

**SUPPLY CHAIN SECURITY**

**+**

**THE XZ BACKDOOR**

**DAVID GSTIR + RICHARD WEINBERGER**  
**SIGMA STAR GMBH**

## DAVID GSTIR

- > Focus on cryptography, security protocols and Linux kernel
- > Manages penguins at sigma star gmbh

## RICHARD WEINBERGER

- > Focus on Linux kernel, low-level components, virtualization, security
- > Co-founder of sigma star gmbh



# DISCLAIMER

- > THIS IS A 30MIN TALK
- > WE WON'T COVER EVERYTHING
- > FEEL FREE TO TALK TO US  
AFTERWARDS



**WHAT IS A SUPPLY CHAIN?**



**MOST OBVIOUS:**

- > Libraries
- > Frameworks
- > SDKs
- > Closed and Open Source

## **LESS OBVIOUS:**

Everything else that transforms your source into the binary/artifact you ship like CI/CD pipeline, IDE, shell, OS, Firmware in HW components, etc.



**DO YOU KNOW YOUR SUPPLY  
CHAIN?**



- > Do you have a list of all your dependencies?
- > How often do you update dependencies?
- > How do you check dependencies for updates?
- > Do you use containers/Kubernetes/Cloud service?
- > What about your (build) infrastructure?
- > How do you check your dependencies for vulnerabilities? Review? Scanners?
- > How do you select dependencies?



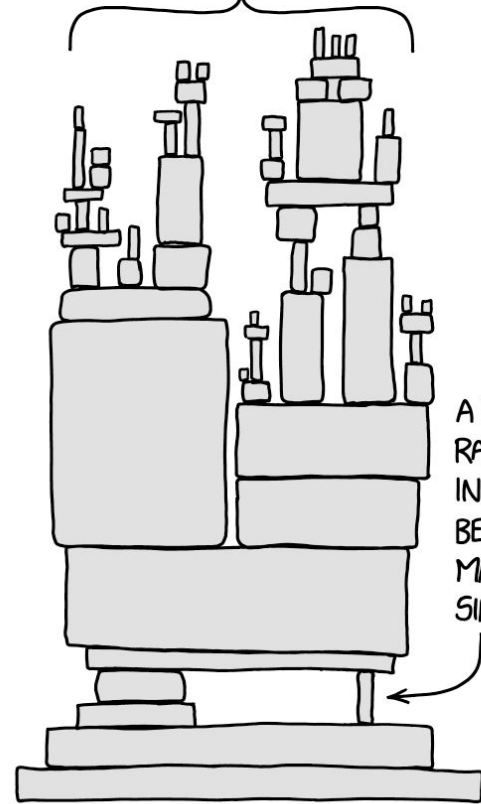


# *SUPPLY CHAIN THREATS*

# SUPPLY CHAIN THREATS

- > Availability: Is there support? For how long?
- > Control: You use foreign code
- > And ...

ALL MODERN DIGITAL  
INFRASTRUCTURE




A PROJECT SOME  
RANDOM PERSON  
IN NEBRASKA HAS  
BEEN THANKLESSLY  
MAINTAINING  
SINCE 2003

# ... IT CAN INTRODUCE SECURITY VULNERABILITIES

## EXAMPLES:

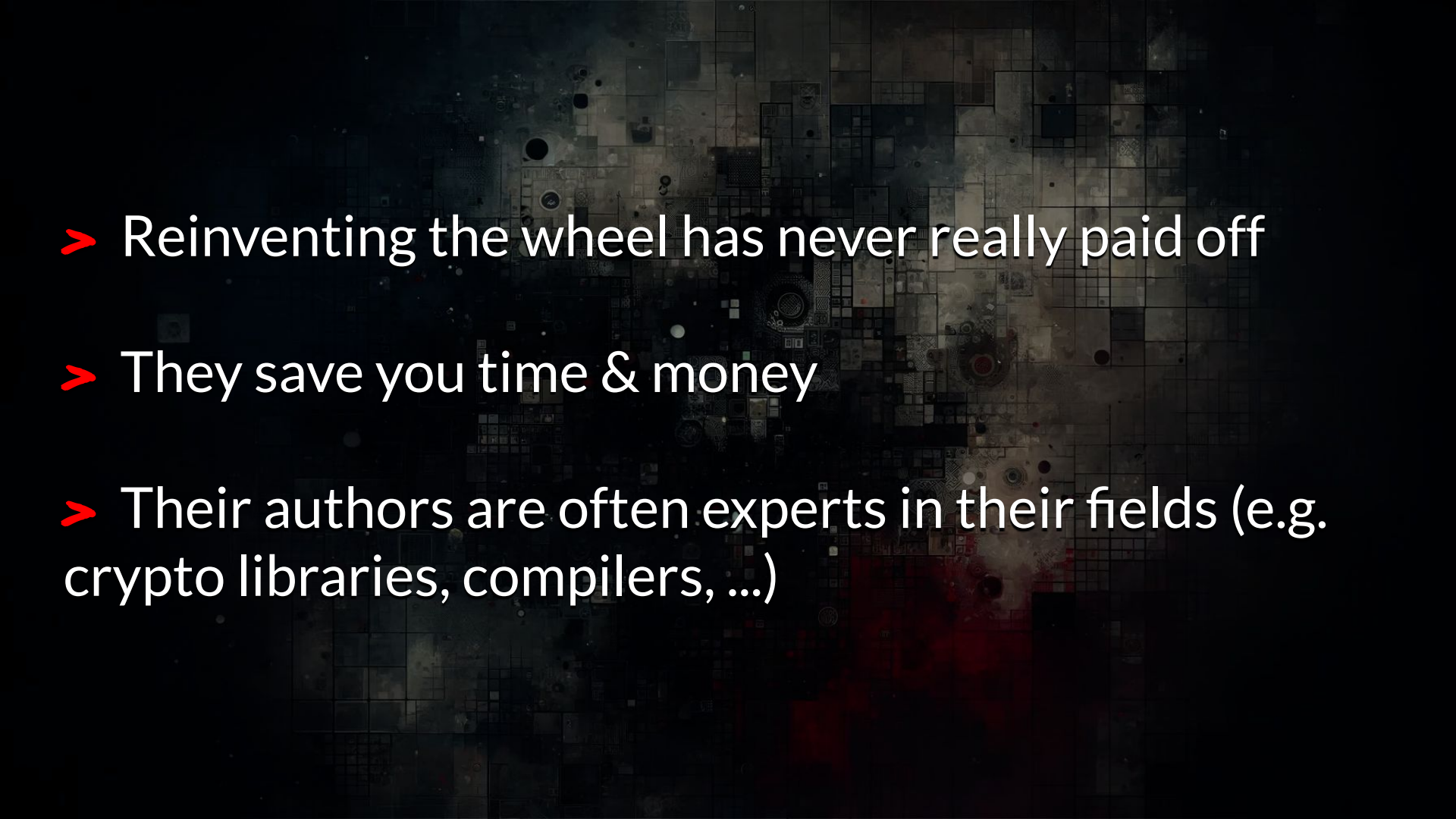
- > Debian OpenSSL fiasco
- > log4j
- > SolarWinds backdoor
- > xz backdoor





**SO, DEPENDENCIES ARE  
BAD?**

**NO!**

- 
- > Reinventing the wheel has never really paid off
  - > They save you time & money
  - > Their authors are often experts in their fields (e.g. crypto libraries, compilers, ...)



**HOW DO WE MAKE A  
SUPPLY CHAIN MORE  
SECURE THEN?**

The background is a dark, textured surface with a grid pattern, possibly representing a floor or a wall. There are various shades of grey and black, with some lighter areas and a prominent red stain in the bottom right corner. The text is written in a bold, red, stylized font with a slight shadow effect.

**...BY LEARNING FROM  
PAST MISTAKES**





# *THE XZ BACKDOOR*

## OVERVIEW + TIMELINE

- > xz-utils (short xz) liblzma contained malicious code
- > Solely maintained by Lasse Collin in his spare time
- > 2021: “Jia Tan” arrives and starts contributing
- > 2022: Various (likely fake) people start pressuring Lasse that development is slow
- > Lasse pushes back but “Jia Tan” is suggested as co-maintainer
- > “Jin Tan” becomes co-maintainer soon after - stealthy, hostile project takeover achieved
- > 2023-06: First ground work for backdoor submitted by (likely fake) contributor “Hans Jansen”



## TIMELINE (CONTD)

- > 2024-02-23: “Jia Tan” merges new *binary test files* which in reality contain backdoor code
- > 2024-02-24: “Jia Tan” builds and tags v5.6.0 and publishes xz-5.6.0.tar.gz
- > **Release tarball contains crafted build script**
- > xz v5.6.0 lands in various (rolling release) distros like Debian unstable, openSUSE Tumbleweed, Fedora 40, but also macOS Homebrew
- > Backdoor triggers some warnings which lead to (stealthy) fix and release v5.6.1
- > 2024-03-28: Backdoor accidentally detected by Andres Freund because of *slow SSH login*



# THE BACKDOOR

- > Injects malicious code into sshd (OpenSSH) on all Debian (including Ubuntu and friends) and RPM based distros
- > liblzma is an *indirect* dependency to sshd via libsystemd
- > Backdoor executes `system()`, controlled via magic packet
- > Command is encoded in SSH certificate sign key
- > You need the correct Ed448 private key signature to use the backdoor; disable switch via ENV variable
- > Full feature set of backdoor is still unknown



# POST MORTEM

WHAT WENT WRONG AND HOW TO GET  
BETTER?

## Q: HOW COULD THE BACKDOOR HAVE BEEN DETECTED?

- > Well, it was detected
- > Through an unexpected way, but still
- > Detection will get harder - sophisticated attack
- > More eyes (i.e. users) always help
- > Everybody needs to take more care when selecting and updating dependencies



## Q: WHY WASN'T THIS DETECTED BEFORE THE RELEASE?

- > The backdoor was very well hidden
- > The attacker was one of the maintainers
- > Activation logic only in release tarballs, not repo
- > Backdoor code was hidden (obfuscated, encrypted) in binary test file



# REAL QUICK! SPOT THE BUG

```
--- a/CMakeLists.txt
+++ b/CMakeLists.txt
@@ -901,10 +901,29 @@ endif()

# Sandboxing: Landlock
if(NOT SANDBOX_FOUND AND ENABLE_SANDBOX MATCHES "^ON$|^landlock$")
-   check_include_file(linux/landlock.h HAVE_LINUX_LANDLOCK_H)
+   # A compile check is done here because some systems have
+   # linux/landlock.h, but do not have the syscalls defined
+   # in order to actually use Linux Landlock.
+   check_c_source_compiles("
+       #include <linux/landlock.h>
+       #include <sys/syscall.h>
+       #include <sys/prctl.h>
+
+       void my_sandbox(void)
+       {
+           (void)prctl(PR_SET_NO_NEW_PRIVS, 1, 0, 0, 0);
+           (void)SYS_landlock_create_ruleset;
+           (void)SYS_landlock_restrict_self;
+           (void)LANDLOCK_CREATE_RULESET_VERSION;
+           return;
+       }
+
+       int main(void) { return 0; }
+   "
+   HAVE_LINUX_LANDLOCK)
-   if(HAVE_LINUX_LANDLOCK_H)
-       set(SANDBOX_COMPILE_DEFINITION "HAVE_LINUX_LANDLOCK_H")
+   if(HAVE_LINUX_LANDLOCK)
+       set(SANDBOX_COMPILE_DEFINITION "HAVE_LINUX_LANDLOCK")
+       set(SANDBOX_FOUND ON)

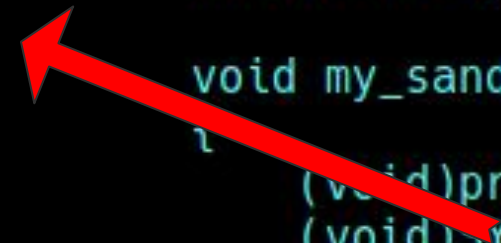
# Of our three sandbox methods, only Landlock is incompatible
```





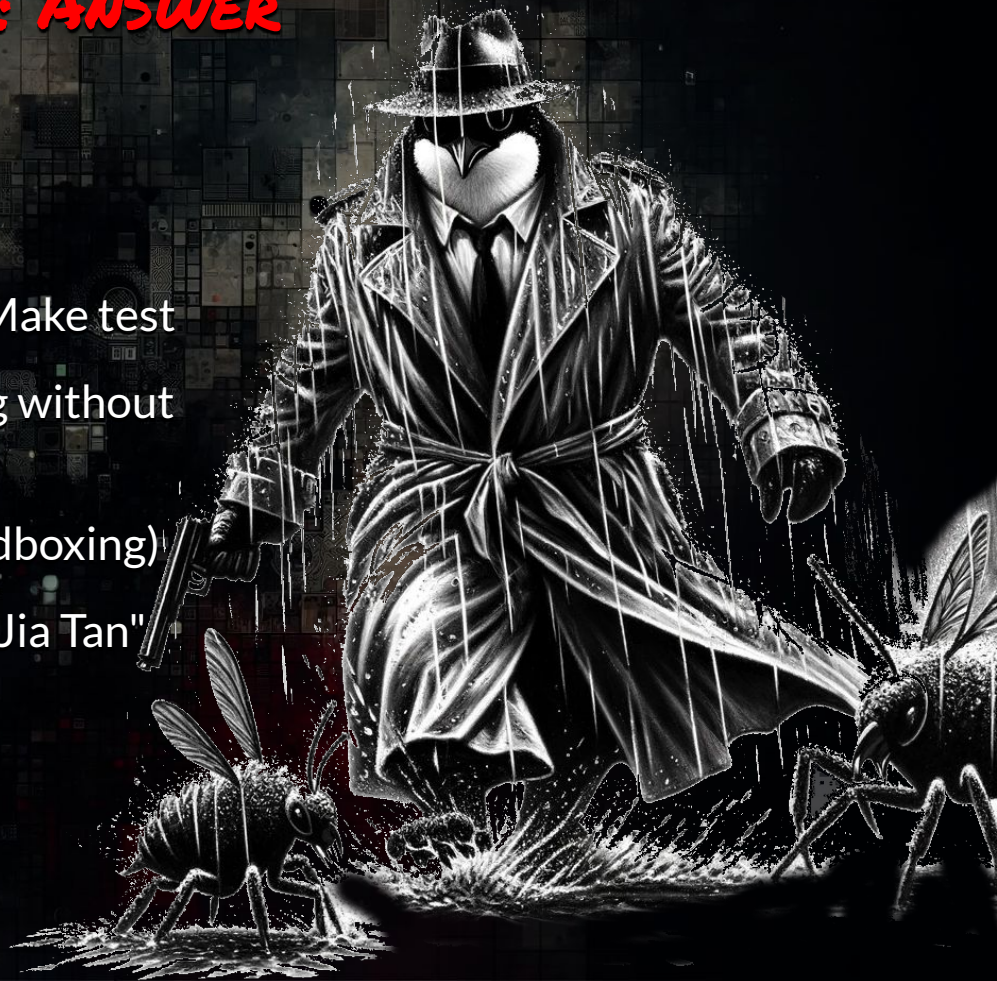
# REAL QUICK! SPOT THE BUG

```
+ # linux/landlock.h, but
+ # in order to actually
+ check_c_source_compile
+ #include <linux/landlock.h>
+ #include <sys/syscall.h>
+ #include <sys/prctl.h>
+
+ void my_sandbox(void)
+ {
+     (void)prctl(PR_SET_NO_NEW_PRIVS, 1, 0, 0, 0);
+     (void)SYS_landlock_restrict_self(0, 0);
+     (void)SYS_landlock_restrict_self(0, 0);
+     (void)LANDLOCK_CREATE_FILE_DESCRIPTOR(0);
+ }
```



## REAL QUICK! SPOT THE BUG: ANSWER

- > The dot (.) *silently* fails the build of this CMake test
- > This will always results in CMake building without landlock support
- > Landlock is a Linux Security Module (sandboxing)
- > It is not fully clear why this was done by "Jia Tan"



## Q: WHY WASN'T THIS DETECTED BY THE DISTRO MAINTAINERS?

- > Andres Freund was faster ;-)
- > They're not used to such attacks
- > Let's hope RHEL or SLES would have
- > They got a free training on supply chain attacks



## Q: WOULD CLOSED SOURCE SOFTWARE HAVE HELPED?

- > Not at all!
- > Does your threat model include a hostile employee?
- > See NSA & Edward Snowden saga
- > ... or SolarWinds and Microsoft
- > Open Sources gives you a chance to spot bugs





**WHAT WE CAN LEARN**

# LEARNINGS FOR SOFTWARE ENGINEERS

- > At some point you need to trust someone, choose them with care
- > Keep your dependency tree in shape (size, known, updated, reviewed)
- > Complex build systems should be simplified
- > Auxiliary code like tests can harm you
- > Non-reproducible binaries in repo should be avoided whenever possible
- > Continuously reassess your dependencies: a “good” one might turn “bad” in the future
- > Recurring audits of code *and dependencies* should be normal



# LEARNINGS FOR USING OPEN SOURCE

- > Determine the “health” of a projects community before using it use metrics like issues, age, number of maintainers, ...
- > Contribute back to open source projects
- > Money is good, but man power is sometime more useful
- > Contribute back what you change and fix!



## SUMMARY

- > There are no silver bullets
- > xz is an extreme example
- > Know your supply chain
- > Review code
- > This is not about Open vs. Closed Source
- > Investigate unexpected behaviour
- > A healthy community is important
- > Together we're strong!





**FIN**

QUESTIONS?

{DAVID,RICHARD}@SIGMA-STAR.AT

# LINKS

<https://www.openwall.com/lists/oss-security/2024/03/29/4>

<https://research.swtch.com/xz-timeline>

<https://boehs.org/node/everything-i-know-about-the-xz-backdoor>

<https://github.com/amlweems/xzbot>

<https://www.schneier.com/blog/archives/2024/04/backdoor-in-xz-utils-that-almost-happened.html>

<https://gynvael.coldwind.pl/?lang=en&id=782>

<https://securelist.com/xz-backdoor-story-part-1/112354/>

<https://gist.github.com/thesamesam/223949d5a074ebc3dce9ee78baad9e27>

pictures: <https://chat.openai.com>

