<Cadey> Hello! Thank you for visiting my website. You seem to be using an ad-blocker. I understand why you do this, but I'd really appreciate if it you would turn it off for my website. These ads help pay for running the website and are done by Ethical Ads. I do not receive detailed analytics on the ads and from what I understand neither does Ethical Ads. If you don't want to disable your ad blocker, please consider donating on Patreon or sending some extra cash to **xeiaso.eth** or **0xeA223Ca8968Ca59e0Bc79Ba331c2F6f636A3fB82**. It helps fund the website's hosting bills and pay for the expensive technical editor that I use for my longer articles. Thanks and be well!

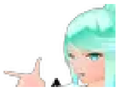
🥺: the best sudo replacement

Read time in minutes: 14

Image generated by Waifu Diffusion -- 1girl, fox ears, blue hair, blue eyes, katana, bamboo forest, kimono, long hair, princess, pokemon, fluffy hair, shouting, coffee, chibi, portrait, dialogue, monado, running, fox tail, blue tail

<Mara> I wonder how many people's RSS/JSONFeed readers we broke with the title...

<Aoi> Come on, it couldn't have been *that* many, things support Unicode now, right?



<Numa> >implying things support Unicode properly in the year of our lord two thousand and twenty-three

<Aoi> They do support Unicode though...right? They have to.

<Cadey> We'll find out.

Security is impossible. We just like to pretend otherwise so that we can constantly project this aura of impenetrability that will save us from having to admit the reality that it's impossible. One of the biggest targets in the modern information security world is sudo. It is a command that lets you *s*et *u*ser and then *do* a command. Sudo is one of the most widely deployed programs on the Internet and is widely regarded as critical infrastructure.

<Aoi> Sooo the creators and maintainers of sudo take things very seriously by using something like Rust, maintain a high quality standard of malicious inputs by fuzzing all public attack surfaces, and try to minimize the amount of code involved in order to prevent vulnerabilities from being a problem?

A prior version of this conversation snippet was badly phrased. You are reading an edited version in case this is relevant in internet comment arguments.

<Cadey> I don't know about the code quality standards of the sudo project, but overall I don't see them doing any concerted effort to try to migrate away from C (or to reduce the complexity of sudo) and there are frequent security vulnerabilities that result in attackers getting root access anyways. I really wish the industry as a whole would take languages like Rust a bit more seriously and start actually moving towards programs being safer to use because security vulnerabilities in core infrastructure result in emergency patches. It was disappointing to see an attempt at using Rust in an important Python library torpedoed by users of obscure architectures not supporting Rust. Maybe the solution there is to use WebAssembly as a compile target instead of making everything be native code. I wouldn't wish hppa's reverse stack growth on anyone trying to write a compiler though.

<Aoi> Oh god...

I'm tired of this situation and I bet a lot of the ecosystem is too. There's been talk and ideas, but not enough in the action department. I made a new tool. A better tool. One that will let all of us proceed towards the future we deserve. I made a sudo replacement named 🥺.

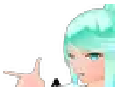
🥺

🥺 has no pronounceable name in English or any other speakable human language. It is named 🥺, but it is referred to as `xn--ts9h` (the punycode form of 🥺) in situations where emoji are not yet supported (such as Debian package names).

To use 🥺, install it (such as from the Debian package) and then run it in place of sudo:

```
$ id
uid=1000(xe) gid=1000(xe) groups=1000(xe),102(docker)

$ 🥺 id
uid=0(root) gid=0(root) groups=0(root),102(docker),1000(xe)
```

<Mara> Wait, what? That's it? How is this even secure at all? If it doesn't ask you for your password how can you be sure that an actual human is making the request and not some malicious script?



<Numa> Using this program requires you to be able to type an emoji. Most attack code is of such poor quality that they are unable to run commands named with emoji. This makes the program secure.

<Aoi> This is not how any of this works.

Here it is broken down statement by statement.

First, I pull in a bunch of imports from the standard library and also the syslog to write a message to syslog about what's going on:

```
use std::{env, os::unix::process::CommandExt, process::Command};
use syslog::{unix, Facility::LOG_AUTH, Formatter3164};
```

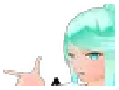
Next, I create a main function that returns an `io::Result`, this is an error that is returned by most of the standard library functions that do I/O operations with the OS.

```
fn main() -> io::Result<()> {
```

The correct usage of this program is to run it like 🥺 id, so if the user doesn't specify a program to run, then it should blow up with an error message instead of panicking:

```
if env::args().len() == 1 {
    eprintln!("usage: {} <command> [args]", env::args().nth(0).unwrap());
    return Ok(());
}
```

<Aoi> Wait, what? Why is it returning that everything is okay if the user is doing it wrong? Shouldn't it return some kind of error code that the running program or shell can catch?



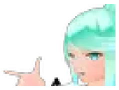
<Numa> It's a feature.

<Aoi> I really hope I never have to maintain any of your code.

Next, we grab the program name and arguments from the command line arguments of 🥺 and send a message to syslog that it's being run so that there is *some* accountability after-the-fact:

```
let program = env::args().nth(1).unwrap();
let args = env::args().skip(2).collect::<Vec<String>>();
let mut writer = unix(Formatter3164 {
    facility: LOG_AUTH,
    hostname: None,
    process: "🥺".into(),
    pid: 0,
})
.unwrap();
writer
    .err(format!("running {:?} {:?}", program, args))
    .unwrap();
```

<Aoi> Wait so the emoji works there, but it probably isn't going to work in people's RSS feed readers? How does that make any sense?



<Numa> It doesn't, lololol

<Cadey> UNIX is mostly devoid of the concept of character sets. Any character is fine as long as it doesn't have a null terminator (this ends the string in C). I'd be more amazed if the emoji use broke something, as there are legitimate uses for putting non-Latin characters into message buses like that. Also most RSS feed readers have very poor code quality.

Finally, the actual command is executed:

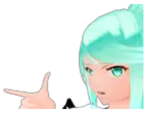
```
Err(Command::new(program).args(args).uid(0).gid(0).exec().into())
```

This works because I'm using the `CommandExt` trait implementation of `Command` that adds some methods we need:

- `uid(&mut self, id: u32)` to set the user ID of the child process (`setuid(2)` in C)
- `gid(&mut self, id: u32)` to set the group ID of the child process (`setgid(2)`, though groups are starting to die out due to them not being across multiple machines without extra effort like configuration managment)
- `exec(&mut self)` which runs the `execvp(3)` system call that *replaces* the current 🥺 with the child process

The key part is the `exec` call at the end. One of the interesting things about the `exec`-family of system calls in UNIX is that it *replaces* the current process if it succeeds. This means that the function will never return unless some error happened, so the `exec` method *always* returns an error. This will make error handling happen properly and if things fail the process will exit with a non-zero error code:

```
$ cargo run --release ls
    Finished release [optimized] target(s) in 0.06s
     Running `target/release/🥺 ls`
Error: Os { code: 1, kind: PermissionDenied, message: "Operation not permitted" }
```



<Numa> Sure, this error message could be better, but that's a 2.0 feature. This is a disruptive program poised to totally reshape the security industry so we have to *move fast and break things*!

I'm fairly sure that this program has no bugs that aren't either a part of the syslog crate or the Rust standard library.

Installation

You can install 🥺 by downloading the **.deb** file from my fileserver and installing it with **dpkg -i**. This will give you the 🥺 command that you can use in place of **sudo**.

<**Numa**> This will let you stick it to the man and let you self-host your own sudo on a $5 a month VPS from a budget host. You can't have any vulnerabilities if there are no bugs to begin with!

This is also known to work on Amazon Linux 2, so you can create blursed things like this:

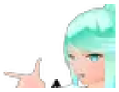
```
$ ssh -A xe@10.77.131.103
Warning: Permanently added '10.77.131.103' (ED25519) to the list of known hosts.
Last login: Fri Jan 20 04:09:11 2023 from 10.77.131.1

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[xe@inez-rengenne ~]$ 🥺 id
```

```
uid=0(root) gid=0(root) groups=0(root),10(wheel),1000(xe)
```

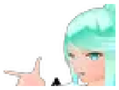
<**Mara**> Pro tip! You can apparently pass a URL to a `.rpm` file to `yum install` and it will just download and install that `.rpm` file. This is incredibly cursed.

The `.deb` package was built on Ubuntu 18.04 and the `.rpm` package was built on Amazon Linux 2, so it should be compatible with enough distributions that you don't have to care.

<**Mara**> There's even a manpage you can read with `man 8 🥺`!



<**Numa**> And most importantly, patches welcome!



<**Numa**> By the way, there are many more lovely ways to get root than just by asking nicely with `setuid`. Why doesn't this program use those?

<**Cadey**> We gotta save *something* for part 2, otherwise that would spoil all the *fun*.

<**Aoi**> I don't know if I like what you mean by "fun" there...

▶ Share on Mastodon

This article was posted on M01 20 2023. Facts and circumstances may have changed since publication Please contact me before jumping to conclusions if something seems wrong or unclear.

Tags: **infosec sudo rust**

This post was not WebMentioned yet. You could be the first!

The art for Mara was drawn by Selicre.

The art for Cadey was drawn by ArtZora Studios.

Some of the art for Aoi was drawn by @Sandra_Thomas01.

Like what you see? Donate on Patreon like these awesome people!

Looking for someone for your team? Take a look here.

See my salary transparency data here.

Served by /nix/store/d9g49fmfpwry1ls1fxr0cjmwmfz525iw-xesite-3.0.0/bin/xesite, see source code here.