StageX: Rebuilding Trust Through Multi-Signed, Full-Source Bootstrapped, and Reproducible Builds

Danny Grove

@groved@mastodon.social

Lance Vick

@lrvick@mastodon.social

No trust in a single human or computer

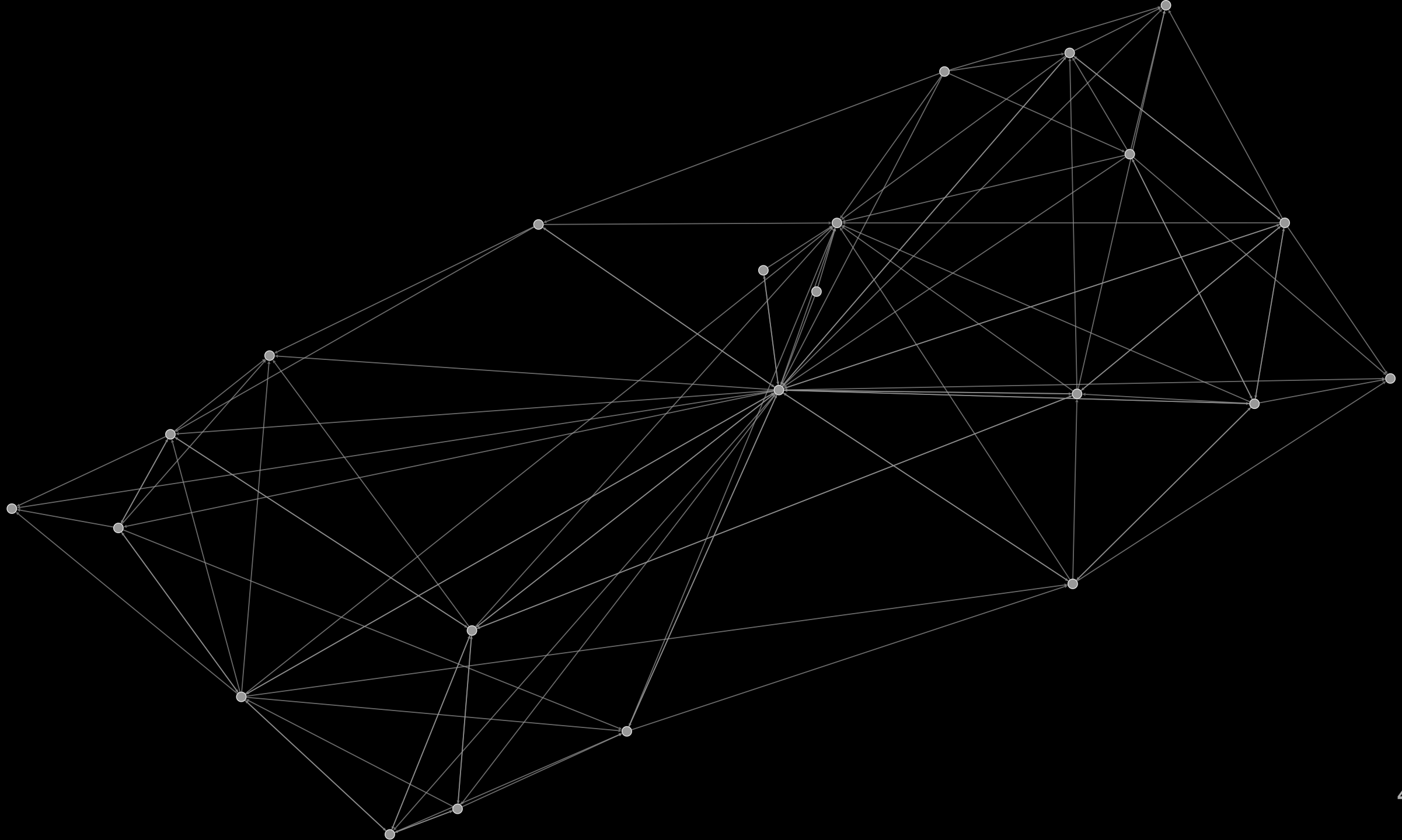
Reproducible

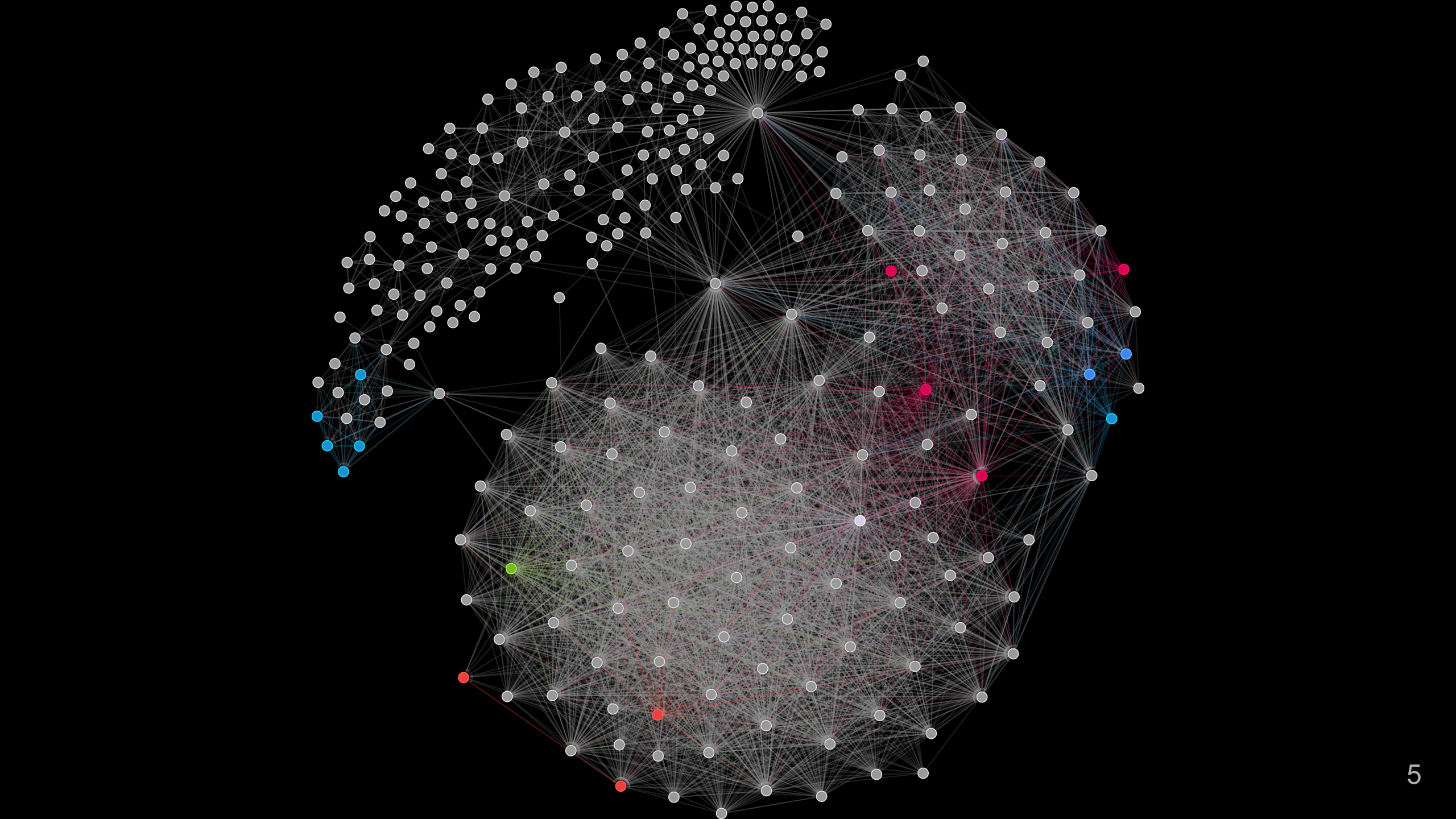
Full-Source Bootstrapped

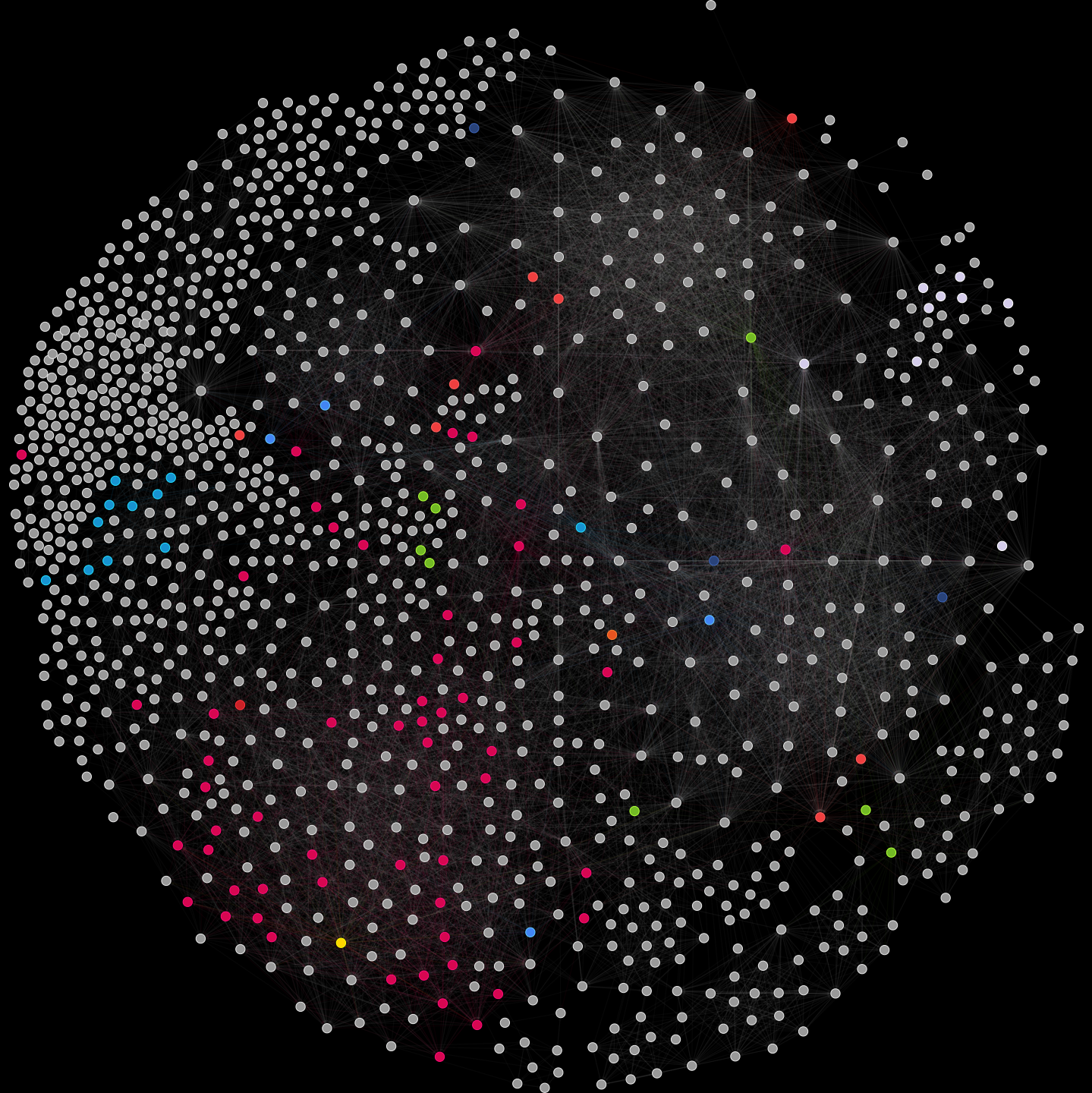
Hermetic

Container native

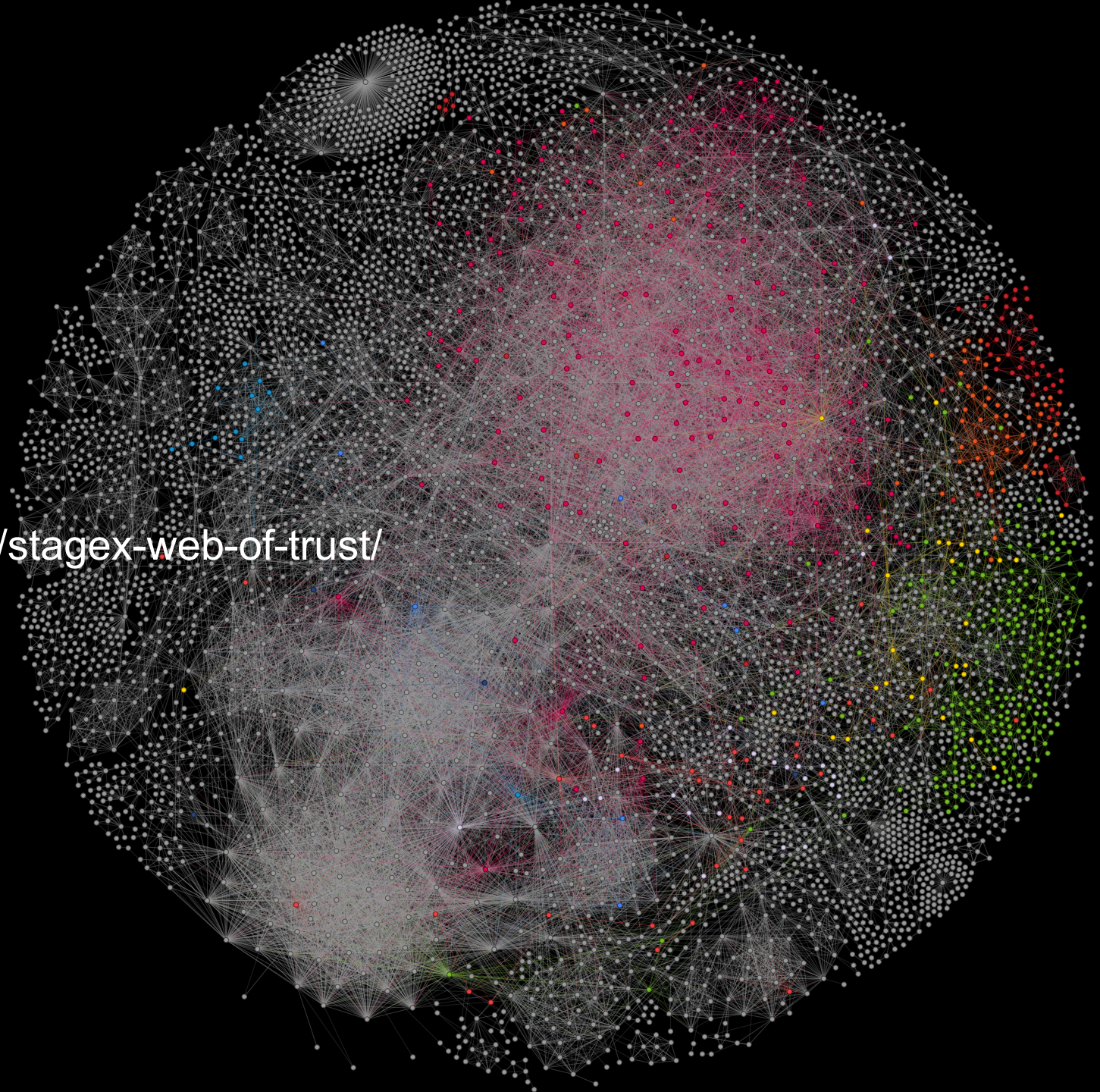
Multi-Signed







https://kron.fi/en/posts/stagex-web-of-trust/



1984

1992

2013

2015

2016

2019

2021

NOW

TURING AWARD LECTURE

Reflections on Trusting Trust

To what extent should one trust a statement that a program is free of Trojan horses? Perhaps it is more important to trust the people who wrote the software.

KEN THOMPSON

INTRODUCTION

I thank the ACM for this award. I can't help but feel that I am receiving this honor for timing and serendipity as much as technical merit. UNIX[1] swept into popu-

programs. I would like to present to you the cutest program I ever wrote. I will do this in three stages and try to bring it together at the end.

1984

1992

2013

2015

2016

2019

2021

NOW

> We shipped that "P3" progressive release in October 1992. Then we released a stability update in December, and further updates every three months for years thereafter. The 93Q2 release of June 1993 is the first release in which we made the results bit-for-bit identical on all platforms except the RS/6000. That is documented in the emails below. Note the progression of dates.
>
> John

https://lists.reproducible-builds.org/pipermail/rb-general/2023-November/003133.html

1984

1992

2013

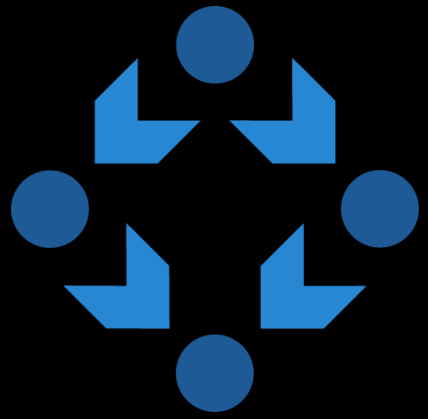
2015

2016

2019

2021

NOW



Reproducible Builds

1984

1992

2013

2015

2016

2019

2021

NOW

I suddenly realize it's in the compiler. It was the compiler. And every time you compile the original code and run it puts in the subliminal message code into the source code

1. it examines any call to `fopen()`, searches the file opened looking for Dr. Phelp's questions; if it finds them then
2. it rewrites the 15 files to the current directory when compiling that specific program.
3. It then compiles Dr. Phelps program using the 15 files and outputs to the `-o` name in the link phase.

Excerpt from "What Is A Coders Worst Nightmare?" (Mick Stute)

1984

1992

2013

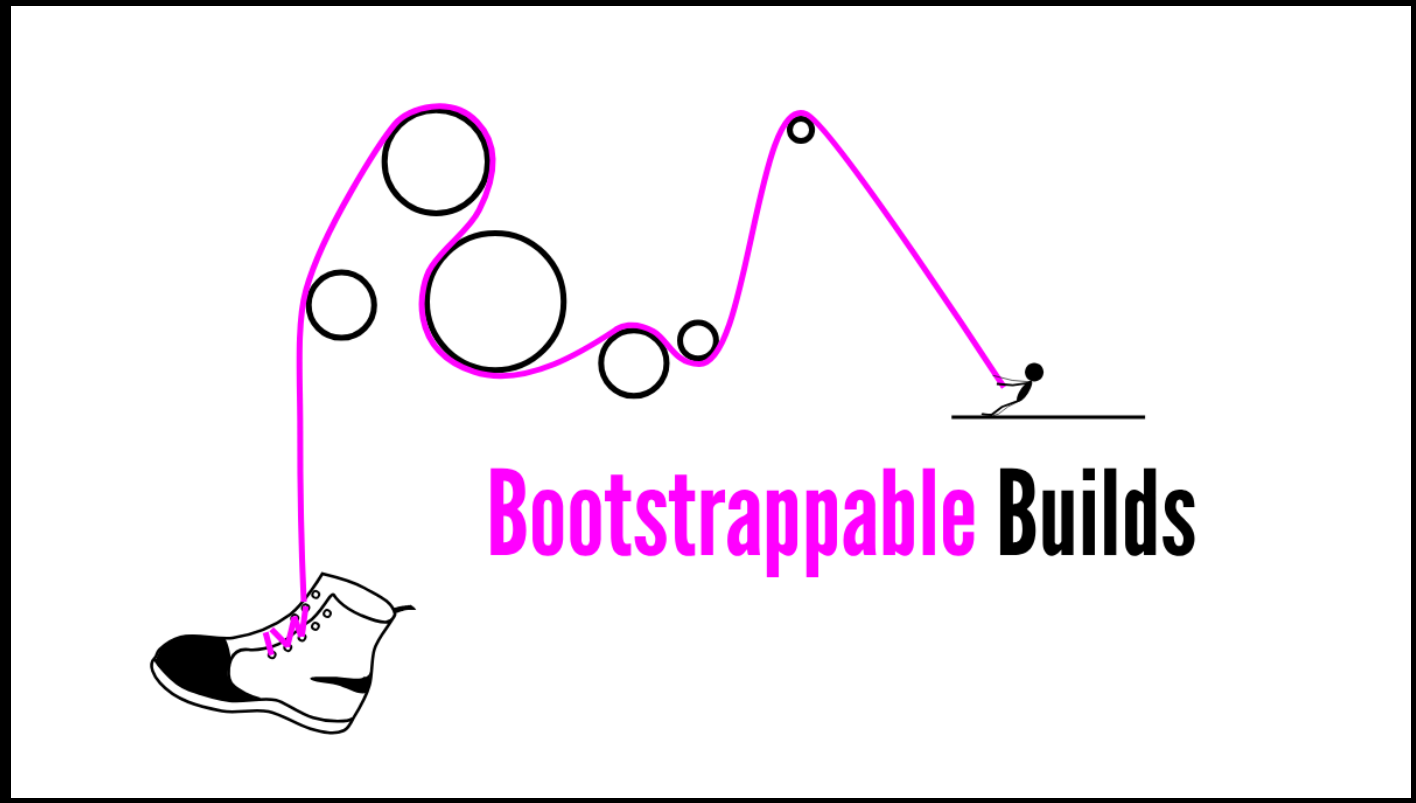
2015

2016

2019

2021

NOW



1984

1992

2013

2015

2016

2019

2021

NOW

> *The holy grail for bootstrappability will be connecting hex0 to mes.*

— Carl Dong

1984

1992

2013

2015

2016

2019

2021

NOW

https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/

1984

1992

2013

2015

2016

2019

2021

NOW


Distribution Maintainers
Source Code
Build Process
Binary Packages
User Trust

Supply chain attack hits 100 million-download Axios npm package

TeamPCP Hacks Checkmarx GitHub Actions Using Stolen CI Credentials

Ravie Lakshmanan Mar 24, 2026 DevSecOps / Vulnerability

Popular LiteLLM PyPI package backdoored to steal credentials, auth tokens

By **Lawrence Abrams**

March 24, 2026 06:29 PM 0

2003

2011

2016

2024

2025

Hacked Gentoo Linux server taken offline

PATRICK GRAY

PUBLISHED DECEMBER 4, 2003

Update on compromised Debian machines

- *To*: debian-devel-announce@lists.debian.org
- *Subject*: Update on compromised Debian machines
- *From*: Wichert Akkerman <wichert@wiggy.net>
- *Date*: Fri, 21 Nov 2003 17:47:31 +0100
- *Message-id*: <[🔎] 20031121164731.GB18224@wiggy.net>
- *Mail-followup-to*: Wichert Akkerman <wichert@wiggy.net>, debian-devel-announce@lists.debian.org

FYI: ftp.gnu.org compromised

From: John Cartwright <johnc () grok org uk>
Date: Wed, 13 Aug 2003 21:16:58 +0100

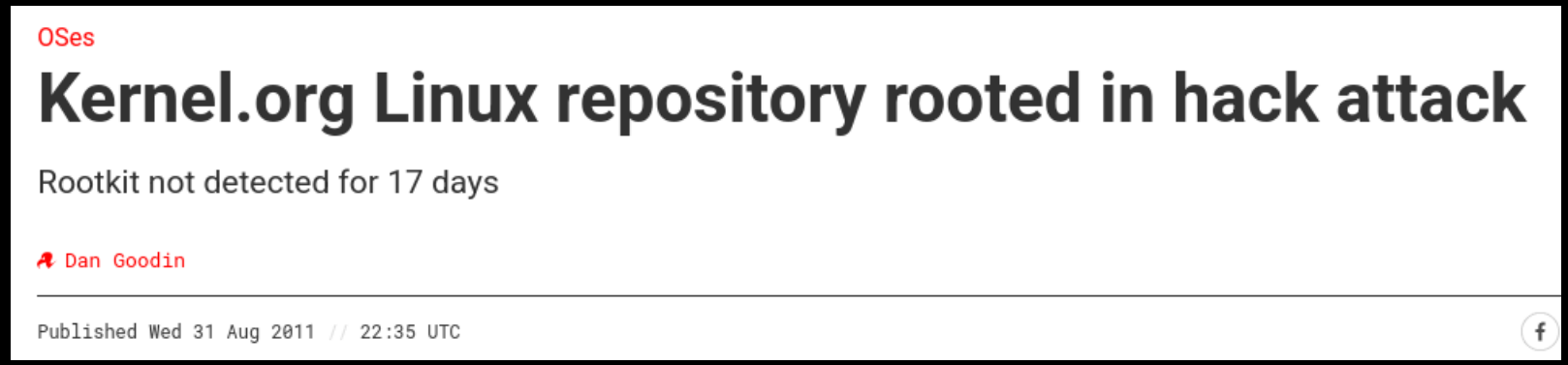
2003

2011

2016

2024

2025



OSes

Kernel.org Linux repository rooted in hack attack

Rootkit not detected for 17 days

Dan Goodin

Published Wed 31 Aug 2011 // 22:35 UTC

2003

2011

2016

2024

2025

OSes

Linux Mint hacked: Malware-infected ISOs linked from official site

Downloaded Linux Mint on February 20th? Check for infection NOW

Tim Anderson

Published Sun 21 Feb 2016 // 14:05 UTC

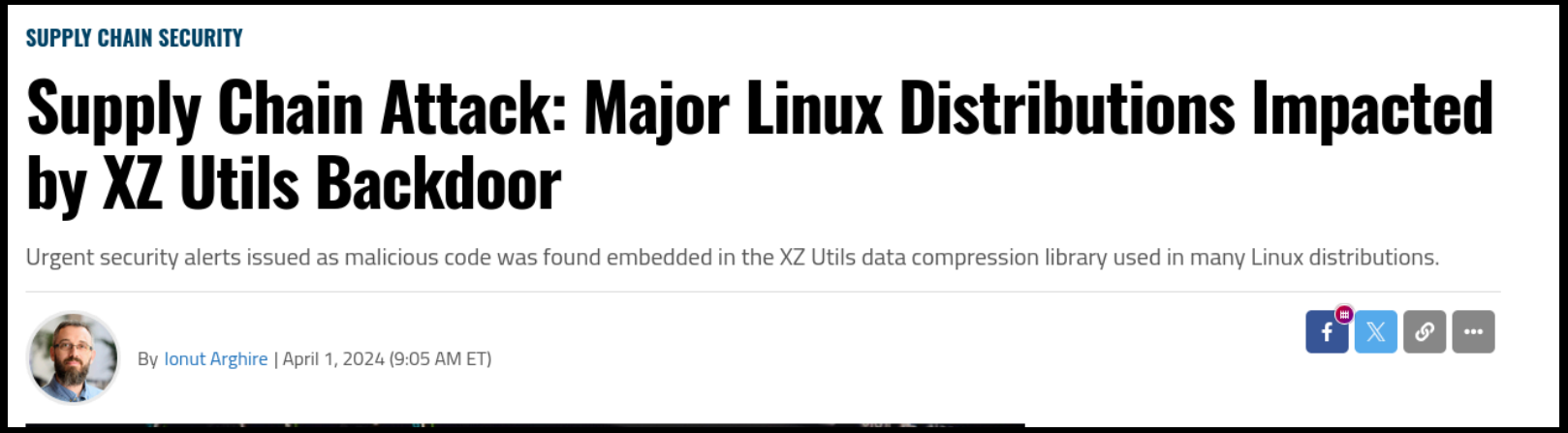
2003

2011

2016

2024

2025



SUPPLY CHAIN SECURITY

Supply Chain Attack: Major Linux Distributions Impacted by XZ Utils Backdoor

Urgent security alerts issued as malicious code was found embedded in the XZ Utils data compression library used in many Linux distributions.

By Ionut Arghire | April 1, 2024 (9:05 AM ET)

2003

2011

2016

2024

2025

LWN.net
News from the source

Content
Weekly Edition
Archives
Search
Kernel
Security
Events calendar
Unread comments

LWN FAQ
Write for us

Edition
Return to the Briefs page

User: Password: Log in | Subscribe | Register

Supply Chain Attacks on Linux distributions (Fenrisk)

[Posted March 19, 2025 by corbet]

A security company called Fenrisk has posted an overview of a pair of claimed successful supply-chain attacks on the Fedora and openSUSE distributions.

> We successfully identified vulnerabilities in the Pagure, the Git forge used by Fedora to store their package definitions. We also compromised Open Build Service, the all-in-one toolchain used and developed by the openSUSE project for compilation and packaging.
>
> Their exploitation by malicious actors would have led to the compromise of all the packages of the distributions Fedora and openSUSE, as well as their downstream distributions, impacting millions of Linux servers and desktops.

[**Update**: SUSE has put out a statement about the vulnerability; "While this is a serious vulnerability that needed to be fixed quickly, the impact was inaccurately described."]

> m

>

>

```
/* We will NOT put a fucking timestamp in the header here. Every time you
   put it back, I will come in and take it out again.  I'm sorry.  This
   field does not belong here.  We fill it with a 0 so it compares the
   same but is not a reasonable time. -- gnu at cygnus.com  */
internal_f.f_timdat = 0;
```

Technical Decisions

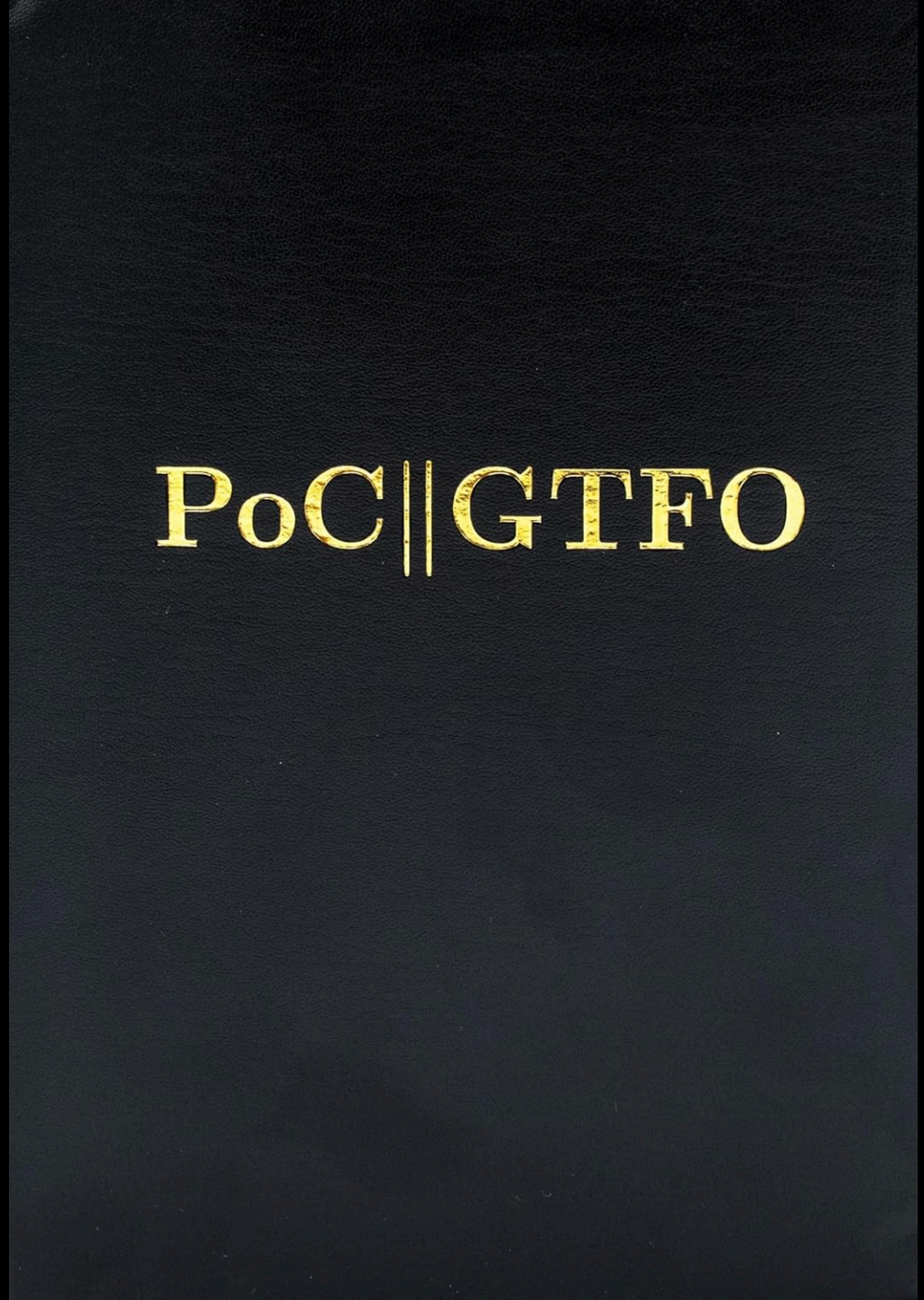
Choice	Reason
OCI	Standard, Multiple Implementations, Multisig
LLVM	Native Cross-Compliation for most architectures
musl	Modern, minimal
mimalloc	security focused, fast
mold	fast (twice as slow as cp)

```
FROM scratch AS sources
ARG STAGE0_POSIX_X86_SOURCE
ARG M2_MESOPLANET_SOURCE
ARG M2_PLANET_SOURCE
ARG M2LIBC_SOURCE
ARG MESCC_TOOLS_SOURCE
ARG MESCC_TOOLS_EXTRA_SOURCE
ADD fetch/${STAGE0_POSIX_X86_SOURCE} .
ADD fetch/${M2_MESOPLANET_SOURCE} .
ADD fetch/${M2_PLANET_SOURCE} .
ADD fetch/${M2LIBC_SOURCE} .
ADD fetch/${MESCC_TOOLS_SOURCE} .
ADD fetch/${MESCC_TOOLS_EXTRA_SOURCE} .

FROM scratch AS seeds
ARG STAGE0_POSIX_X86_VERSION
COPY --from=sources /stage0-posix-x86-${STAGE0_POSIX_X86_VERSION} /x86
ADD hex0-seed /
WORKDIR /build
RUN ["/hex0-seed", "/x86/hex0_x86.hex0", "hex0"]
RUN ["./hex0", "/x86/kaem-minimal.hex0", "kaem-minimal"]

...
```

```
FROM stagex/bootstrap-stage3 AS build
ARG TARGETARCH
ARG VERSION
ADD fetch/make-${VERSION}.tar.gz .
WORKDIR /make-${VERSION}
RUN --network=none <<-EOF
  ./configure \
    --prefix=/usr \
    --mandir=/usr/share/man \
    --infodir=/usr/share/info \
    --disable-nls
  make -j "$(nproc)"
  make DESTDIR="/rootfs" install
EOF
FROM stagex/core-filesystem AS package
COPY --from=build /rootfs/ /
```


PoC||GTFO

```
FROM stagex/pallet-clang

COPY <<-EOF hello.c
  #include <stdio.h>
  int main() {
    printf("Hello, World!");
    return 0;
  }
EOF

RUN ["/usr/bin/clang","hello.c"]
```

```
# syntax=docker/dockerfile@sha256:b6afd42430b15f2d2a4c5a02b919e98a525b785b1aaff16747d2f623364e39b6
FROM --platform=amd64 stagex/pallet-go@sha256:5477bcf690aa52d1afdd2ed4ed0e6cc661cabf028a93ac639106b2ad06d7fa9a AS pallet-go

FROM pallet-go AS build
ARG TARGETOS
ARG TARGETARCH

ENV GOOS=${TARGETOS}
ENV GOARCH=${TARGETARCH}
ADD . /containerfile-updater
WORKDIR /containerfile-updater
RUN go mod download
RUN --network=none <<-EOF
  set -eu
  go build \
    -trimpath \
    -v \
    -mod=readonly \
    .
  install -Dm0755 -t /rootfs-${TARGETOS}-${TARGETARCH}/usr/bin/ containerfile-updater
  install -Dm0644 -t /rootfs-${TARGETOS}-${TARGETARCH}/usr/share/licenses/containerfile-updater/ LICENSE
  install -Dm0644 -t /rootfs-${TARGETOS}-${TARGETARCH}/usr/share/licenses/containerfile-updater/ COPYRIGHT
  install -Dm0644 -t /rootfs-${TARGETOS}-${TARGETARCH}/etc/ssl/certs/ /etc/ssl/certs/ca-certificates.crt
EOF

FROM scratch AS package
ARG TARGETOS
ARG TARGETARCH
COPY --from=build /rootfs-${TARGETOS}-${TARGETARCH}/ /
```

```
FROM stagex/pallet-rust@sha256:9c38bf1066dd9ad1b6a6b584974dd798c2bf798985bf82e58024fbe0515592ca AS builder

COPY . /app
WORKDIR /app

ENV RUSTFLAGS="-C target-feature=+crt-static"
ENV CARGO_TARGET_DIR=/tmp/cargo-target
RUN cargo build --release --target x86_64-unknown-linux-musl

FROM scratch

COPY --from=builder /tmp/cargo-target/x86_64-unknown-linux-musl/release/steve /usr/bin/steve
EXPOSE 1337
ENTRYPOINT ["/usr/bin/steve"]
```

Native Multi-Arch Builds

Broad Cross-Compile Support

Automatic Build Validation

Decentralized Mirror Validation

Decentralized Source Code Review

Questions?

- https://stagex.tools
- https://matrix.to/#/#stagex:matrix.org
- ircs://irc.oftc.net:6697/#stagex
- https://docs.stagex.tools
- https://codeberg.org/stagex/stagex
- https://codeberg.org/stagex/whitepaper

	StageX	Guix	Debian	Arch	Nix	Yocto	Buildroot	Chimera	Alpine	Fedora
Trust model	Decentralized	Distributed	Distributed	Distributed	Centralized	Centralized	Centralized	Centralized	Centralized	Centralized
OCI	Native	Exported	Published	Published	Exported	Exported	Exported	Published	Published	Published
Language	Containerfile	Custom	Custom	Shell	Custom	Custom	Makefile	Python	Shell	Custom
Bootstrapped	Yes	Yes	No	No	Partial	No	No	No	No	No
Reproducible	Yes	Mostly	Mostly	Mostly	Mostly	No	No	No	No	No
Toolchain base	LLVM	GNU	GNU	GNU	GNU	GNU	GNU	LLVM	GNU	GNU
C standard library	musl	glibc	glibc	glibc	glibc	glibc	glibc	musl	musl	glibc
Memory allocator	mimalloc	glibc	glibc	glibc	glibc	glibc	glibc	mimalloc	mallocng	glibc