0-DAY NO CVE HIGH PUBLIC

ProFTPD RewriteMap utf8trans BSS overflow

ProFTPD 1.3.9d RewriteMap int:utf8trans copies a client FTP argument into two 1024-slot BSS buffers with no dest cap. ASan global-buffer-overflow WRITE of size 8. Session child aborts. Class Crash. Labbed. Not RCE.

NAME ProFTPD RewriteMap utf8trans BSS overflow	TYPE 0-day analysis
CVE n/a	CVE RISK high
DISCLOSURE STATUS public	VENDOR ProFTPD Project
AFFECTED ProFTPD 1.3.9d (f59f248) contrib mod_rewrite when RewriteEngine is on and a RewriteRule feeds int:utf8trans. Issue 2173 unescape is a different bug already patched on this pin.	PUBLISHED 12 Oct 2026
UPDATED 12 Oct 2026	TAGS 0-day, proftpd, mod-rewrite, crash, cwe-120

TL;DR

Lab files and proof-of-concept here.

01 1024 slots, 512 KiB of argument

I am @abraxas_null. The proof of concept is on GitHub: **abraxas/proftpd-rewrite-utf8trans** (loopback client). The lab stack is `lab/`: Dockerfile, docker-compose.yml, run.sh, poc.py. Authorized lab only. It talks to loopback.

This is **ProFTPD 1.3.9d** (tag `v1.3.9d`, commit `f59f248e26bdfa514443e42bf72a145641df635d`). `include/version.h` says `PROFTPD_VERSION_TEXT "1.3.9d"` and `PROFTPD_VERSION_NUMBER 0x0001030908`. No CVE yet. CWE-120. Class Crash. Not RCE.

If a `RewriteRule` feeds a client FTP argument into `RewriteMap int:utf8trans` before login, I score that 7.5 High (`CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H`). The lab did not. The lab used `STOR` after login, `RewriteCondition %m STOR`, so that run is PR:L. Same write. Same child abort. Different who can send the name.

`rewrite_map_int_utf8trans` copies that argument into two function-scope BSS arrays of `PR_TUNABLE_BUFFER_SIZE` (1024) slots. `rewrite_utf8_to_ucs4` writes one `unsigned long` per ASCII byte with no destination cap. `CommandBufferSize` can be 512 KiB. ASan on the pin: `global-buffer-overflow` `WRITE of size 8`, then `ABORTING`. The session child died. A fresh login still worked. The daemon stayed up.

That is the leftover. Not a new map type. Not Apache `prg:`. A builtin that believed a 1024-slot static was large enough for an FTP argument.

02 Why ProFTPD

I had been reading daemons that already spent 2026 closing the High rows everyone already had a CVE number for. FTP is the other listener people still put on a public address because a scanner once needed a drop box. ProFTPD is the one that still looks like a small Unix inetd cousin and then turns out to be a prefork process with a

contrib tree the size of a second product: SFTP, TLS, SQL, copy, rewrite, SITE extras, wrap2, SNMP.

The security model is honest if you read it. Prefork. Sample `basic.conf` leaves `DefaultRoot` commented out. `RootRevoke` defaults on, so the worker is the login UID, not 0. Core static modules are `mod_core`, `mod_xfer`, `mod_auth`, `mod_ls`, `mod_log`, `mod_site`, `mod_delay`, `mod_facts`. Contrib is off until you load it. Debian-style shared builds commonly ship those DSOs anyway.

I only wanted Critical or High. Remote and Local. Medium stays on the floor unless it chains. Crash without a write primitive is usually skip. A write primitive that ASan classifies as `WRITE of size 8` into BSS, on a client-controlled FTP argument, is the exception. Availability of the session child is the proven row. Nearby BSS integrity is inferred. RIP is closed. I am not going to inflate Crash to RCE because the destination is a global.

Sixteen hunts on the pin: FTP parser, path canonicalization, SITE, MLSD/facts, chroot/privs, Controls, mod_sftp, mod_tls, mod_sql, rewrite / `.ftpaccess` / exec, auth, data/REST, wrap2/ldap/radius/snmp, digest/json/display, ifsession/DSO. Default core unpublished Crit/High came back empty. The leftover in this post is contrib `mod_rewrite` after the 2173 unescape patch.

03 Pinning 1.3.9d, not the rc

1.3.9d shipped 14 August 2026. That is the latest 1.3.9 tag I hunted. `1.3.10rc4` is the same calendar day and a different commit. `1.3.10rc5` is unreleased master NEWS. Pinning an rc is how you "find" a bug the next tarball already closed, or how you miss a leftover that only the stable tag still ships.

The clone is the GitHub tree at tag `v1.3.9d`. HEAD of that tag is `f59f248e26bdfa514443e42bf72a145641df635d`. The lab copies that tree, then refuses to compile unless `.pin-commit` is that SHA, `.pin-tag` is `v1.3.9d`, and

`include/version.h` still says `1.3.9d` / `0x0001030908`. Mismatch is FAIL. You do not crash a different binary and call it this leftover.

04 The skip list is half the hunt

Before you look for leftovers, you write down what is already public so you do not rediscover a NEWS item with a press release. ProFTPD's 2026 crop on this pin is already closed or skip-public:

- CVE-2026-42167 `mod_sql` `is_escaped_text` SQLi. Fixed 1.3.9a/b.
- CVE-2026-44331 `mod_wrap2_sql` rDNS SQLi. Fixed 1.3.9b.
- CVE-2026-35025 authenticated `RNFR /proc/self/root` ACL skip via unresolved `dir_canonical_path`. Public. Upstream declined a core `/proc` special case. `mod_procfs` is 1.3.10rc, not this tag. Skip-public, including the DELE sibling of the same class.
- CVE-2026-53994 / CVE-2026-63090 authenticated SFTP heap overflow in packet reassembly. Fixed 1.3.9c.
- CVE-2026-63091 authenticated SCP signed integer overflow. Fixed 1.3.9c.

NEWS 1.3.9a-d already ate the parser leftovers (2019/2020/2030), Controls 2210, TLS JSON 2166, SSH ETM/payload rows, STAT `-C` UAF, empty-password auth, and Issue 2282 `.ftpaccess` ancestor Limit skip. Public issues still sitting in this tarball but disclosed (2256 RNTO EXDEV, 2286 REST/RANG, 2319 stale `authenticated_without_pass`, 2330 `mod_radius` PASS stack, 2175 SITE RMDIR filter, 2095 SITE COPY source READ, 831 UserAlias) stay skip-public even when the bytes are still here.

The rewrite row on that list is Issue 2173. `RewriteMap unescape` one-byte OOB write, plus a guard that looked at the encoded key instead of the decoded byte. NEWS

1.3.9c. Patched on this pin. `rewrite_map_int_unescape` now `pcalloc`s `strlen(key) + 1` and rejects decoded `'/'` and `NUL`. That is a different builtin. That is not `utf8trans`.

05 Default core is a handshake, then a wall

Unauthenticated FTP, then `USER` / `PASS`, then the command loop. `pr_cmd_read` caps the line at `CommandBufferSize`. Default when the directive is absent is `PR_DEFAULT_CMD_BUFSZ`, which is `PR_TUNABLE_PATH_MAX + 7`. On Linux `MAXPATHLEN` is 4096, so the default command line is already about 4 KiB. The *maximum* you may set is `PR_TUNABLE_CMD_BUFFER_SIZE` `(512 * 1024)`, Bug 4014. Either number is larger than 1024 slots.

Parser leftovers from 2019/2020/1683/2030 are closed on this pin. Controls default deny-all after 2210. `RootRevoke` on. REST/RANG 2286 is public. The default daemon unpublished Crit/High hunt is NONE. I am not going to dress up a public `/proc` RNFR as a new core RCE, and I am not going to pretend sample `basic.conf` is this bug.

Contrib is the other half of the product. `mod_rewrite` is Apache-shaped FTP command rewriting. `RewriteEngine` is off until you turn it on. `rewrite_engine` starts `FALSE`. `rewrite_fixup` is `PRE_CMD` `C_ANY`: every FTP command with an argument, before the handler, if the engine is on. A `RewriteCondition` can limit that to `STOR`. It can also not.

06 Then I opened utf8trans

The howto is not shy about the pattern. Uppercase every upload name:

PLAIN TEXT Copy

```
<IfModule mod_rewrite.c>
  RewriteEngine on
```

```
  RewriteMap uppercase int:toupper
  RewriteCondition %m STOR
  RewriteRule ^(.*) ${uppercase:$1}
</IfModule>
```

The documented map lookup is `${map-name:lookup-key}`. The documented catch-all is `RewriteRule (.*) ${map:$1}`. Swap `toupper` for `utf8trans` and you have the RFC 2640 internationalization builtin: "Translates UTF-8 encodings in the lookup up key into Latin-1 characters." Same `int:` slot. Same client argument. The module doc lists `idnatrans`, `replaceall`, `toupper`, `tolower`, `unescape`, `utf8trans`.

`rewrite_subst_maps` finds `${u8:$1}`, looks up the `RewriteMap` named `u8`, and calls `rewrite_subst_maps_int`, which is a function pointer into the builtin. For `utf8trans` that pointer is `rewrite_map_int_utf8trans`. The key is the FTP argument. On the lab config, that is the `STOR` filename.

`tolower` / `toupper` walk the key in place. `unescape` allocates from the command pool, and after 2173 that allocation is `strlen(key) + 1`. `idnatrans` talks to libidn and heap. `utf8trans` is the one that still keeps two file-lifetime statics:

C Copy

```c
static char *rewrite_map_int_utf8trans(pool *map_pool, char *key) {
  int ucs4strlen = 0;
  static unsigned char utf8_val[PR_TUNABLE_BUFFER_SIZE] = {'\0'};
  static unsigned long ucs4_longs[PR_TUNABLE_BUFFER_SIZE] = {0L};

  if (key == NULL ||
      strlen(key) == 0) {
    return NULL;
  }

  memset(utf8_val, '\0', sizeof(utf8_val));
  memset(ucs4_longs, 0, sizeof(ucs4_longs));

  ucs4strlen = rewrite_utf8_to_ucs4(ucs4_longs, strlen(key),
    (unsigned char *) key);
```

`PR_TUNABLE_BUFFER_SIZE` is 1024. That is the "miscellaneous tasks" buffer in `options.h`, the same constant used for random copies elsewhere in the daemon. It is not `CommandBufferSize`. It is not `PR_TUNABLE_CMD_BUFFER_SIZE`. The `memset`s are sized to the destination. The conversion is not.

07 One unsigned long per ASCII byte, no cap

The comment on `rewrite_utf8_to_ucs4` says most of the code is RFC 2640 and that UCS-4 here "just happens to coincide with ISO-8859-1." The signature is:

```c
static int rewrite_utf8_to_ucs4(unsigned long *ucs4_buf,
    size_t utf8_len, unsigned char *utf8_buf)
```

`utf8_len` is `strlen(key)`. There is no `ucs4_buflen`. The ASCII fast path is the lab path:

```c
while (utf8_buf != utf8_endbuf) {
  pr_signals_handle();

  /* ASCII chars - no conversion needed */
  if ((*utf8_buf & 0x80) == 0x00) {
    *ucs4_buf++ = (unsigned long) *utf8_buf;
    utf8_buf++;
    ucs_len++;
```

One `unsigned long` per input byte. On the ASan aarch64 lab that is a `WRITE of size 8`. 200 ASCII bytes stay inside 1024 slots. 2048 ASCII bytes do not. The 2- through 6-byte UTF-8 arms do the same uncapped `*ucs4_buf++`. A multi-byte lead on the last byte is a 1-byte OOB read of the source; that is DoS-shaped and not this High.

On success, if `ucs4strlen > 1`, the function copies again:

```c
for (i = 0; i < ucs4strlen; i++) {
  utf8_val[i] = (unsigned char) ucs4_longs[i];
}

return pstrdup(map_pool, (const char *) utf8_val);
```

`utf8_val` is 1024 `unsigned char`s. Same missing cap. ASan never got there. It aborted on the first `unsigned long` past `ucs4_longs`. Empty keys and length-1 keys return `NULL` before that loop. The control name is 200 bytes on purpose: long enough to exercise the map, short enough to stay in bounds.

The second argument to `rewrite_utf8_to_ucs4` is named as if it were a destination length. It is the source length. That is the whole bug in one parameter.

08 Closed doors next to the leftover

Issue 2173 is the sibling people will try to re-file. Unescape was `pcalloc(map_pool, strlen(key))` plus a NUL at `value[i]`, and a slash/NUL check on the wrong index. 1.3.9c allocated one more byte and checked the decoded value. `utf8trans` was not in that diff. Master still has the 1024-slot statics and the uncapped walker. Same class (RewriteMap builtin OOB write on a client-controlled map key), larger, still here.

There is no `prg:` map type. `set_rewritemap` accepts `int:`, `fifo:`, `txt:`. Apache `prg:/path` does not exist. FIFO (`rewrite_subst_maps_fifo`) writes `lookup_key + "\n"` to an already-open named pipe and reads one line back, cap `REWRITE_FIFO_MAXLEN` 256. It does not `fork`, `exec`, or `popen`. The other end is an admin-started program. A shell that `eval`s the key is that program's bug.

`RewriteMap` is `CHECK_CONF(..., CONF_ROOT|CONF_VIRTUAL|CONF_GLOBAL)`. It is not `CONF_DYNDIR`. A user cannot plant `RewriteEngine` or `RewriteMap` in `.ftpaccess`.

`AllowOverride` cannot appear in `.ftpaccess` either. `ExecOnCommand` is `execve` argv, not `system`, and the program path is a config token, not a substituted filename. Those hunts closed. They are not this leftover.

`mod_exec` Issue 2358 is public DoS on 1.3.10rc5. Skip.

09 The lab

Host is macOS. One container. Compose project `proftpd-rewrite-utf8trans`. Loopback only.

The image is debian bookworm-slim, ProFTPD 1.3.9d built from the pinned `src/` tree with contrib `mod_rewrite` statically linked (`--with-modules=mod_rewrite`), PAM/cap/nls/ident off. ASan on the daemon:

DOCKERFILE Copy

```
ENV CFLAGS="-g -O1 -fsanitize=address -fno-omit-frame-pointer" \
    LDFLAGS="-fsanitize=address" \
    ASAN_OPTIONS="detect_leaks=0:abort_on_error=0"
```

Runtime `ASAN_OPTIONS` is `detect_leaks=0:abort_on_error=1:halt_on_error=1:log_path=/tmp/asan`. `libasan8` is in the final image. The entrypoint `LD_PRELOAD`s it. Pin checks run in the build stage against `.pin-commit`, `.pin-tag`, `include/version.h`, and the two function names in `contrib/mod_rewrite.c`.

Published `127.0.0.1:18740` is FTP 21. Passive `127.0.0.1:18741-18750`. Auth is `mod_auth_file`, user `labuser`. `RewriteEngine on`. The rewrite block is the howto STOR recipe with `utf8trans`:

PLAIN TEXT Copy

```
<IfModule mod_rewrite.c>
  RewriteEngine        on
```

```
  RewriteLog            /var/log/proftpd/rewrite.log
  RewriteMap            u8 int:utf8trans
  RewriteCondition      %m STOR
  RewriteRule           ^(.*) ${u8:$1}
</IfModule>
```

The PoC logs in, then `STOR`s. Control name is 200 ASCII bytes. Trigger name is 2048 ASCII bytes. SUCCESS requires the short `STOR` to live, the long `STOR` to kill the session child with an ASan write, and a fresh login after that to still work. A trigger that succeeds is FAIL. A trigger that dies without ASan and also takes the daemon with it is FAIL. The oracle is the sanitizer, not a hung accept loop.

Control: `STOR` of a 200-byte name, `226`, then `NOOP command successful`.

Trigger: `STOR` of 2048 `A`s. The control connection reset. `EOFError`. ASan:

PLAIN TEXT Copy

```
==26==ERROR: AddressSanitizer: global-buffer-overflow on address 0xaaaad4755780 a
WRITE of size 8 at 0xaaaad4755780 thread T0
SUMMARY: AddressSanitizer: global-buffer-overflow (/usr/local/sbin/proftpd+0x1b7c
==26==ABORTING
```

`WRITE of size 8` is one `unsigned long` past `ucs4_longs`. Prefork: that is the session child. Post-trigger login returned `pwd='/'`. `control-alive=1`. The parent was still accepting on 18740.

Last run:

PLAIN TEXT Copy

```
label=PROFTPD-REWRITE-UTF8TRANS ftp=127.0.0.1:18740 user=labuser
control_len=200 trigger_len=2048
banner pin=1.3.9d
== control STOR short ASCII ==
control stor=ok name_len=200 noop=200 NOOP command successful
== trigger STOR utf8trans map key ==
trigger conn=EOFError() reset=1 name_len=2048
```

```
IOC ==26==ERROR: AddressSanitizer: global-buffer-overflow on address 0xaaaad4755
IOC WRITE of size 8 at 0xaaaad4755780 thread T0
IOC SUMMARY: AddressSanitizer: global-buffer-overflow (/usr/local/sbin/proftpd+0
IOC ==26==ABORTING
trigger ok=0 reset=1 timeout=0 asan=global-buffer-overflow write=1 child-dead=1
post-trigger login pwd='/'
control-alive=1
SUCCESS PROFTPD-REWRITE-UTF8TRANS asan=global-buffer-overflow write=yes child-de
```

Replay:

PLAIN TEXT Copy

```
git clone https://github.com/abraxas/proftpd-rewrite-utf8trans
cd proftpd-rewrite-utf8trans/lab
./run.sh
```

Bind it to loopback. The loopback client is `lab/run.sh` with a banner on it. Crash witness. No RIP payload. No heap spray. The oracle is ASan aborting the child.

10 Wrong turns, in the order I made them

Pinning 1.3.10rc. Same week as 1.3.9d. Different tree. The leftover class is "this GA still ships it."

Treating this as Issue 2173. Unescape is patched on this pin. The allocation is `strlen(key) + 1`. The slash/NUL check is on the decoded byte. `utf8trans` is the builtin that still has a 1024-slot static. NEWS listed unescape. NEWS did not list this.

Assuming I needed `CommandBufferSize 512K`. The tunable *max* is 512 KiB. Default on Linux is `PATH_MAX + 7`, about 4 KiB. 2048 already overflows 1024 slots. 200 does not. The lab never raised the directive.

Treating 1025 bytes as the crash. One extra `unsigned long` is the first OOB write. ASan will fire there. The lab used 2048 so the overflow is not a rounding argument

about `STOR ` plus CRLF plus the name. The SUCCESS token is `asan=global-buffer-overflow write=yes`, not a particular length past 1024.

Calling this RCE. Function-scope statics live in BSS. Nearby rewrite state (`rewrite_engine`, `rewrite_logfd`, `rewrite_vars`, pools, `rewrite_parse_map_str`'s `substr`) is in the same segment in some layouts and not in others. That integrity row is inferred and compiler-dependent. The lab did not move RIP. The lab did not exec. Class Crash. Inflating it is how you get ignored.

A `.ftpaccess` plant. `RewriteMap` is not allowed there. If the admin did not put `int:utf8trans` in `proftpd.conf`, a login UID cannot add it.

`prg:` as a shell. There is no `prg:`. FIFO is line I/O to an admin process that already exists. Do not turn a missing Apache feature into a ProFTPD RCE.

Compiling without ASan. A non-ASan 2048-byte `STOR` can still kill the child. The SUCCESS line I will stand behind is the sanitizer write. The Dockerfile is ASan on purpose.

Treating a dead accept loop as SUCCESS. Prefork. One client process. The parent must still answer a fresh `USER`/`PASS`. `control-alive=0` with ASan would still be a write, and it would also be a different bug. This leftover is the child.

A reverse shell. Theatre. The oracle is `WRITE of size 8` then `ABORTING`. No payload in the pack.

Bundling SITE COPY or SFTP `copy-file`. Other contrib leftovers on this pin are other compose projects, other loopback ports, other posts. This pack is `int:utf8trans`.

Scoring a prefork child crash as below High because the daemon stayed up. The write is real. The argument is client-controlled. The map is a documented recipe. Availability of that session is A:H for the process that held it. A:H on the whole host would have been a different lab.

11 What an attacker can do

Need contrib `mod_rewrite` loaded, `RewriteEngine on`, a `RewriteMap ... int:utf8trans`, and a `RewriteRule` whose lookup key is client text. The howto pattern is enough. Pre-auth if the rule is not limited to a post-login command (`USER` is an argument). The lab limited it to `STOR`, so the proven path is an authenticated upload name.

METHOD	SINK
FTP command whose arg is the utf8trans map key, length > 1024	`rewrite_utf8_to_ucs4` then `utf8_val[i] = ucs4_longs[i]`

METHOD	SINK
RIP / code execution	none claimed
`.ftpaccess` **plant of RewriteMap**	`CONF_DYNDIR`
`prg:` **shell**	no such map type

You do not get a shell. You do not get the daemon. You do not get a file write. You get the session process aborting because a 1024-slot `unsigned long` array was asked to hold a command line the rest of ProFTPD will happily accept.

Reach is Remote. Contrib, not default-compiled. Typical distro SFTP/FTPS vhosts that also turned rewrite on for case folding or UTF-8 names are the victim shape. Unix DAC still applies to whatever that UID could already touch. Default core FTP without this map is not this bug.

12 The fix

Cap the destination. `rewrite_utf8_to_ucs4` needs the capacity of `ucs4_longs`, not only `strlen(key)`. Stop writing past `PR_TUNABLE_BUFFER_SIZE` slots. Return an error (or a truncated conversion the caller already treats as "not a proper UTF-8 string") instead of walking off the array. The `utf8_val[i]` loop needs the same bound. A 1024-slot BSS buffer is not a sink for `CommandBufferSize`.

Until a release with that cap, do not point `int:utf8trans` at a client-controlled argument. `toupper` / `tolower` / post-2173 `unescape` do not have this static. Turning `RewriteEngine` off is the config-level close. Raising `CommandBufferSize` makes it worse.

I am not going to print a 2048-byte `STOR` you can paste at someone else's listener. The missing dest length is the useful part. If you own the box, run the proof of concept against loopback.

13 References

- Proof of concept: abraxas/proftpd-rewrite-utf8trans · proftpd-rewrite-utf8trans-Abraxas-Labs.py
- Lab: `lab/` · Dockerfile · docker-compose.yml · run.sh · poc.py
- @abraxas_null · github.com/abraxas · abraxaslabs.tech · abraxas.null@proton.me
- CWE-120
- CVSS **High 7.5** if the rule is pre-auth `CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H` · lab path is PR:L (`STOR` after login)
- Tree: proftpd/proftpd v1.3.9d (`f59f248e26bdfa514443e42bf72a145641df635d`) · `contrib/mod_rewrite.c` `rewrite_map_int_utf8trans`, `rewrite_utf8_to_ucs4`, `rewrite_subst_maps_int`, `rewrite_fixup` · `include/options.h` `PR_TUNABLE_BUFFER_SIZE` 1024, `PR_TUNABLE_CMD_BUFFER_SIZE` 512 KiB · `include/version.h`
- Docs: mod_rewrite.html RewriteMap · howto/Rewrite.html · RFC 2640

- Sibling, patched on this pin: Issue 2173 unescape OOB
- Last lab line: `SUCCESS PROFTPD-REWRITE-UTF8TRANS asan=global-buffer-overflow write=yes child-dead=yes PROFTPD-REWRITE-UTF8TRANS-WITNESS`

Abraxas Labs

Independent security research by abraxas.

Analysis of 0-days, n-days, malware, and the

leftovers that still matter.

GitHub **@abraxas_null** **abraxas.null@proton.me** **abraxaslabs.tech**

Independent research. Private analysis by arrangement. No ads. abraxas.null@proton.me · For defensive awareness and authorized testing only.

© 2026 Abraxas Labs