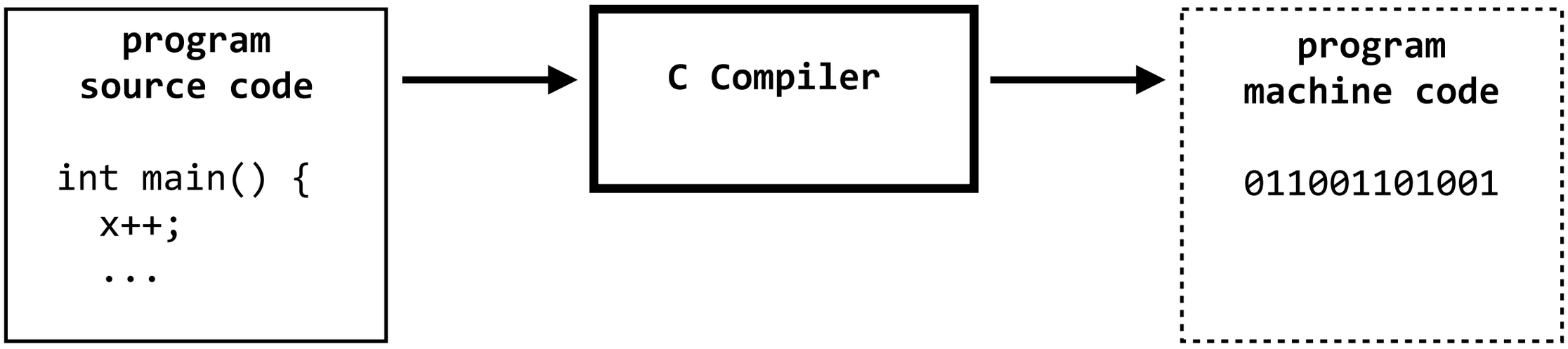
6.033 Spring 2019
Lecture #26

- **Low-level exploits**

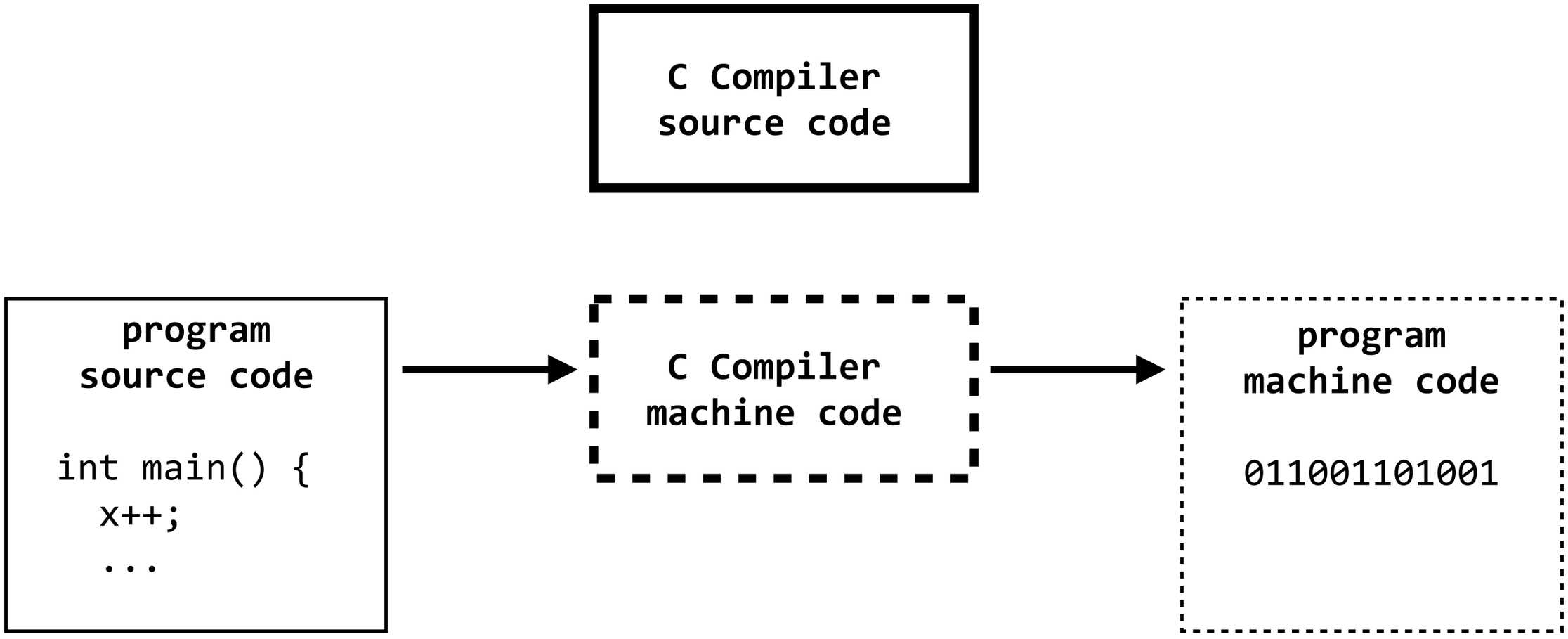
can we prevent buffer overflows?

why not do bounds checking?

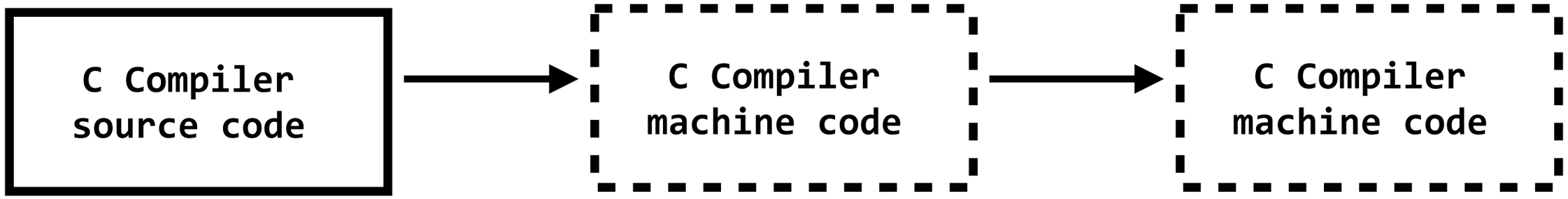
compilers: can we trust them?



compilers: can we trust them?



compilers: can we trust them?

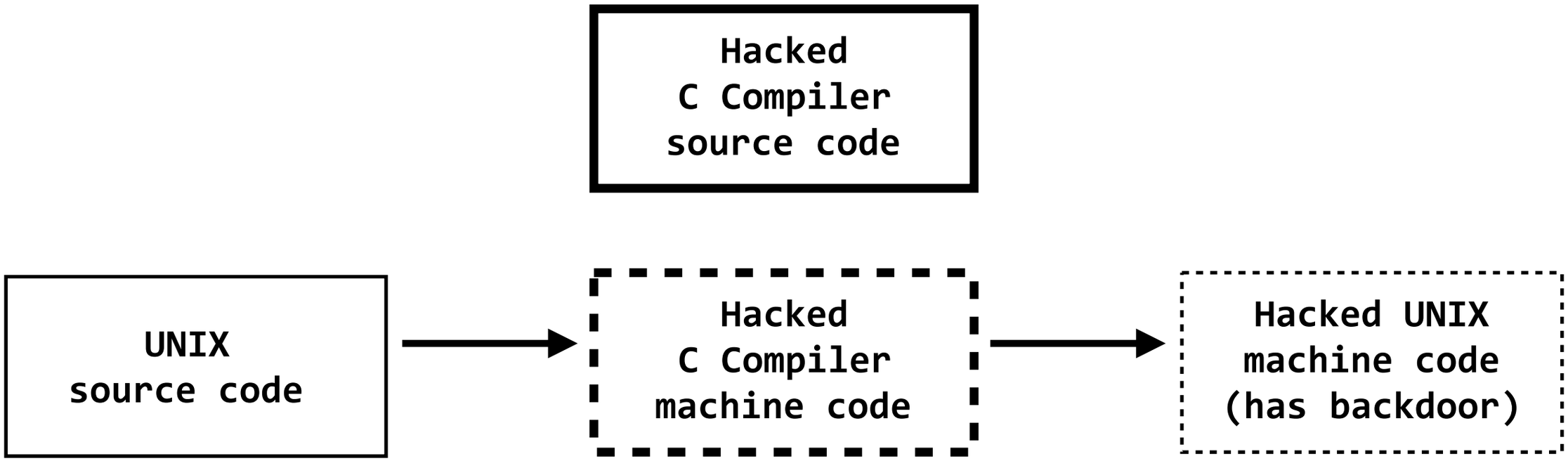


compilers: can we trust them?

compilers: can we trust them?

this backdoor is easily discovered in the hacked UNIX source

compilers: can we trust them?

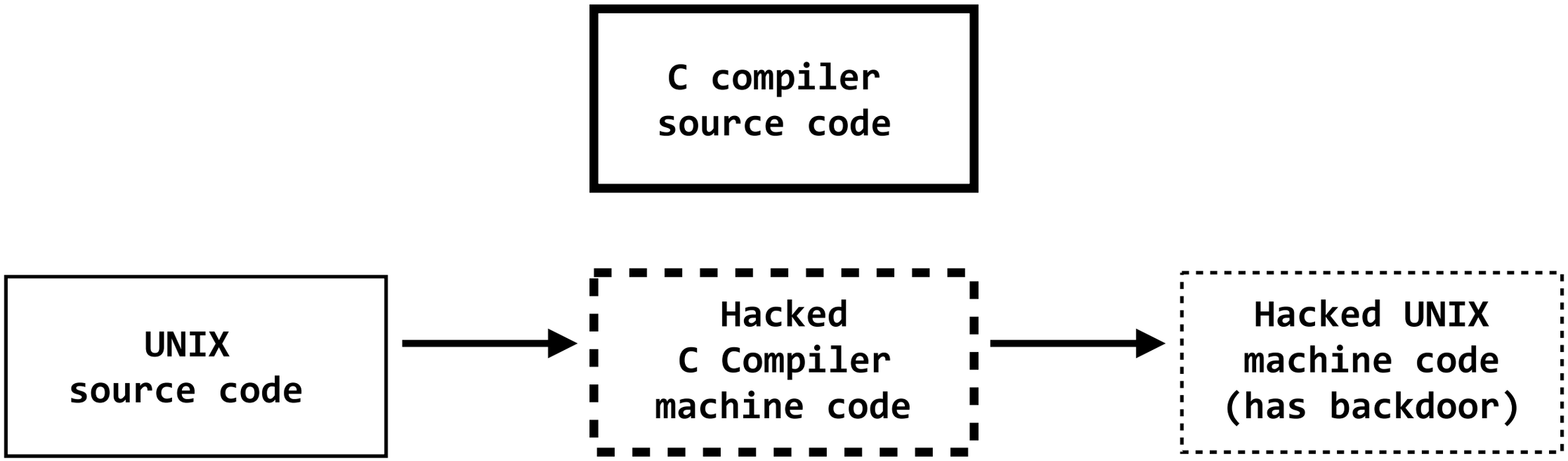


The hacked C compiler has code
that *inserts* a backdoor into UNIX

this backdoor *does not* exist in the UNIX source…

but it does exist in the hacked C Compiler source

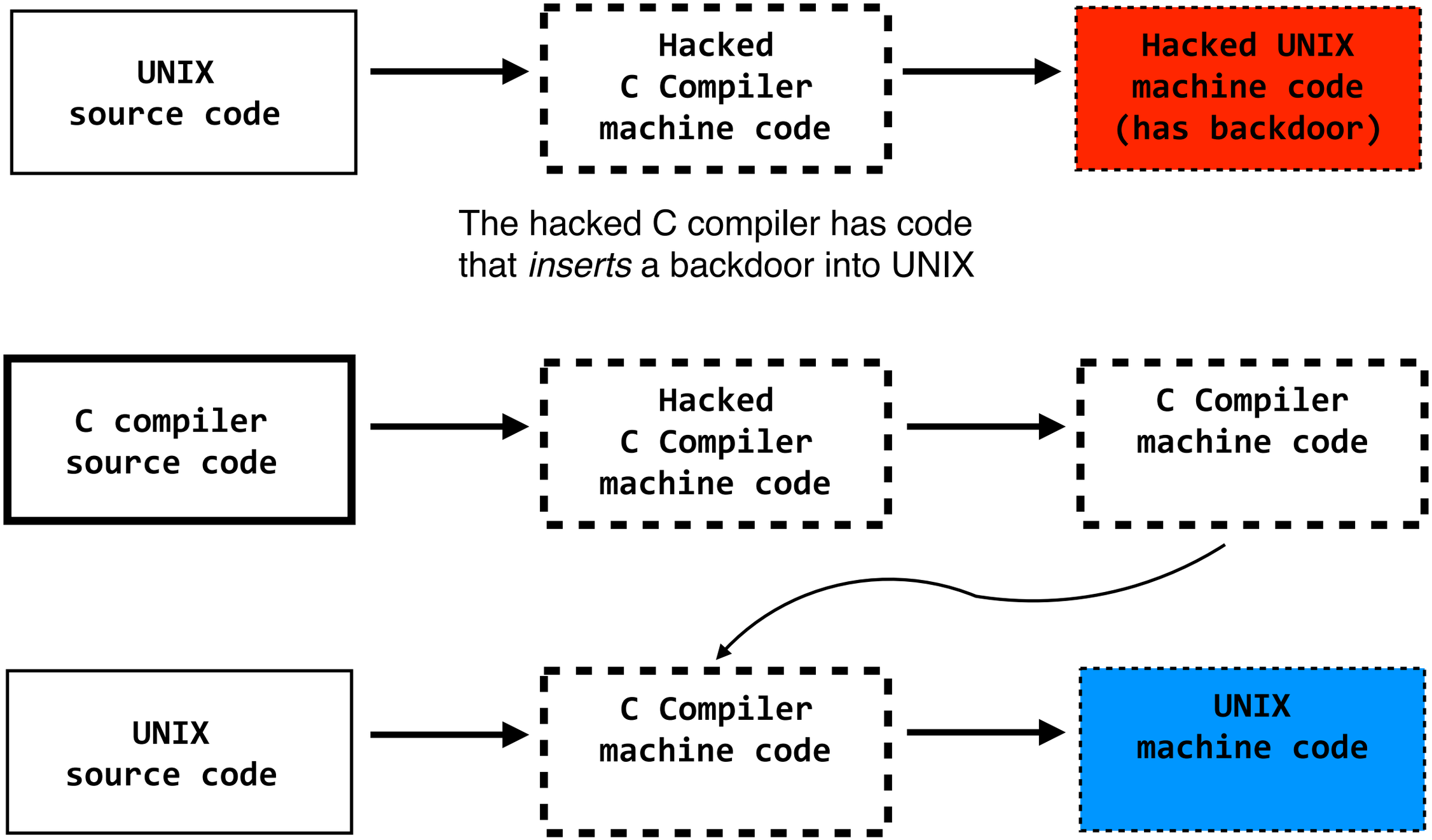
compilers: can we trust them?



The hacked C compiler has code
that *inserts* a backdoor into UNIX

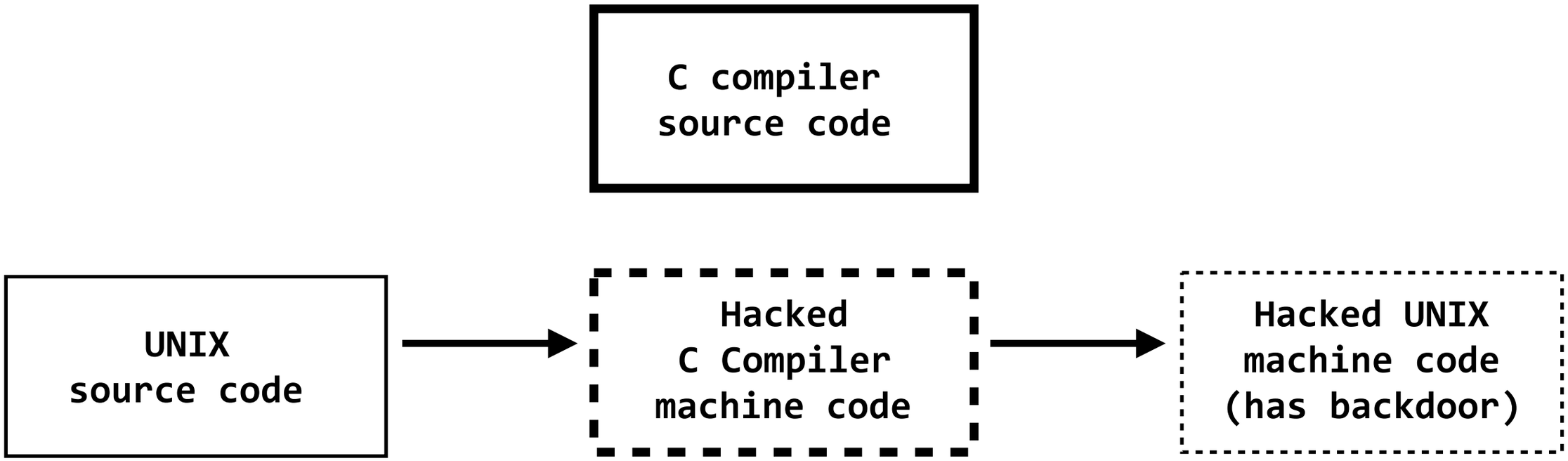
what if i just lie, and tell you that the hacked C compiler was generated from the clean C compiler source? can you check?

compilers: can we trust them?



The hacked C compiler has code that *inserts* a backdoor into UNIX

compilers: can we trust them?

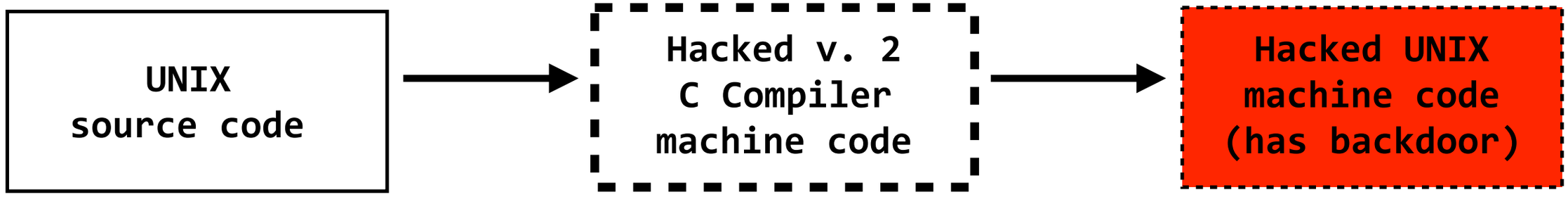


The hacked C compiler has code
that *inserts* a backdoor into UNIX

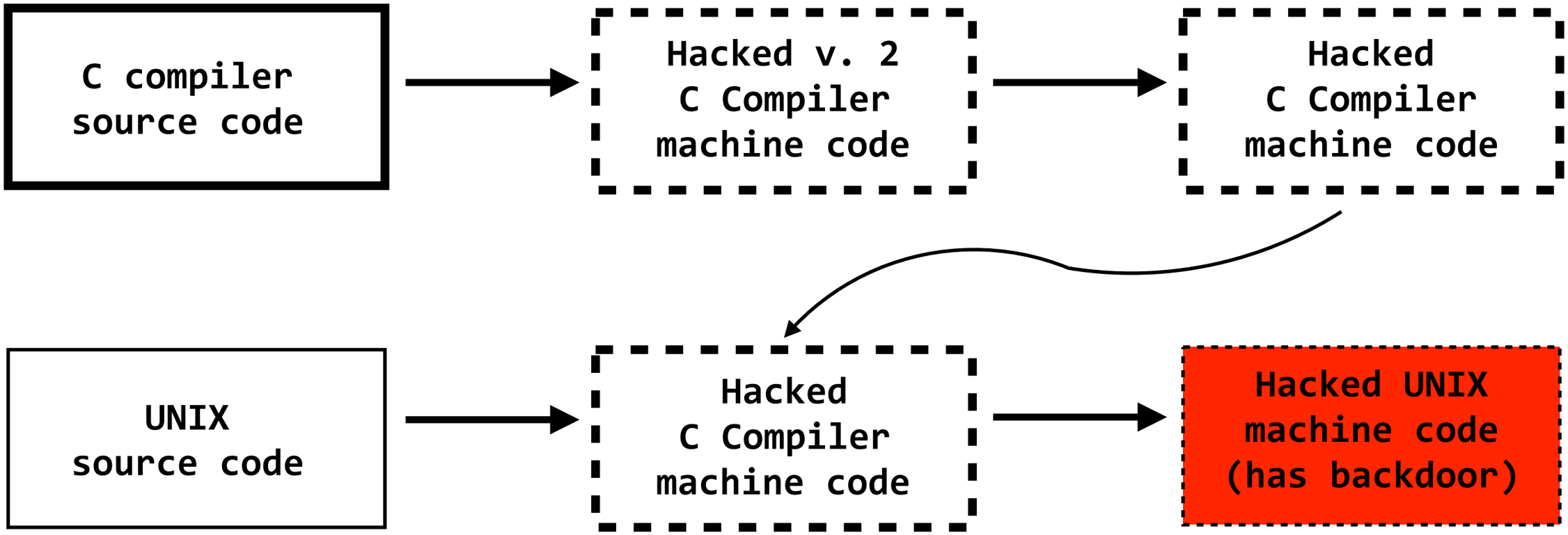
what if i just lie, and tell you that the hacked C compiler was generated from the clean C compiler source? can you check?

Yes: by recompiling the compiler, and then compiling the UNIX source

compilers: can we trust them?



The hacked C compiler has code that *inserts* a backdoor into UNIX *and* code to insert backdoor-inserting code into C compilers



REFERENCES

1. Bobrow, D.G., Burchfiel, J.D., Murphy, D.L., and Tomlinson, R.S. TENEX, a paged time-sharing system for the PDP-10. *Commun. ACM 15*, 3 (Mar. 1972), 135–143.
2. Kernighan, B.W., and Ritchie, D.M. *The C Programming Language*. Prentice-Hall, Englewood Cliffs, N.J., 1978.
3. Ritchie, D.M., and Thompson, K. The UNIX time-sharing system. *Commun. ACM 17*, (July 1974), 365–375.
4. Unknown Air Force Document.

Karger, P.A., and Schell, R.R.
Multics Security Evaluation: Vulnerability Analysis
ESD-TR-74-193, Vol II, June 1974, page 52

where to go from here

6.828 - Operating Systems
6.829 - Computer Networks
6.830/6.814 - Database Systems

6.858 - Computer Systems Security
6.857 - Network and Computer Security
6.875 - Cryptography and Cryptanalysis

more systems ↓ **more math**

6.824 - Distributed Systems ← **a natural follow-up to 6.033**
6.826 - Principles of Computer Systems
6.852 - Distributed Algorithms

6.903 - Intellectual Property
6.904 - Ethics for Engineers