Breaking the Links: Exploiting the Linker

"I thought this was common knowledge (it's known amongst my circle Unix admins of my generation, mid-40's +), but it doesn't appear to be well documented any more." - Anonymous

Who am I?

Tim Brown

- pentester at Portcullis for 6 years
- 16 years working with *NIX
- contributor to a variety of F/OSS projects
- previous research on KDE and Vista

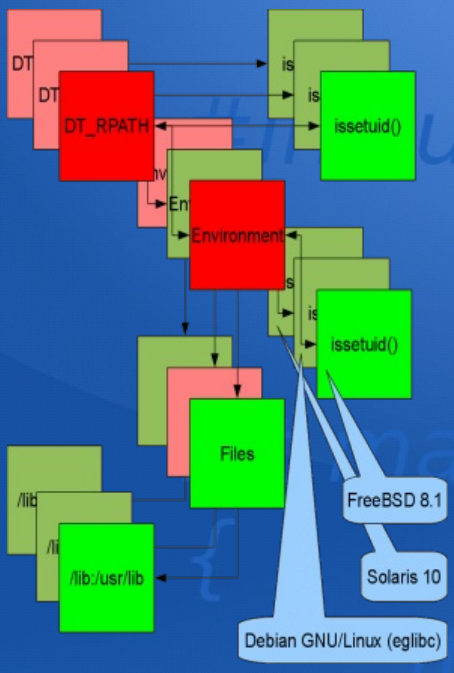
Introduction

- What is the linker?
- The linker attack surface
- Real world exploitation
- Auditing shell scripts, binaries and source

What is the linker?

- The link editor
- The runtime linker

The linker attack surface

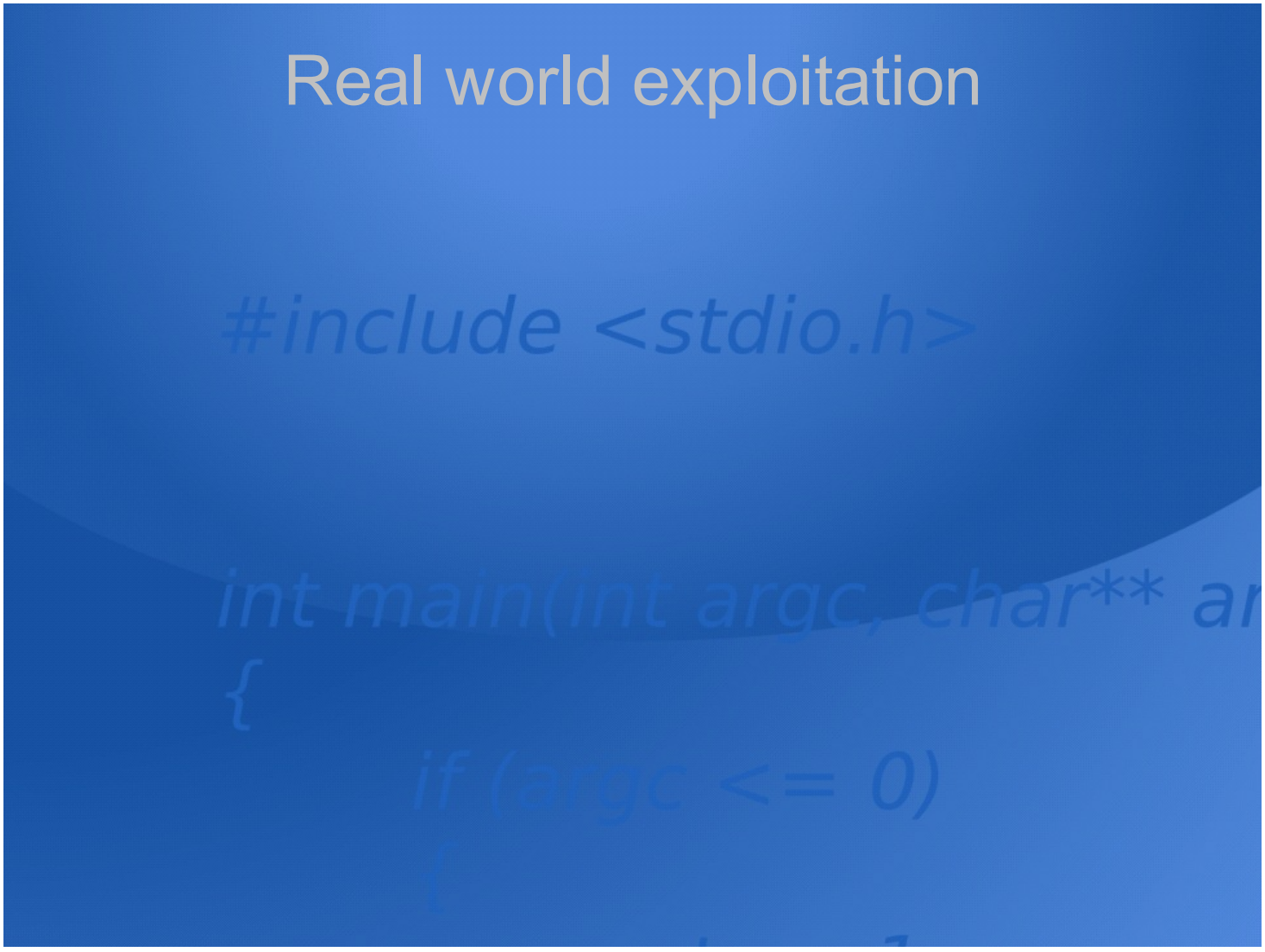


- The link editor
- The runtime linker

Environment

- Solaris 10 supports 23 environment variables
- Debian GNU/Linux (eglibc) supports 17 environment variables
- FreeBSD 8.1 supports 13 environment variables
- Only 3 common to Solaris 10, Debian GNU/Linux (eglibc) and FreeBSD 8.1

Real world exploitation



Auditing scripts

- Unsafe concatenation
  - touch ./libc.so.6 && sudo ... (@kees_cook mentioned this technique on Twitter)
  - grep "LD_" ...

Auditing binaries

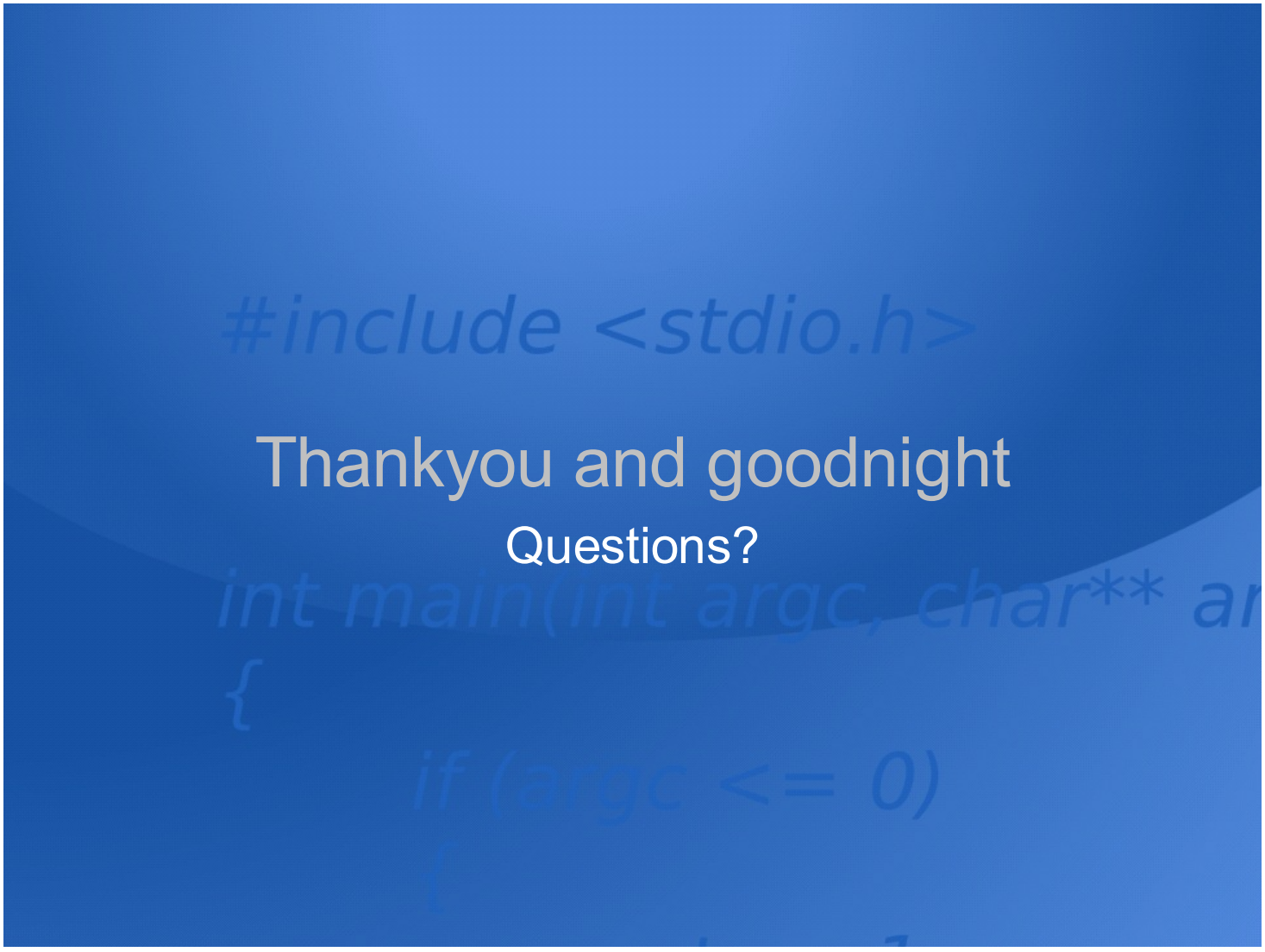
- DT_RPATH and DT_RUNPATH
  - objdump -x ..., readelf -a ..., scanelf (from PaX) and elfdump (from Sun)
- +s binaries that untaint without cleaning environment variables

How about source?

- Build scripts honouring LD_RUN_PATH
- Compiler and linker flags
  - gcc -Wl,-R,...
  - ld [-rpath|-rpath-link]=...
  - ld -R ...

Further research

- Other linkers
- Statically linked binaries
- Libraries depending on libraries
- Real world consequences
- Single stepping SetUID processes
- Hardening future linkers
- Linker scripts


#include <stdio.h>
Thankyou and goodnight
Questions?
int main(int argc, char** ar
{
if (argc <= 0)

Contacting me

timb@nth-dimension.org.uk

@timb_machine