# Himitsu

## A novel secret storage system

Drew DeVault
SourceHut

May 28, 2022

# What is Himitsu?

Himitsu is a simple and highly extensible system for safely storing and making use of secret data like passwords and private keys.

- Website logins
- SSH keys
- PGP keys
- TOTP keys
- And anything else...

# The basics of Himitsu

Himitsu is essentially a key/value store, where each "key" stores one secret (and metadata relevant to it) as a set of key/value pairs.

```
proto=web host=meta.sr.ht username=sircmpwn password!

proto=web host=codeberg.org username=sircmpwn password!

proto=ssh type=ssh-ed25519 comment=sircmpwn@homura
        skey! pkey=pF7SljE25sVLdWvIn04gfqpJbbjxI6j+tIUcNWzVTHU=
```

# Himitsu key format

Himitsu does not care what the key/value pairs are – it just stores them. In practice, the "proto" key defines, by convention, the meaning of the other key/value pairs.

- **proto=web** keys are form fields, values are their value
- **proto=ssh** pkey= public key, skey!= private key (base64)
- **proto=totp** secret!= HMAC key
- **proto=imap** host, port, username, password, ssl mode, etc
- etc...

# Setting up a key store

Pretty straightforward:

```
$ himitsu-init
Initializing a new himitsu secstore.
Please enter a passphrase:
Please enter the same passphrase again:
Successfully initialized new secstore.
```

# Communication with the key store

It's just a Unix socket.

```
$ nc -U $XDG_RUNTIME_DIR/himitsu
=> query proto=web host=meta.sr.ht
<= key proto=web host=meta.sr.ht username=sircmpwn password!
<= end
=> query proto=web host=meta.sr.ht username password!
<= key proto=web host=meta.sr.ht username=sircmpwn password!
<= end
```

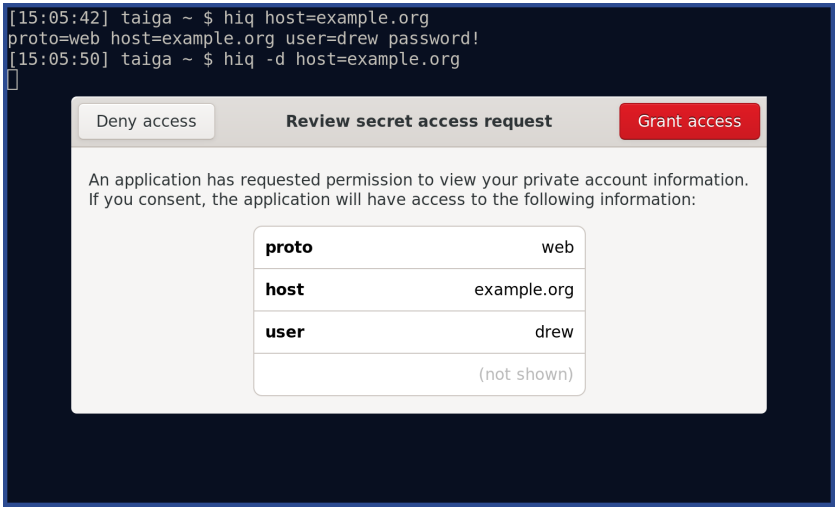
# Communication with the key store via hiq

But you don't generally use netcat:

```
$ hiq proto=web host=meta.sr.ht
proto=web host=meta.sr.ht username=sircmpwn password!
$ hiq proto=web host=meta.sr.ht username? password!
proto=web host=meta.sr.ht username=sircmpwn password!
# Request decryption of secret keys:
$ hiq -d host=example.org
proto=web host=example.org user=drew password!=hunter2
```

hiq supports querying, adding, deleting, and decrypting keys.

# Key decryption requires user consent



# Himitsu extensibility

You can customize virtually everything outside of the core key/value store features:

- Custom consent prompters (GTK, Qt, TTY, etc)
- Top-down integrations for new protocols (SSH, PGP, TOTP, etc)
- Custom key store initialization protocols*
- Authentication agents which do not disclose your password to third-party software*

* Aspirational

# Himitsu extensibility

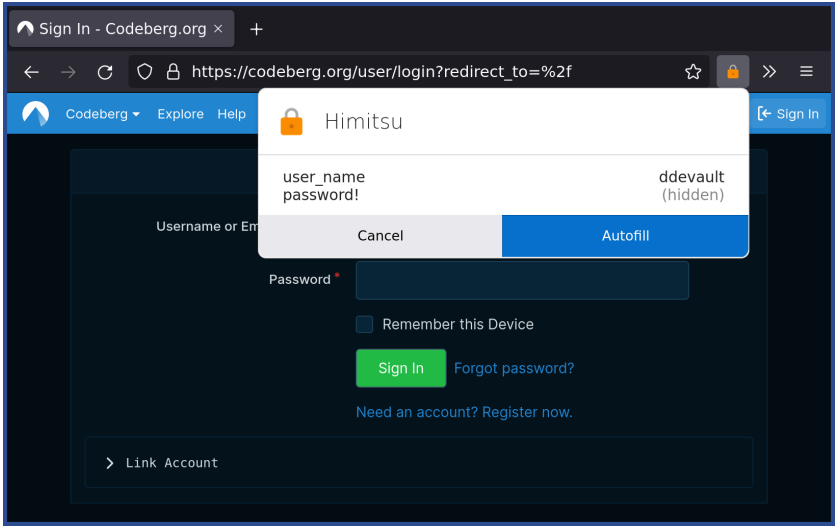
Some specific examples:

- GTK+ prompter, TTY prompter, 2FA via your phone, etc
- SSH agent support and OpenSSH import/export
- Your mail client can grab your creds without additional configuration
- Storing your full disk encryption key in Himitsu

# Prompter protocol

```
<= version
=> version 0.0.0
<= key proto=web host=meta.sr.ht username=admin password!
<= key proto=web host=meta.sr.ht username=ddevault password!
<= unlock
<= prompt disclose unlock
   Prompter presents itself, requests consent from user
=> password hunter
<= incorrect-password
=> password hunter2
<= EOF
=> Exit status 0
```

# Usage with web logins



# Usage with SSH

Provided by a *separate* package: “himitsu-ssh”

```
$ ./hissh-import < ~/.ssh/id_ed25519
Enter SSH key passphrase:
key proto=ssh type=ssh-ed25519 pkey=[...] skey! comment=sircmpwn@homura
$ hissh-agent &
Listening at /tmp/runtime/sircmpwn/hissh-agent/socket
$ export SSH_AUTH_SOCK=/tmp/runtime/sircmpwn/hissh-agent/socket
$ ssh-add -l
256 SHA256:kPr5ZKTNE54TRHGSaanhcQYiJ56zSgcpKeLZw4/myEI sircmpwn@homura (ED25519)
$ ssh drewdevault.com
```

# Usage with... anything else?

Got any ideas? It's probably possible, and easy!

# Security?

- Key store encryption key is derived with argon2id
- Each key is individually encrypted with AEAD using XChaCha20+Poly1305
- Secret values are stored separately, also XChaCha20+Poly1305
- Unix security bits: Linux keyctl(2), mlockall, container-friendly, etc
- **However**: Nothing has been audited yet

# Future ideas: Key store syncronization

Easy to configure, assuming you either set up the sync daemon on your own server, or got someone to host it for you:

```
$ hiq -a proto=sync host=himitsu.sr.ht
# That's it, that's the last step
```

# Future ideas: Early boot Himitsu

1. Put Himitsu in your initramfs along with a minimal key store
2. Put your FDE key in said minimal key store
3. Log in with your username and key store passphrase during early boot
4. Load your FDE key from Himitsu and proceed with boot
5. Automatically logs you in with himitsud already running

# Future ideas: Two-factor prompter

In short: what if instead of the prompter popping up on your laptop, it popped up on your phone instead? Android support? PinePhone support? Is zero configuration possible?

# We need your help!

An incomplete list of things we could use help with:

- himitsu-firefox improvements (web devs welcome!)
- Chromium support (web devs welcome!)
- Apps for phones (mobile devs welcome!)
- Key management frontends (GUI/TUI)
- Turn the security up to 11 – smartcards? U2F?
- More cryptographic primitives for himitsu-ssh
- PGP support
- FDE support
- A bunch of other shit

# Himitsu

https://himitsustore.org
https://sr.ht/~sircmpwn/himitsu