct signatures in their delivery methods, infrastructure, and client-side code, which makes attribution fairly predictable. But recent samples began showing traits from two different kits at once, blurring those distinctions.

That's exactly what ANY.RUN analysts saw with Salty2FA and Tycoon2FA: a sudden drop in Salty activity, the appearance of Tycoon indicators inside Salty-linked chains, and eventually single payloads carrying code from both frameworks. This overlap marks a meaningful shift; one that weakens kit-specific rules, complicates attribution, and gives threat actors more room to slip past early detection.

Let's examine how this hybrid emerged, why it signals a shift in 2FA phishing, and what measures defenders should take in response.

Key Takeaways

- **Salty2FA activity collapsed abruptly in late October 2025**, dropping from hundreds of weekly uploads to ANY.RUN's Interactive Sandbox to just a few dozen.
- **New samples began showing overlapping indicators from both Salty2FA and Tycoon2FA**, including shared IOCs, TTPs, and detection rule triggers.
- **Code-level analysis confirmed hybrid payloads**: early stages matched Salty2FA, while later stages reproduced Tycoon2FA's execution chain almost line-for-line.
- **Salty2FA infrastructure showed signs of operational failure**, forcing samples to fall back to Tycoon-based hosting and payload delivery.
- The overlap aligns with earlier hypotheses suggesting a **possible connection to Storm-1747**, who are known operators of Tycoon2FA.
- **Attribution remains essential**: Distinguishing between these "2FA" phishing kits helps analysts maintain accurate hunting hypotheses and track operator behavior.
- **Defenders should update detection logic** to account for scenarios where Salty2FA and Tycoon2FA appear within the same campaign or even a single payload.
- **More cross-kit overlap is likely**, meaning future phishing campaigns may blend infrastructures, payloads, and TTPs across frameworks.

Part 1: Numbers Don't Lie – A Sudden Drop in Salty2FA Activity

It all started around the end of October 2025, when the number of the ANY.RUN sandbox submissions showing activity linked to Salty2FA dropped sharply compared to previous periods.

Weekly phishing reports (see the company's X posts) show that, despite the usual fluctuations in overall upload volume, the average number of Salty2FA-related analysis sessions consistently stayed in the range of several hundred per week.

However, once November began, the decline became dramatic: By November 11, 2025, Salty2FA had fallen to the bottom of the weekly threat rankings, with only **51 submissions**, compared to its typical **250+ per week**.

Fig.1: Salty2FA activity chart

Along with indicators of compromise (IOCs) and hunting rules, the ANY.RUN sandbox's **network block** previously triggered a near-constant alert tied to Salty-specific HTTP activity.

Fig.2: Last sandbox analyses showing detection of Salty2FA TTPs

This refers to the Suricata rule **sid:85002719**. If we filter *Public submissions* for analysis sessions where this rule fired, the most recent match dates back to **2025-11-01**:

Check recent analysis session

The first assumption was obvious: the detection logic became outdated, the framework received an update, and analysts simply hadn't refreshed the signatures in time. But what about **infrastructure indicators or domains**?

While IOCs sit lower on the Pyramid of Pain than Tools/TTP coverage, they are easy to track at scale and often remain in use long enough to provide meaningful visibility. They often remain active for some time, leaving repeated traces in the data. These recurring indicators make it easier for analysts to track the threat, update its context, and perform wider hunting to uncover new related domains, behaviors, and activity patterns.

The plan was simple: search for recent analysis sessions tagged with the threat name in ANY.RUN's Threat Intelligence Lookup, examine changes in the kit's behavior and client-side code, and then update the detection methods:

threatName:"salty2fa"

Fig.3: TI Lookup provides a complete overview of the latest Salty2FA attacks

But then things became even more unusual. In almost every analysis executed after **November 1**, the samples were either completely non-functional (examples 1, 2, 3) or behaved in ways that didn't align with Salty2FA at all.

For example, one analysis session showed the use of an **ASP.NET CDN**, which is not typical for this kit. It started to look as if someone had flipped a switch and taken a significant part of the framework's infrastructure offline.

A shutdown, maybe? Not exactly.

Alongside this decline, analysts also began seeing more sessions where the verdict included **both Salty2FA and Tycoon2FA;** two phishing kits that offer similar capabilities but differ in how they're built and operated.

And this didn't resemble a simple misattribution. The Tycoon2FA indicators were supported by long-validated detection logic, including rules that flag **DGA-generated domains** tied to the kit's **fast-flux infrastructure**.

Check analysis session with Salty2FA and Tycoon

Fig.4: Suricata detection showing Tycoon indicators inside a Salty2FA analysis session

This raised another hypothesis: a possible merging of infrastructure between the operators behind these PhaaS platforms. To verify it, we took another look at the JavaScript code used in the phishing pages.

The results turned out to be very interesting!

Part 2: When Two Kits Become One: A Deep Look at the Hybrid Payload

To understand what changed inside this new wave of submissions, we compared the code to earlier versions of both kits. For reference, the previous analyses are available here:

- Salty2FA
- Tycoon2FA

With these baselines in mind, let's take a closer look at the following analysis session:

Check analysis session

Fig.5: ANY.RUN's Sandbox exposes phishing attempts in seconds

The activity begins with the phishing page hosted on **Cloudflare Pages Dev;** a platform intended for front-end development and static site hosting, but one that threat actors frequently abuse due to how easy it is to deploy content there.

A closer look reveals several familiar artifacts: "motivational quotes" embedded in the markup and class names generated using a simple **"word + number"** pattern. These elements closely resemble the older (and certainly not harmless) **Salty2FA** codebase:

Fig.6: Salty2FA "Quotes"

Fig.7: Salty2FA class names

Scrolling a bit further down, we see the trampoline code responsible for retrieving and loading the next payload stage into the DOM; a sequence identical to the older Salty implementation.

But here's the interesting part: the code contains comments noting that the initial payload may fail to load, in which case the script should fetch the payload from an alternative URL. That fallback URL is written directly into the code with no obfuscation whatsoever.

Fig.8: Trampoline code in an older Salty sample

Fig.9: Trampoline code in the new Salty sample

After decoding the function argument, we get the address **hxxps[://]omvexe[.]shop//;** an IOC associated with Salty2FA. However, the payload will never be retrieved. When the script attempts to resolve the domain name, the DNS response is **SERVFAIL**, which differs from **NXDOMAIN** (non-existent domain).

SERVFAIL indicates an issue on the server side; for example, incorrect NS records or delegation problems where the resolver cannot determine which authoritative DNS server is responsible for the domain.

Fig.10: Salty2FA domain resolution errors

In other words, the Salty infrastructure is experiencing issues, and the script switches to a fallback plan, loading the page from the hardcoded secondary address.

After the initial failure, the script switches to a direct request to **hxxps[://]4inptv[.]1otyu7944x8[.]workers[.]dev/**, which delivers the next stage.

The first part of this stage contains obfuscated anti-analysis checks, implemented through Base64 decoding followed by an eval() call.

The second part is obfuscated using a **Base64-XOR** technique and contains the next portion of the payload:

Fig.11: Payload from the “alternative” execution path

After the code above runs, the page content is replaced, and new DOM elements are injected to mimic Microsoft’s official authentication page. The script also reinstates several common defense mechanisms; for example, blocking keyboard

shortcuts that open DevTools and performing execution-timing checks designed to detect debugging attempts using breakpoints.

Fig.12: Blocking DevTools keyboard shortcuts

What's more interesting is that traces of Salty2FA are still present here; in particular, the familiar "salted" source code comments:

Fig.13: Salty2FA traces inside the payload's source code

At the bottom of the page, there is a two-line script that once again executes Base64-decoded code via eval():

Fig.14: Another obfuscated code snippet

Finally, we hit the plot twist: the next stage loads code that mirrors the last steps of the **Tycoon2FA execution chain** almost line for line. The variable values, the order of functions, the way each component is implemented; all of it matches what earlier analyses and reports have already documented for this PhaaS platform.

Here are some of the clearest similarities between this sample and Tycoon:

Fig.15: Variable set with predefined values

Fig.16: Data-encryption function with hardcoded IV/key

Fig.17: Function for encoding stolen data as binary octets

Fig.18: Dynamic URL routing using RandExp patterns

Fig.19: POST request to a server using a characteristic DGA-generated domain name

It was also noted that some test data was not fully removed from the code.

Several sections appear to be entirely commented out, as if the phishing kit operator was making quick edits or testing new functionality but didn’t have time to finish refining it.

Fig.20: Test data inside the code

Fig.21: Fully commented-out function

Fig.22: Disabled IP logging inside one of the 2FA-handling routines

Taken together, this provides clear evidence that a single phishing campaign, and, more interestingly, a single sample, contains traces of both **Salty2FA** and **Tycoon**, with Tycoon serving as a fallback payload once the Salty

infrastructure stopped working for reasons that are still unclear.

So, what does the appearance of this kind of hybrid in the wild mean for PhaaS attribution, for the operators behind these frameworks, and for phishing threat hunting more broadly? Could this point to multiple groups working together within the same operation, especially given earlier assumptions that **Storm-1747** (the Tycoon operators) might also be connected to Salty2FA? Or does it suggest that the major PhaaS kits may ultimately be run by the same people?

Part 3: Are All These “Some2FA” Frameworks Really the Same?

Even though forensic work occasionally uncovers samples that include “a little bit of everything,” proper attribution between different phishing-kit families still matters. Being able to tell one kit from another ensures analysts don’t lose the unique traces that belong to a specific framework and don’t appear anywhere else. Those unique markers allow TI and Threat Hunting teams to build and test focused hypotheses, because trying to hunt under the umbrella of “all phishing attacks in the world” simply doesn’t work.

Clear attribution also helps teams collect and share fresh threat intelligence, write detection rules that map to the upper layers of the **Pyramid of Pain**, and keep those rules effective for as long as possible.

Attribution becomes even more valuable when you look at how it helps track shifts in the behavior and motivation of the groups running these kits. With **Salty2FA**, for example, there has already been speculation that **Storm-1747** may be responsible for maintaining, or even creating, the framework. If that’s true, then the known TTPs, victim profiles, and operational patterns associated with Tycoon2FA would also apply to attacks involving Salty2FA. That overlap can significantly shorten detection and response times.

It also leads to a practical expectation: if the activity of one kit suddenly drops off, defenders should be ready for a surge in another kit that’s likely controlled by the same operators. That means updating detection logic, running new threat-hunting sweeps, carrying out security audits and awareness training, and reviewing incident-response playbooks that reflect **Storm-1747’s** known TTPs.

How Should SOC Teams Respond to This Shift?

For SOC teams, the appearance of Salty2FA–Tycoon2FA hybrids calls for a shift in how these campaigns are detected, correlated, and investigated. When a phishing kit can fall back to a different framework mid-execution, defenders need to adapt their processes accordingly.

1. Treat Salty2FA and Tycoon2FA as part of one threat cluster: The overlap in infrastructure, indicators, and execution stages means detections tied to one kit may surface activity from the other. Correlation rules and enrichment pipelines should consider both families together.

2. Build hunting hypotheses that account for fallback payloads: If Salty infrastructure becomes unavailable, the same campaign may pivot into Tycoon2FA without leaving a clear break. Threat hunting should look for these transitions to avoid missing supporting evidence.

3. Rely more on behavior than static IOCs: Hybrid kits weaken simple signature-based workflows. DOM manipulation patterns, execution-stage logic, DGA activity, and fast-flux domains remain more stable than standalone indicators.

4. Refresh IR playbooks to reflect mixed execution chains: Playbooks should include scenarios where multiple frameworks appear in the same campaign, or where an incident involves a sequence of payloads from different kits.

5. Expect faster TTP propagation: If Storm-1747 is indeed behind both frameworks, changes observed in Tycoon2FA may quickly appear in Salty2FA as well. SOC teams should monitor these shifts to stay ahead of detection gaps.

In short, the rise of hybrid 2FA phishing kits means defenders should prepare for campaigns that operate more flexibly, more modularly, and with a higher tolerance for infrastructure failures; traits that align with increasingly mature threat groups.

Supporting Detection and Response with ANY.RUN

ANY.RUN provides SOC teams with the visibility and speed needed to keep up with hybrid phishing kits. With interactive analysis and real-time intelligence in one workflow, SOC analysts can validate attribution, tune detections, and respond with confidence:

- **Fast investigation of complex threats:** Analysts see initial malicious activity in about 60 seconds in 90% of cases, even for multi-stage phishing kits.
- **Immediate access to fresh IOCs:** ANY.RUN’s Threat Intelligence Feeds aggregate newly observed domains, URLs, IPs, and artifacts from 15,000 organizations and a community of more than 600,000 analysts worldwide, providing early visibility into indicators.

- **Deep inspection of mixed execution chains:** The interactive sandbox gives full visibility into each stage of the attack.
- **One-click enrichment with TI Lookup:** Analysts can instantly view historical use, related samples, and broader activity patterns around any indicator.
- **Reliable correlation signals:** Shared domains, DGA patterns, and reused client-side code become immediately visible across public and private submissions.

Together, these capabilities give SOC analysts a clearer, faster way to deal with hybrid phishing campaigns. They help teams spot changes early, run more focused hunts, and respond before attackers manage to regain traction.

Conclusion

In this analysis, we reviewed a case where payloads from **Salty2FA** and **Tycoon2FA** appeared together, following a sharp decline in Salty2FA activity. This kind of overlap may indicate operational issues on the Salty side, or, just as plausibly, suggest that both frameworks are operated by the same group, namely **Storm-1747**.

Going forward, we should expect to see more overlap in indicators of compromise, TTPs, and victim organizations across phishing campaigns involving these kits. For that reason, defenders should revisit their detection logic and develop hunting hypotheses that account for traces of both **Salty** and **Tycoon** appearing within the same context.

About ANY.RUN

ANY.RUN is a leading provider of interactive malware analysis and threat intelligence solutions used by security teams around the world. The service combines real-time sandboxing with a rich intelligence ecosystem that includes TI Feeds, TI Lookup, and public malware submissions.

More than 500,000 analysts and 15,000 organizations rely on ANY.RUN to speed up investigations, validate TTPs, collect fresh IOCs, and understand emerging threats through live, behavior-based analysis.

By giving defenders an interactive view of how malware behaves from the very first second of execution, ANY.RUN helps teams detect attacks faster, make informed decisions, and strengthen their overall security posture.

Experience how ANY.RUN’s solutions can power your SOC: start 14-day trial

Indicators of Compromise

- 1otyu7944x8[.]workers[.]dev
- xm65lwf0pr2e[.]workers[.]dev
- diogeneqc[.]pages[.]dev
- stoozucha[.]sa[.]com
- omvexe[.]shop
- lapointelegal-portail[.]pages[.]dev
- lathetai[.]sa[.]com

References

- https://app.any.run/tasks/46352ebf-7ee1-4d74-9850-2cdc6f6f0a49
- https://app.any.run/tasks/ccf7d689-7926-495d-b37f-d509536ff42b
- https://intelligence.any.run/analysis/lookup#{%22query%22:%22threatName:%5C%22salty2fa%5C%22%20AND%20threatName:%5C%22tycoon%5C%22%20AND%20domainName:%5C

raptur3

Network Analyst at ANY.RUN | + posts

Network Analyst at ANY.RUN. Keen to become a 'cybersec Swiss Army knife' man. Enjoys reading and writing deep-dive tech research.

Network Analyst at ANY.RUN. Keen to become a 'cybersec Swiss Army knife' man. Enjoys reading and writing deep-dive tech research.
