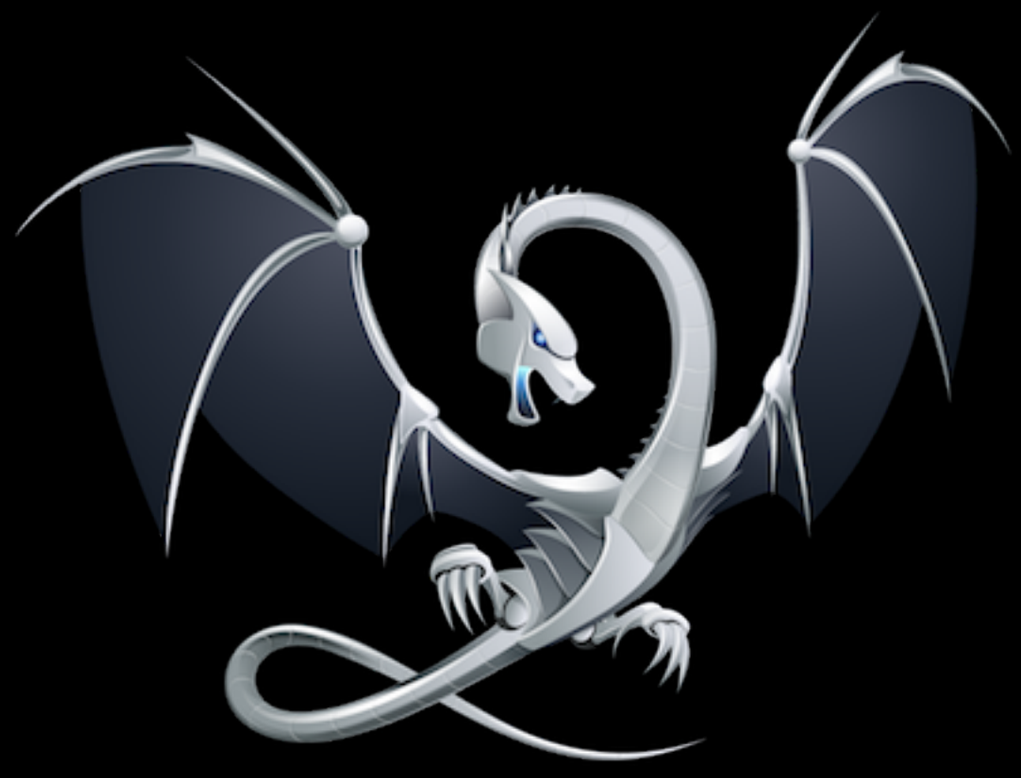


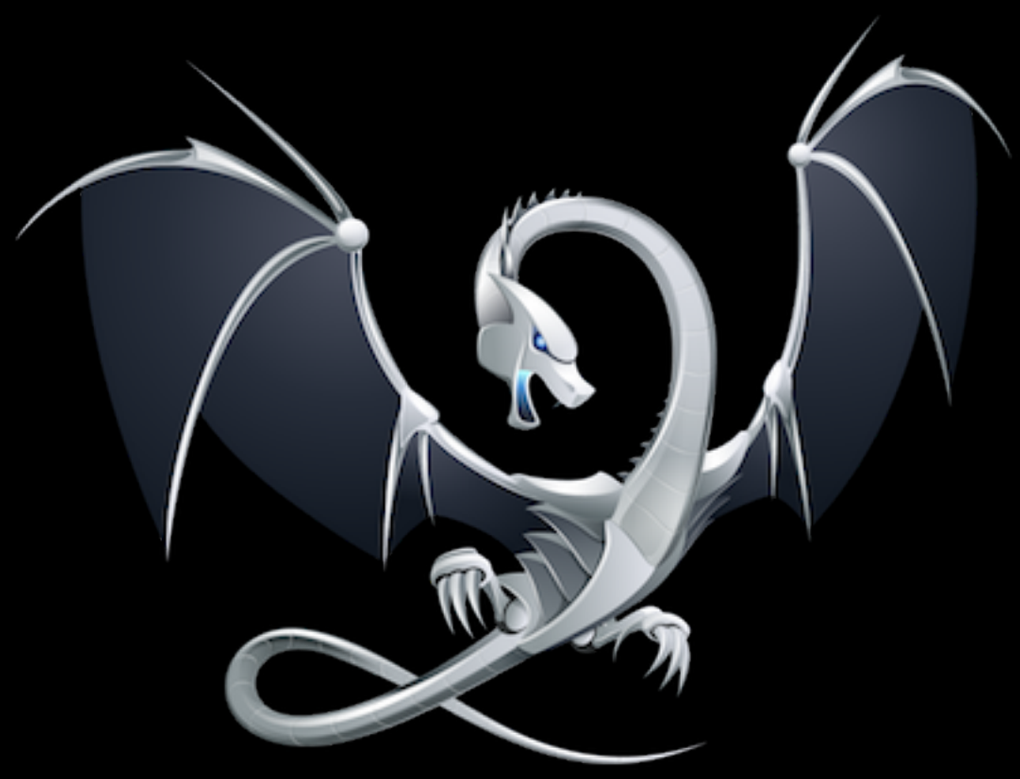
Integrating LLVM into FreeBSD

FOSDEM 2012

Brooks Davis

The LLVM Project is a collection of modular and reusable compiler and toolchain technologies.

-- LLVM.org



GNU Toolchain

- GCC
- Binutils
- GDB

2006

GNU Toolchain

- GCC
- Binutils
- GDB
- BSDElfTools

2006

2007

GPLv3

Free Software

Free as in Freedom

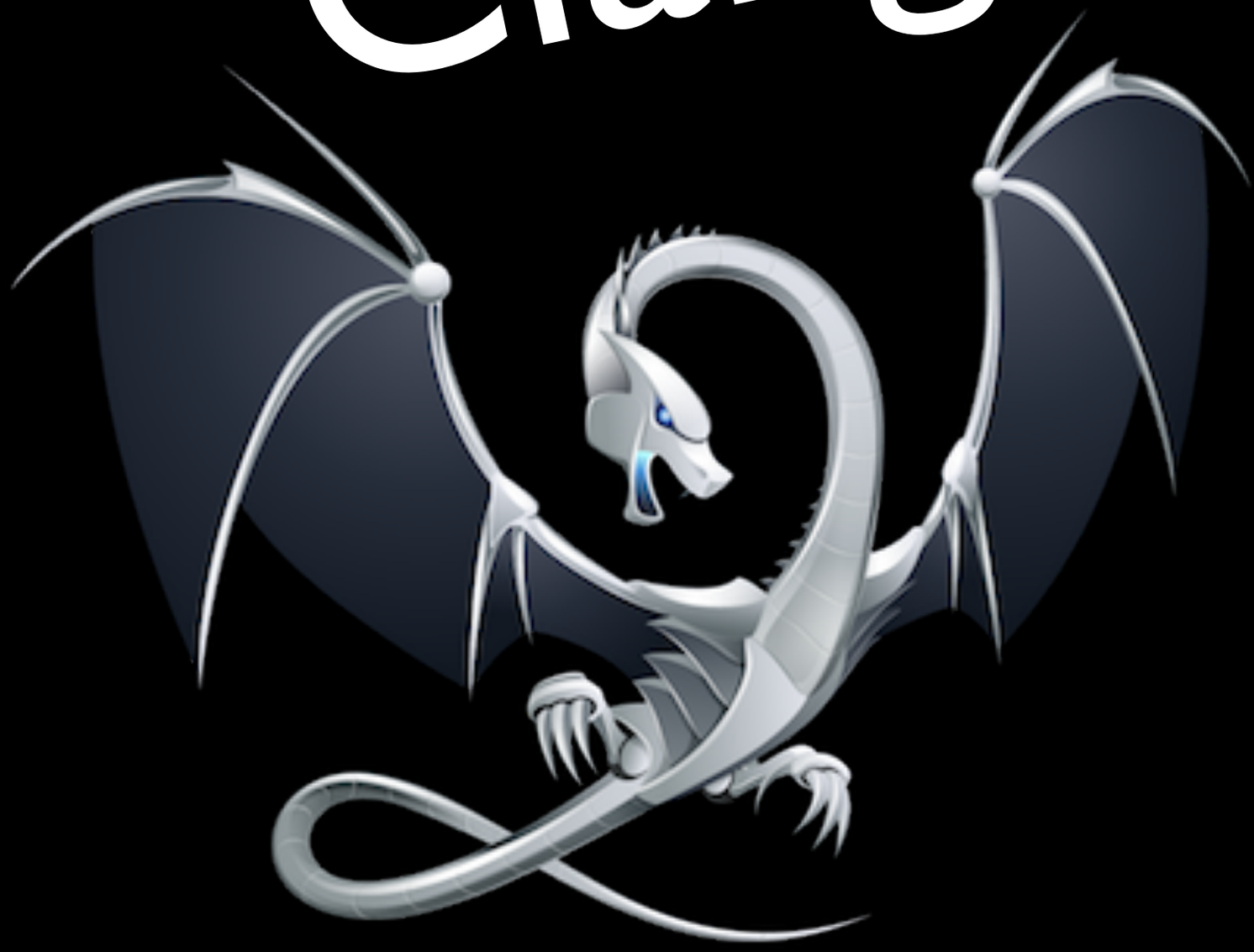
2007

Toolchain Freeze

2007

2007

Clang



2007

2008 & 2009

The Times

2 Feb 2010

Clang Self-Hosting

Cupertino -- Today, Clang completed its first complete self-host! We built all of LLVM and Clang with Clang (over 550k lines of C ++ code). The resulting binaries passed all of Clang and LLVM's regression test suites, and the Clang-built Clang could then build all of LLVM and Clang again. The third-stage Clang was also fully-functional, completing the bootstrap.

Congratulations to all of the Clang developers on this amazing achievement!

BSDCan

BSD Toolchain Summit

2010

FreeBSD

2010

FreeBSD

Clang

2010

2011

FOSDEM BSD Licensed Toolchain Summit

2011

FOSDEM BSD Licensed Toolchain Summit

libc++ ported

2011

FOSDEM BSD Licensed Toolchain Summit

libc++ ported

LLDB port in-progress

2011

FreeBSD 9.0

FreeBSD 9.0

- GCC remains default

FreeBSD 9.0

- GCC remains default
- Clang ~3.0 in base

FreeBSD 9.0

- GCC remains default
- Clang ~3.0 in base
- Much of libgcc replaced

FreeBSD 9.0

- GCC remains default
- Clang ~3.0 in base
- Much of libgcc replaced
- Some BSD ELF Tools

WIPs

WIPs

- libc++ import

WIPs

- libc++ import
- LLDB port

WIPs

- libc++ import
- LLDB port
- switchable ports compiler

TODO

TODO

- Finish libgcc replacement

TODO

- Finish libgcc replacement
- External toolchain support

TODO

- Finish libgcc replacement
- External toolchain support
- Switch for base compiler

TODO

- Finish libgcc replacement
- External toolchain support
- Switch for base compiler
- Clean cross build support

Gaps

Gaps

Linker

Gaps

Linker

ARM

Gaps

Linker

ARM

MIPS

Gaps

Linker

ARM

MIPS

Sparc64

Clang/LLVM specific features?

Clang/LLVM specific features?

When?

Clang/LLVM specific features?

When?

What kind?

Temporally Enhanced Security Logic Assertions (TESLA)

Temporally Enhanced Security Logic Assertions (TESLA)

- Represent assertions as temporal logic or automata

Temporally Enhanced Security Logic Assertions (TESLA)

- Represent assertions as temporal logic or automata
- Assertions are tested on experienced paths at run-time

Temporally Enhanced Security Logic Assertions (TESLA)

- Represent assertions as temporal logic or automata
- Assertions are tested on experienced paths at run-time
- On failure: panic(), stack trace, DTrace events

Berkeley Packet Filter JIT

Berkeley Packet Filter JIT

- Existing i386 and amd64 JIT from WinPcap

Berkeley Packet Filter JIT

- Existing i386 and amd64 JIT from WinPcap
- Pros: Lightweight, simple

Berkeley Packet Filter JIT

- Existing i386 and amd64 JIT from WinPcap
- Pros: Lightweight, simple
- Cons: Hardcoded, unoptimized

Berkeley Packet Filter JIT

- Existing i386 and amd64 JIT from WinPcap
  - Pros: Lightweight, simple
  - Cons: Hardcoded, unoptimized
- LLVM would fix those issues

Firewalls

Firewalls

- Rulesets are DSLs

Firewalls

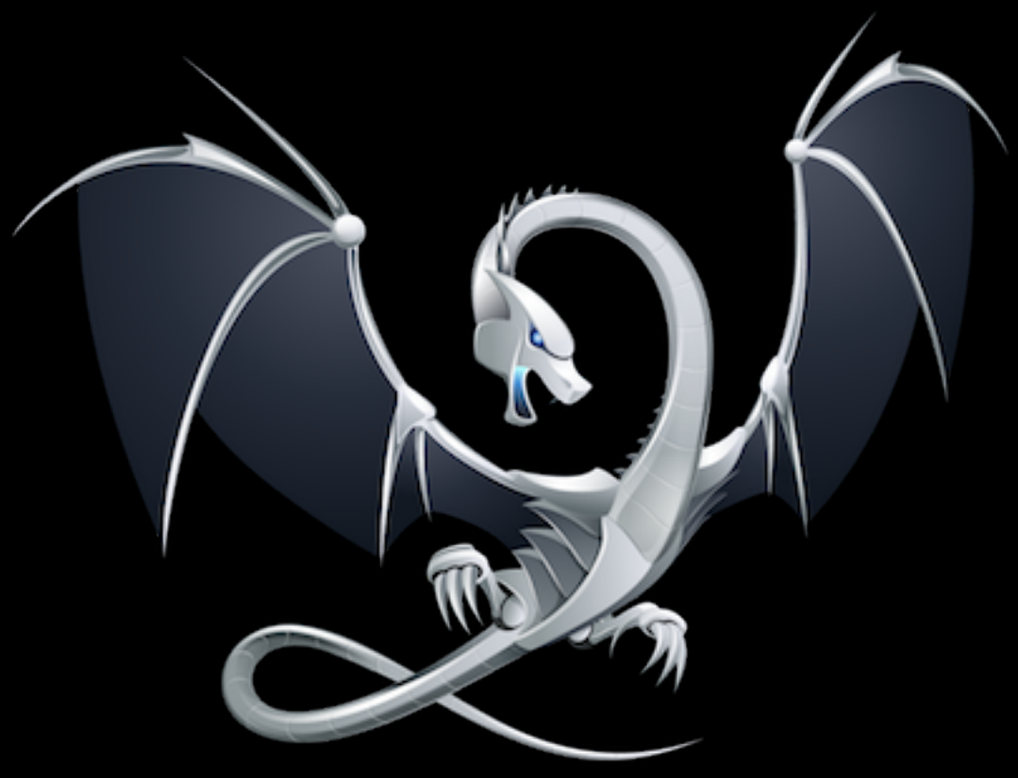
- Rulesets are DSLs
- Often use bytecode

Firewalls

- Rulesets are DSLs
- Often use bytecode
- Too many

Firewalls

- Rulesets are DSLs
- Often use bytecode
- Too many



FreeBSD and LLVM
a powerful combination!