Ditch the Rust Rewrite!

A tale on the creation of libsyslog
and libsyslog-sys

`whoami`

Martin "|cos|" Samuelsson

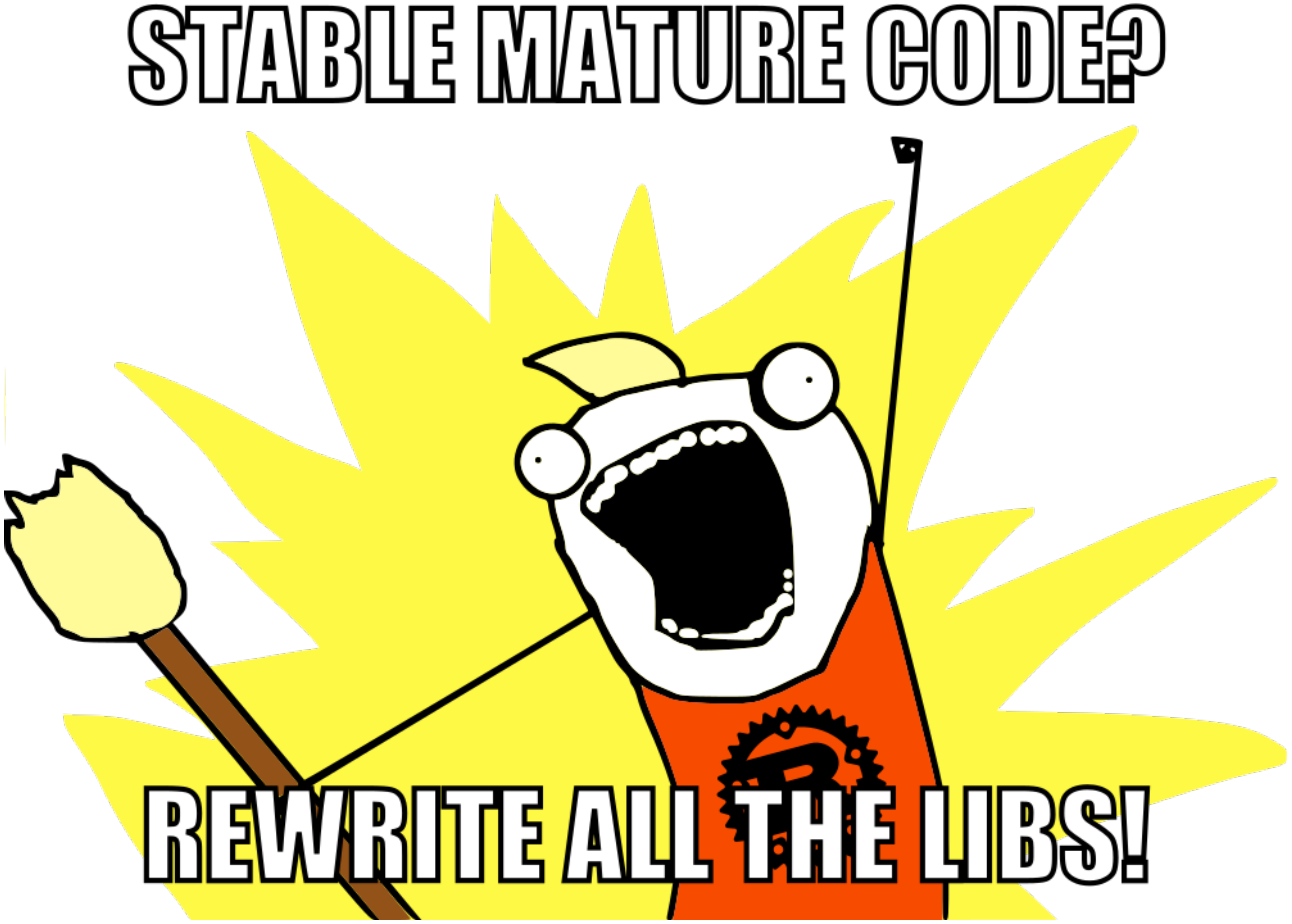
Self-Employed Electrical Engineer
doing Software Development

Residing in Malmø since 2008

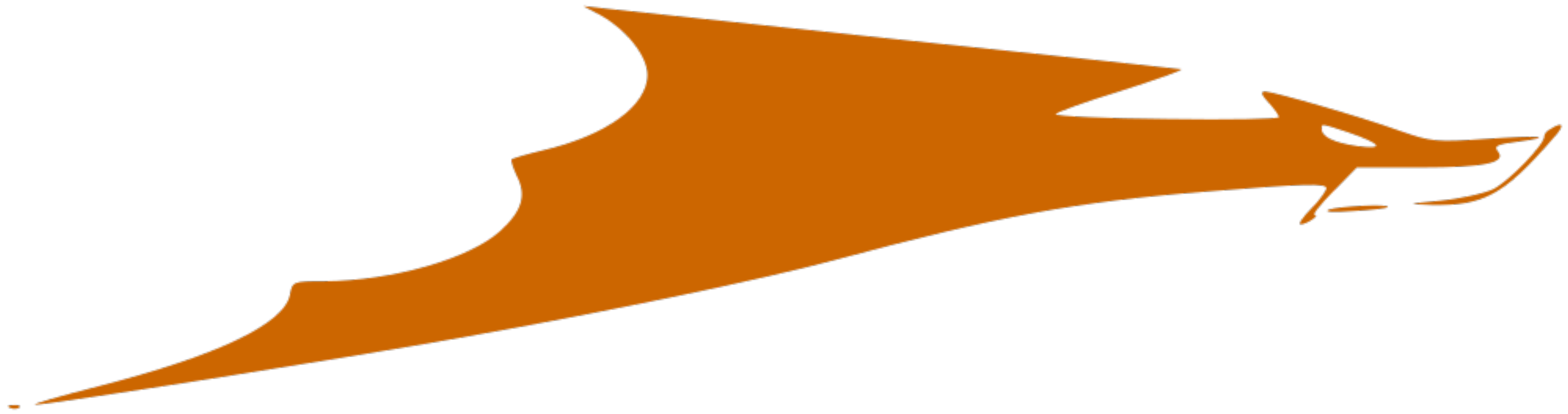
syslog

STABLE MATURE CODE?

syslog

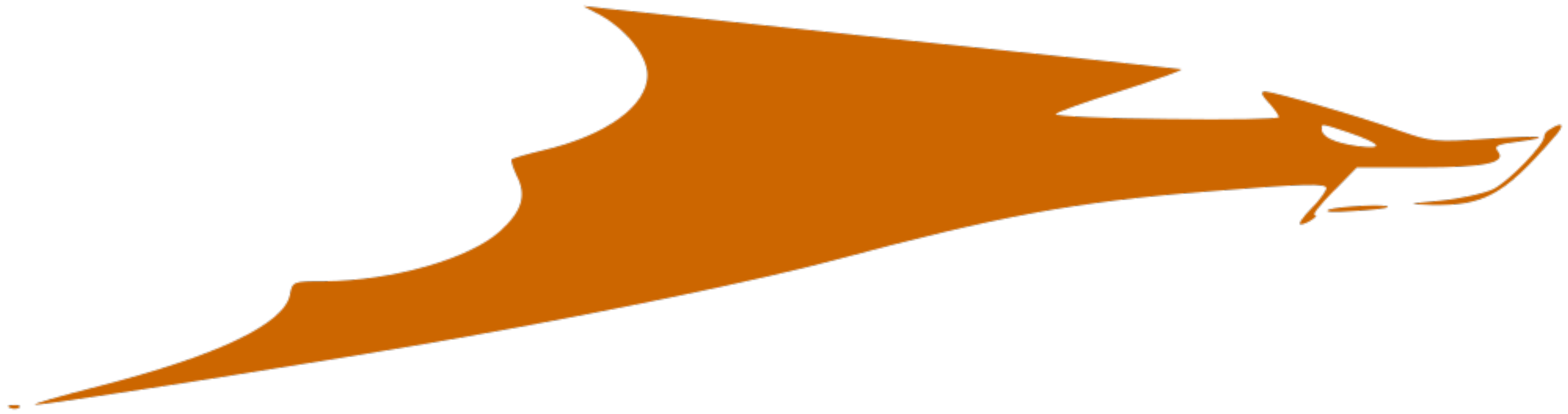

STABLE MATURE CODE?
REWRITE ALL THE LIBS!

```
% cargo search syslog
syslog = "6.0.1"                           # Send log mes
log4rs-syslog = "3.0.3"                    # Syslog appen
super_speedy_syslog_searcher = "0.5.58"    # Speedily sea
syslog-rs = "0.4.3"                        # A native Rus
syslogio = "0.2.1"                         # Command line
flexi_syslog = "0.5.2"                     # A syslog wri
syslog_loose = "0.18.0"                    # A loose pars
syslog-tracing = "0.1.0"                   # syslog backe
fastly-api = "1.2.0"                       # Fastly API c
slog-syslog = "0.13.0"                     # Syslog drain
... and 72 crates more (use --limit N to see more)
```



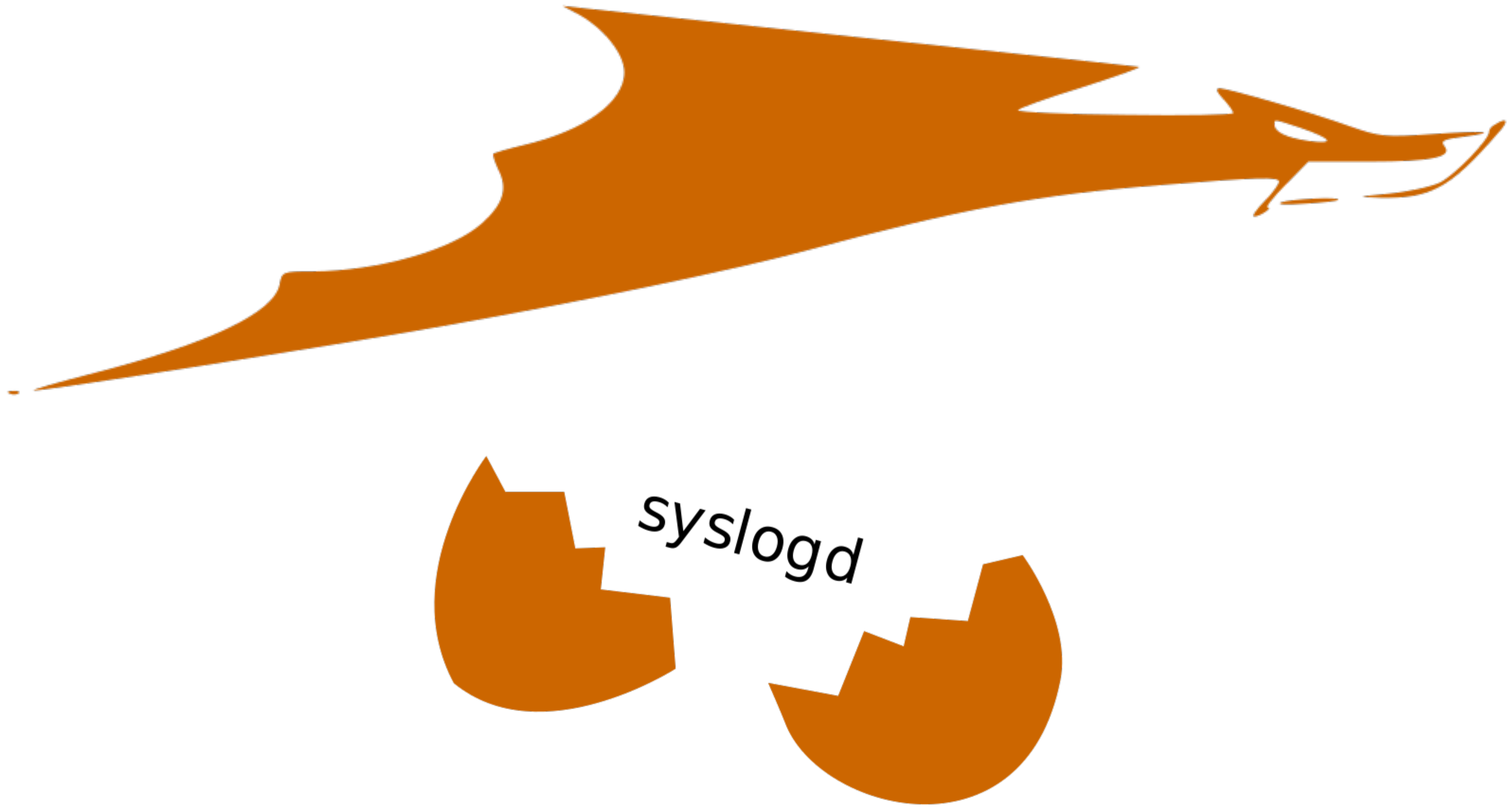
Eric Paul Allman

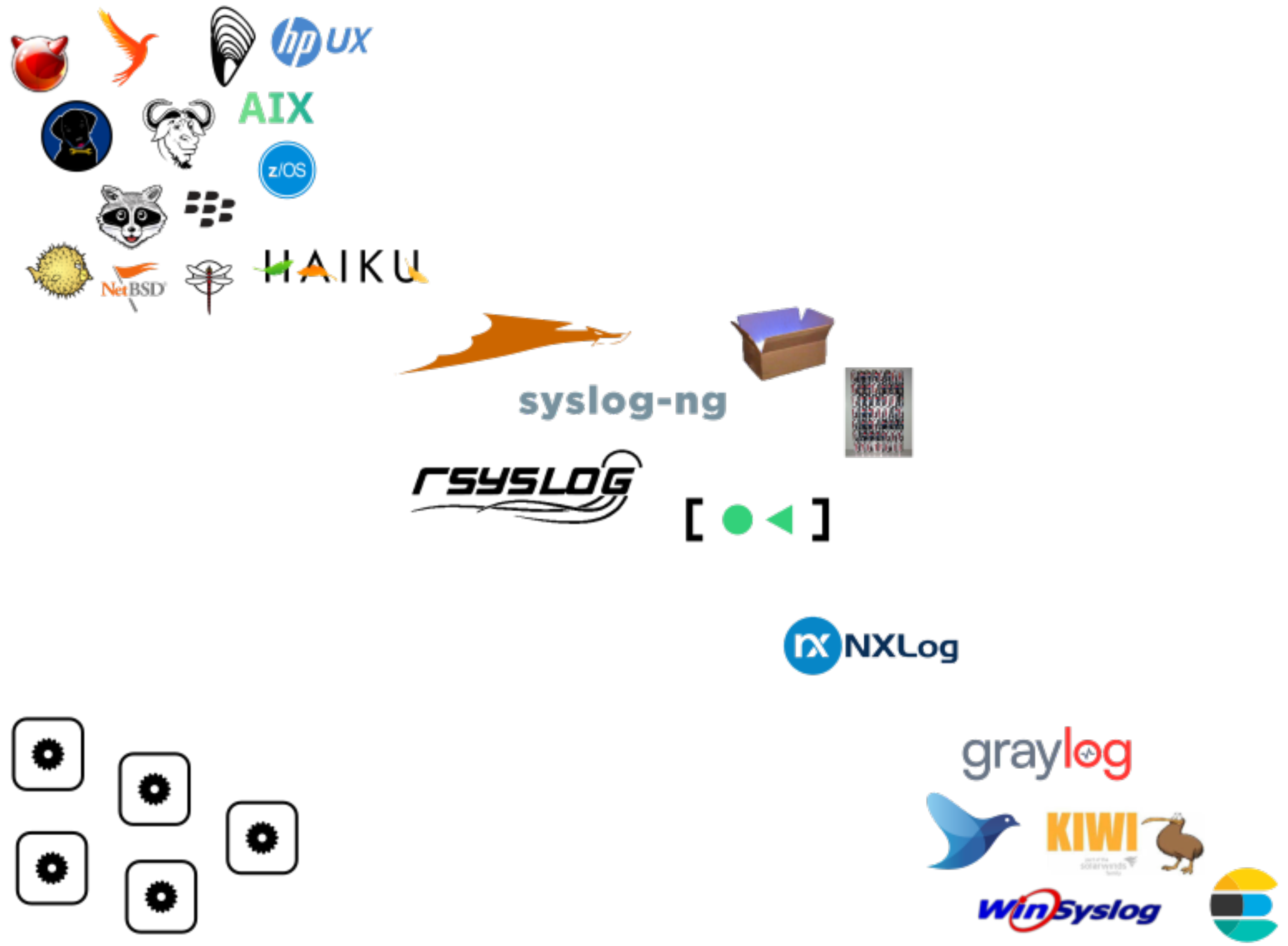
Author of sendmail

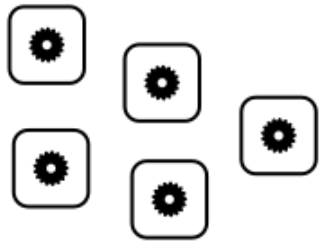


Eric Paul Allman

Author of sendmail




hp UX
AIX
z/OS
NetBSD
HAIKU
syslog-ng
rsyslog
NXLog
graylog
KIWI
WinSyslog



NXLog

graylog

KIWI

WinSyslog

HAIKU

syslog-ng

rsyslog

NXLog

hp UX
AIX
z/OS
NetBSD
HAIKU

C'mon!

15 slides already...

and no Rust code yet?

The log Crate

A logging facade provides a single logging API that abstracts over the actual logging implementation.

```
error!()
warn!()
info!()
debug!()
trace!()
```

https://lib.rs/crates/log

log::Log

Desired Goal

Writing a Rust program which logs messages to a local syslog daemon.

Note: Logging to a remote server is not necessarily desired.
Quirk: Must work under illumos.

Desired Goal

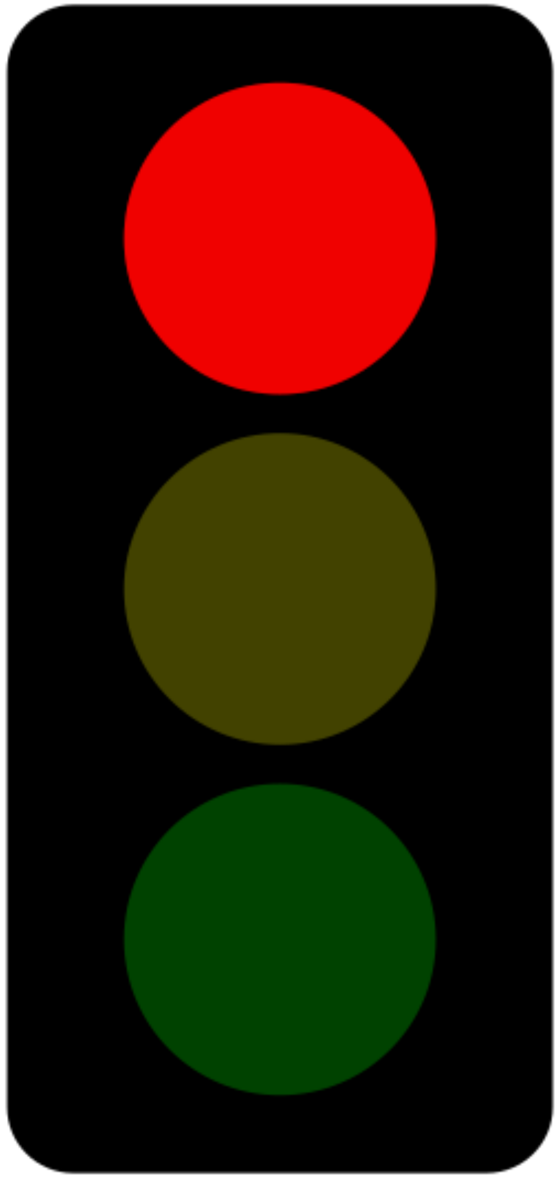
Writing a Rust program which logs messages to a local syslog daemon.

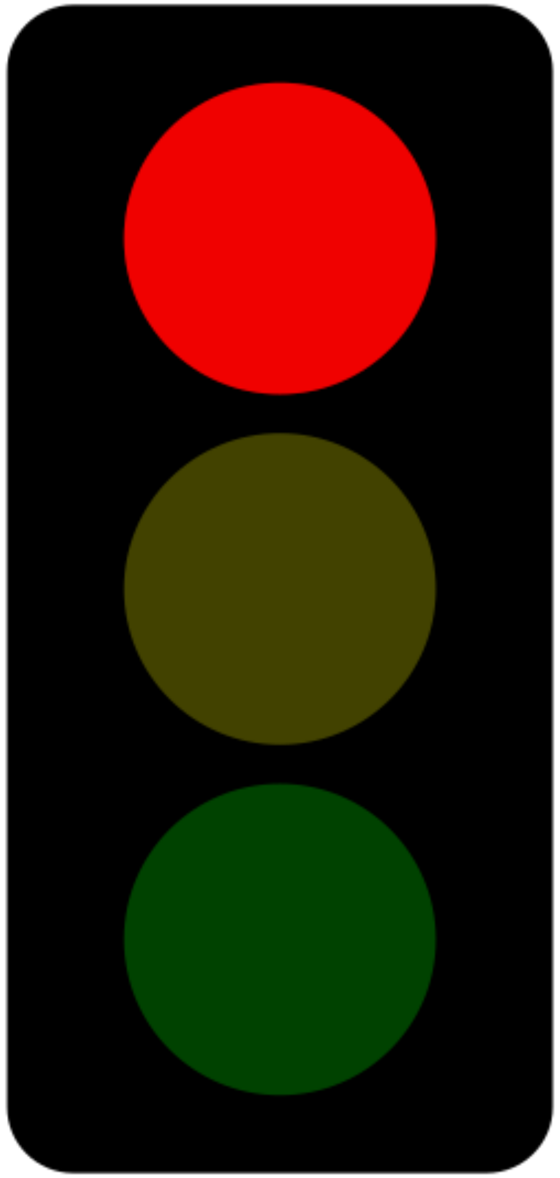
Note: Logging to a remote server is not necessarily desired.
Quirk: Must work under illumos.

```
use log::info;

fn main() {
    // some_syslog_crate::initialize_logging();

    info!("Hello, syslog!");
}
```

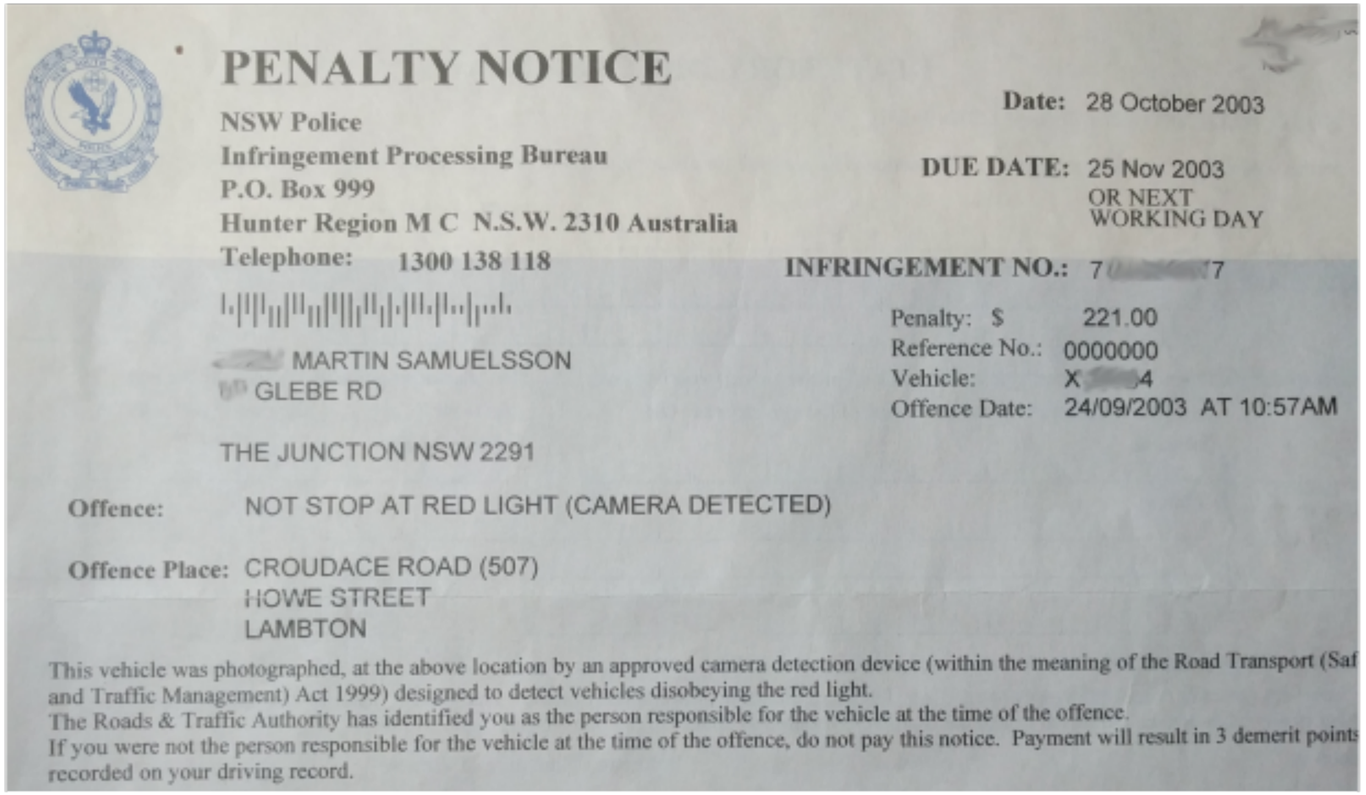
"Understand the interfaces which you are coding to!"

Theo de Raadt, founder of OpenBSD and OpenSSH





Stop, unless having special permissions to break the rules.



PENALTY NOTICE

NSW Police
Infringement Processing Bureau
P.O. Box 999
Hunter Region M C N.S.W. 2310 Australia
Telephone: 1300 138 118

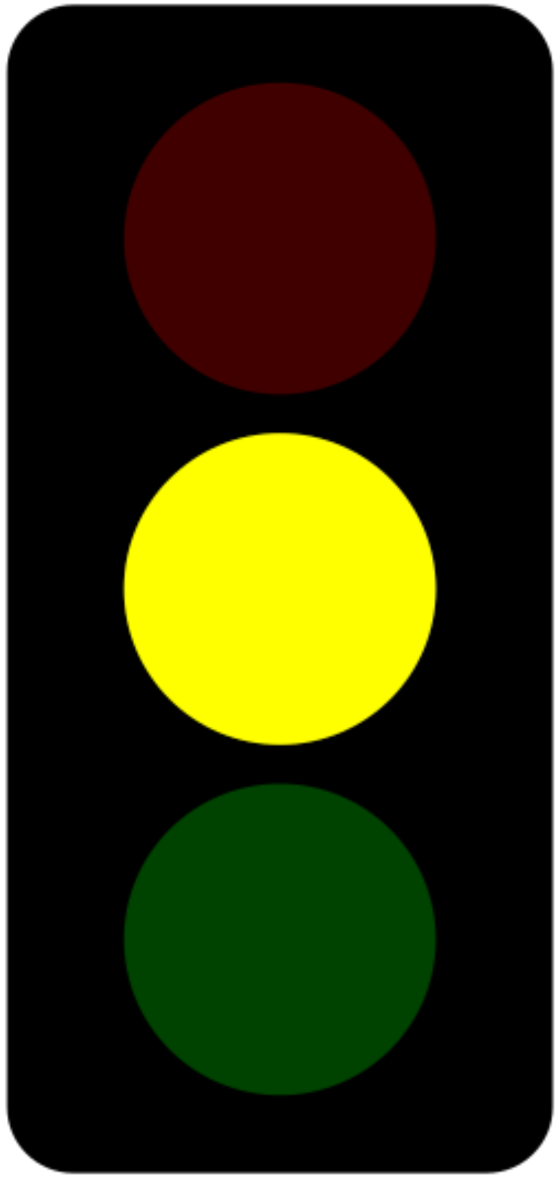
Date:	28 October 2003
DUE DATE:	25 Nov 2003 OR NEXT WORKING DAY
INFRINGEMENT NO.:	7[redacted]7
Penalty: $	221.00
Reference No.:	0000000
Vehicle:	X[redacted]4
Offence Date:	24/09/2003 AT 10:57AM

[redacted] MARTIN SAMUELSSON
[redacted] GLEBE RD

THE JUNCTION NSW 2291

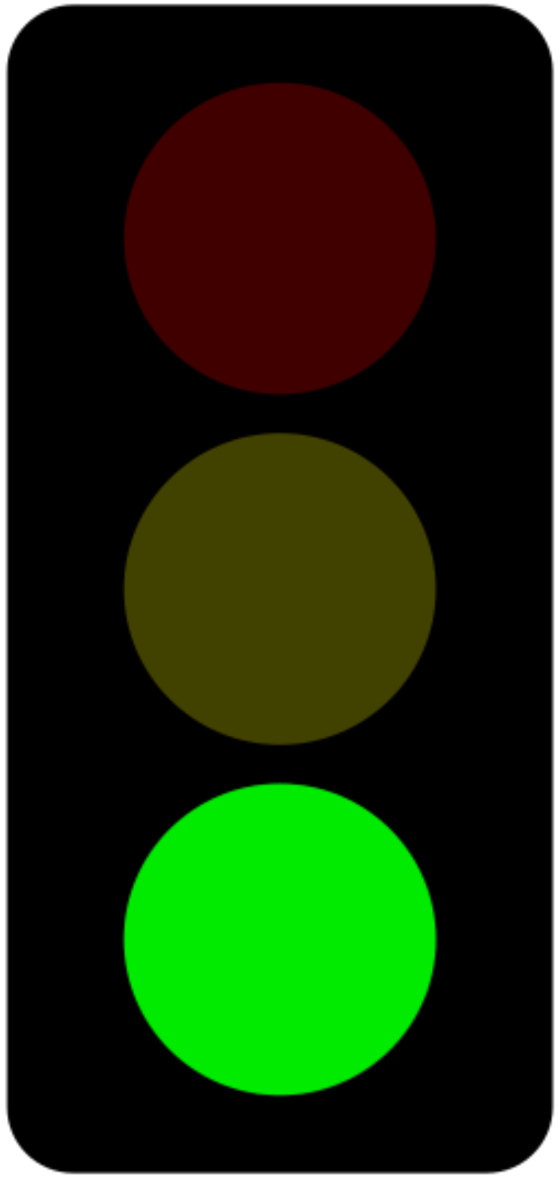
Offence:	NOT STOP AT RED LIGHT (CAMERA DETECTED)
Offence Place:	CROUDACE ROAD (507) HOWE STREET LAMBTON

This vehicle was photographed, at the above location by an approved camera detection device (within the meaning of the Road Transport (Saf
and Traffic Management) Act 1999) designed to detect vehicles disobeying the red light.
The Roads & Traffic Authority has identified you as the person responsible for the vehicle at the time of the offence.
If you were not the person responsible for the vehicle at the time of the offence, do not pay this notice. Payment will result in 3 demerit points
recorded on your driving record.



Stop, unless having special permissions to break the rules.

Allowed, but surely we should prefer to avoid yellow. Right?



Stop, unless having special permissions to break the rules.

Allowed, but surely we should prefer to avoid yellow. Right?

Green makes us happy! Lets hope to see some green soon.

Interface Description Documents

~~IETF RFC 3164 - The BSD syslog Protocol~~ (obsoleted by RFC 5424)

IETF RFC 5424 - The Syslog Protocol

IETF RFC 5425 - Transport Layer Security Mapping for Syslog

IETF RFC 5426 - Transmission of Syslog Messages over UDP

IEEE Std 1003.1-2017 (POSIX®)

Interface Description Documents

- ~~IETF RFC 3164 - The BSD syslog Protocol~~ (obsoleted by RFC 5424)
- IETF RFC 5424 - The Syslog Protocol
- IETF RFC 5425 - Transport Layer Security Mapping for Syslog
- IETF RFC 5426 - Transmission of Syslog Messages over UDP
- IEEE Std 1003.1-2017 (POSIX®)

Interface Description Documents

- 🟡 ~~IETF RFC 3164 - The BSD syslog Protocol~~ (obsoleted by RFC 5424)
- 🟡 IETF RFC 5424 - The Syslog Protocol
- 🟡 IETF RFC 5425 - Transport Layer Security Mapping for Syslog
- 🟡 IETF RFC 5426 - Transmission of Syslog Messages over UDP
- 🟢 IEEE Std 1003.1-2017 (POSIX®)

A Wiktionary Definition

homonym

A word that both sounds and is spelled the same as another word.

Term Disambiguation

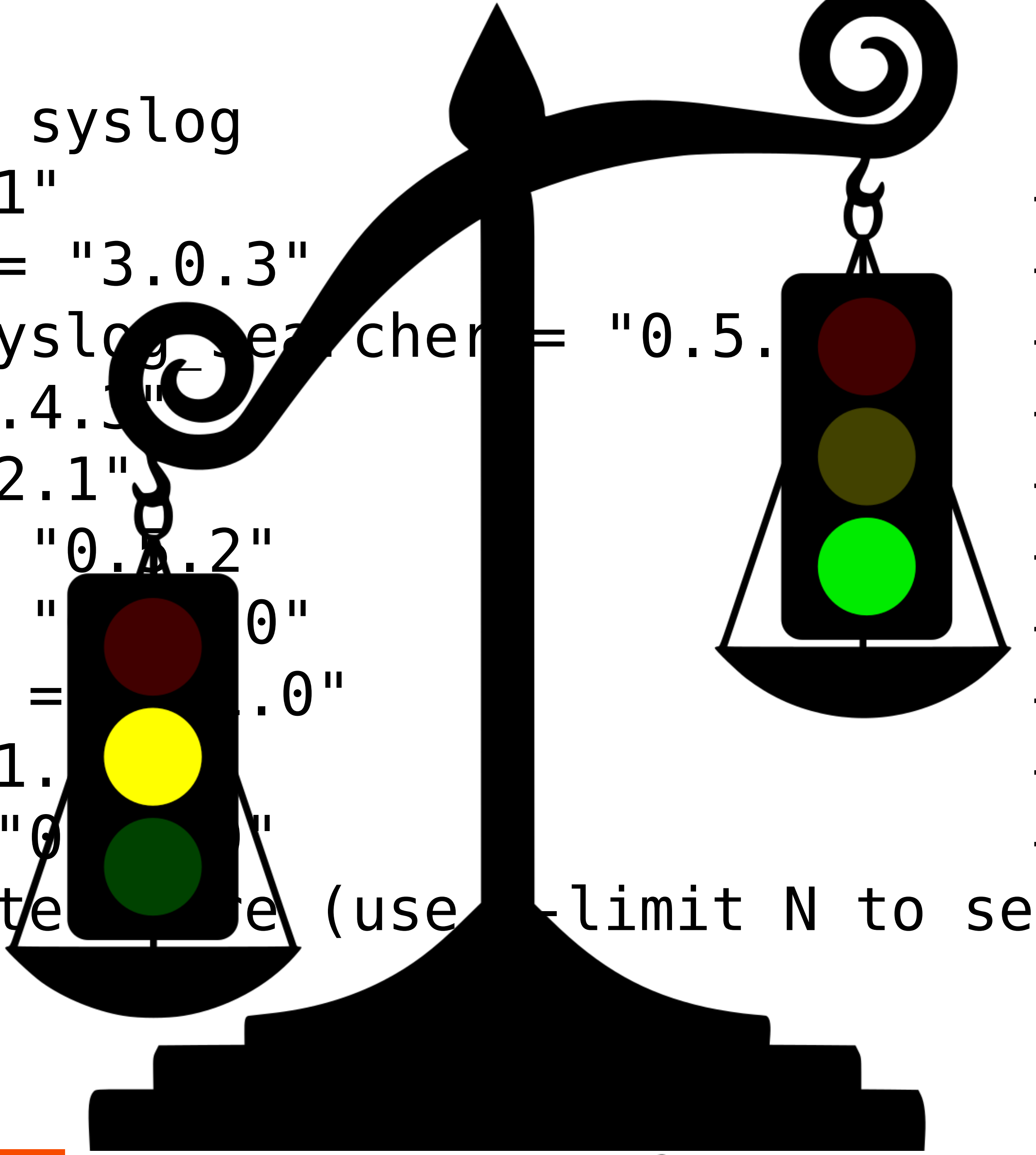
syslog

1. A protocol used for sending messages between network hosts. Described by IETF in RFC5424 and related documents.

2. The POSIX® System Interface for error logging.

3. A linux kernel system call with an unfortunate name. (Lets pretend this one does not exist.)

```
% cargo search syslog
syslog = "6.0.1"                                       # Send log mes
log4rs-syslog = "3.0.3"                                # Syslog appen
super_speedy_syslog_searcher = "0.5.                   # Speedily sea
syslog-rs = "0.4.                                      # A native Rus
syslogio = "0.2.1"                                     # Command line
flexi_syslog = "0. .2"                                 # A syslog wri
syslog_loose = "     0"                                # A loose pars
syslog-tracing =     .0"                               # syslog backe
fastly-api = "1.                                       # Fastly API c
slog-syslog = "0     "                                 # Syslog drain
... and 72 crate    e (use    -limit N to see more)
```



Crate: syslog-rs

Using syslog-rs Crate

```
% cargo init
     Created binary (application) package
% cargo add --no-default-features syslog-rs
    Updating crates.io index
      Adding syslog-rs v0.5.0 to dependencies.
             Features:
             - use_async
             - use_sync
             - use_sync_queue
```

Using syslog-rs Crate

```rust
fn main() {
}
```

Using syslog-rs Crate

```
✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂
42 | /         {
43 | |             #[cfg(any(
44 | |                 target_os = "freebsd",
45 | |                 target_os = "dragonfly",
...  |
53 | |             unsafe{ program_invocation_name }
54 | |         };
   | |_________- this empty block is missing a tail expression
55 |
56 |       let temp = unsafe {CStr::from_ptr(pn)};
   |                          -------------- ^^ expected `*const i8`, found `()`
   |                          |
   |                          arguments to this function are incorrect
   |
   = note: expected raw pointer `*const i8`
                found unit type `()`
✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂
error: could not compile `syslog-rs` due to 7 previous errors
```

Using syslog-rs Crate

In summary:

- API leaves a bit to be desired.
- Does not build on illumos.

Crate: syslog

Using syslog Crate

```
% cargo init
     Created binary (application) package
% cargo add log syslog
    Updating crates.io index
      Adding log v0.4.17 to dependencies
      Adding syslog-rs v6.1.0 to dependencies.
```

Using syslog Crate

```rust
use log::{info, set_boxed_logger};

fn main() {
    let formatter = syslog::Formatter3164 {
        facility: syslog::Facility::default(),
        hostname: None,
        process: "cph.rs".into(),
        pid: 0,
    };
    let logger = syslog::unix(formatter).unwrap();
    set_boxed_logger(Box::new(syslog::BasicLogger::new(logger)))
            .map(|()| log::set_max_level(log::LevelFilter::Info));
    info!("Hello, syslog crate");
}
```

Using syslog Crate

Run-time error:
"Socket operation on non-socket (os error 95)"

Using syslog Crate

Run-time error:
"Socket operation on non-socket (os error 95)"

Contemporary Linux:

```
% file /dev/log
/dev/log: symbolic link to /run/systemd/journal/dev-log

% file --dereference /dev/log
/dev/log: socket
```

Using syslog Crate

Run-time error:
"Socket operation on non-socket (os error 95)"

Contemporary Linux:

```
% file --dereference /dev/log
/dev/log: socket
```

illumos:

```
% file /dev/log
/dev/log:       character special (124/5)
```

Using syslog Crate

In summary:

- API is idiomatic Rust.
- Does not run on illumos.
  - Due to an invalid assumption on /dev/log.

Crate: syslog3

Using syslog3 Crate

In summary:

- The same syslog crate, in a legacy version with another name.
    - "Shim library re-exporting syslog::* from syslog version 3.0"

Crate: slog-syslog

Using slog-syslog Crate

In summary:

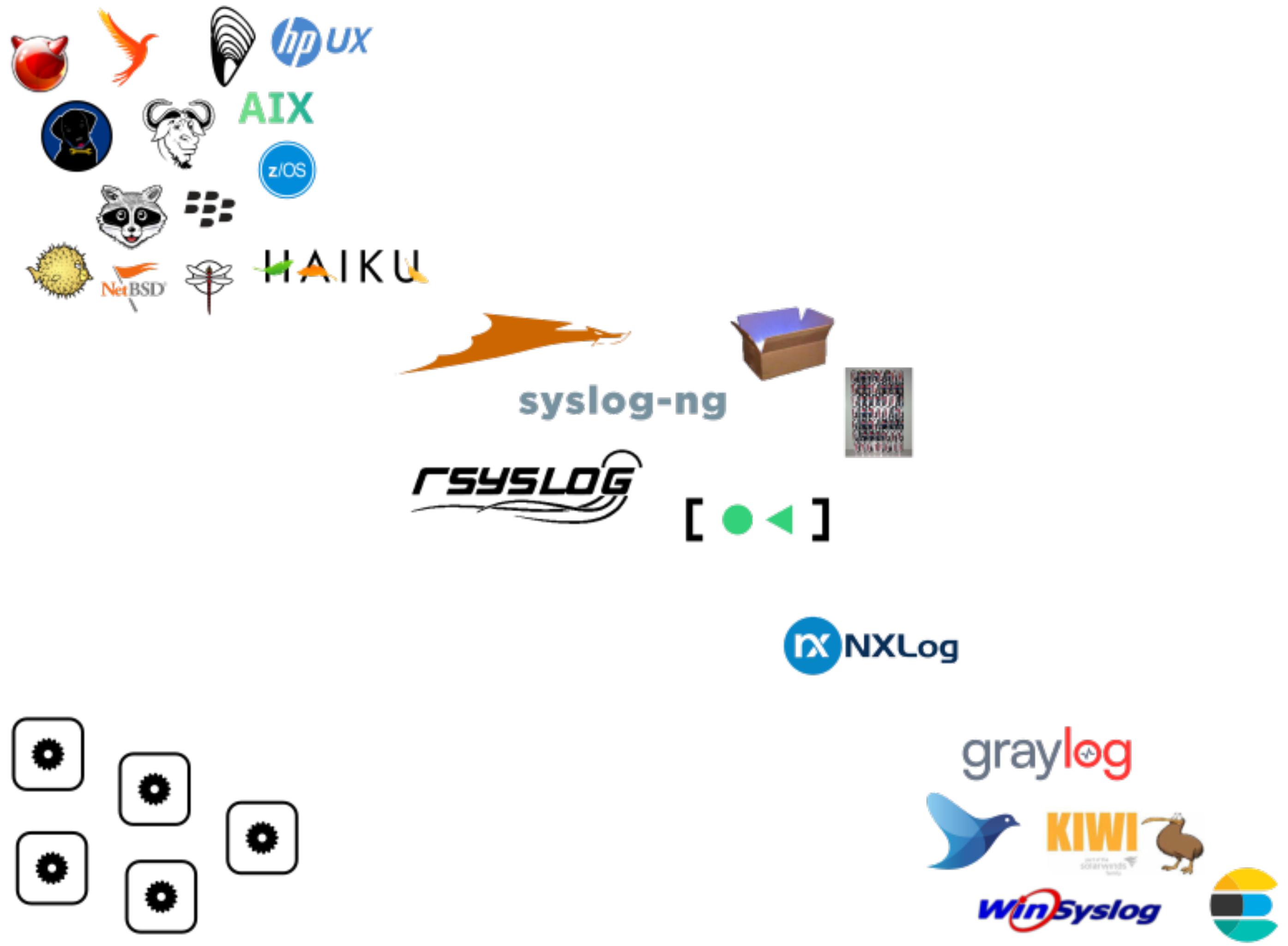
- API seems like idiomatic Rust
  - Yet slog-rs describes their crate as having a steep learning curve.
- Depends on the buggy syslog crate.

Reality Check!

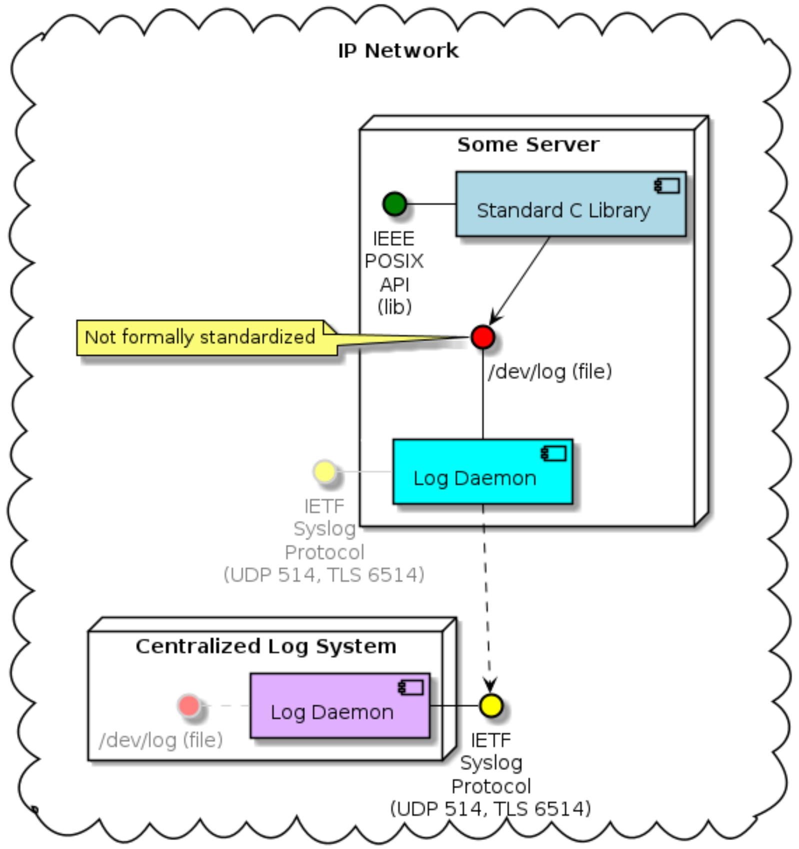
POSIX Syslog API

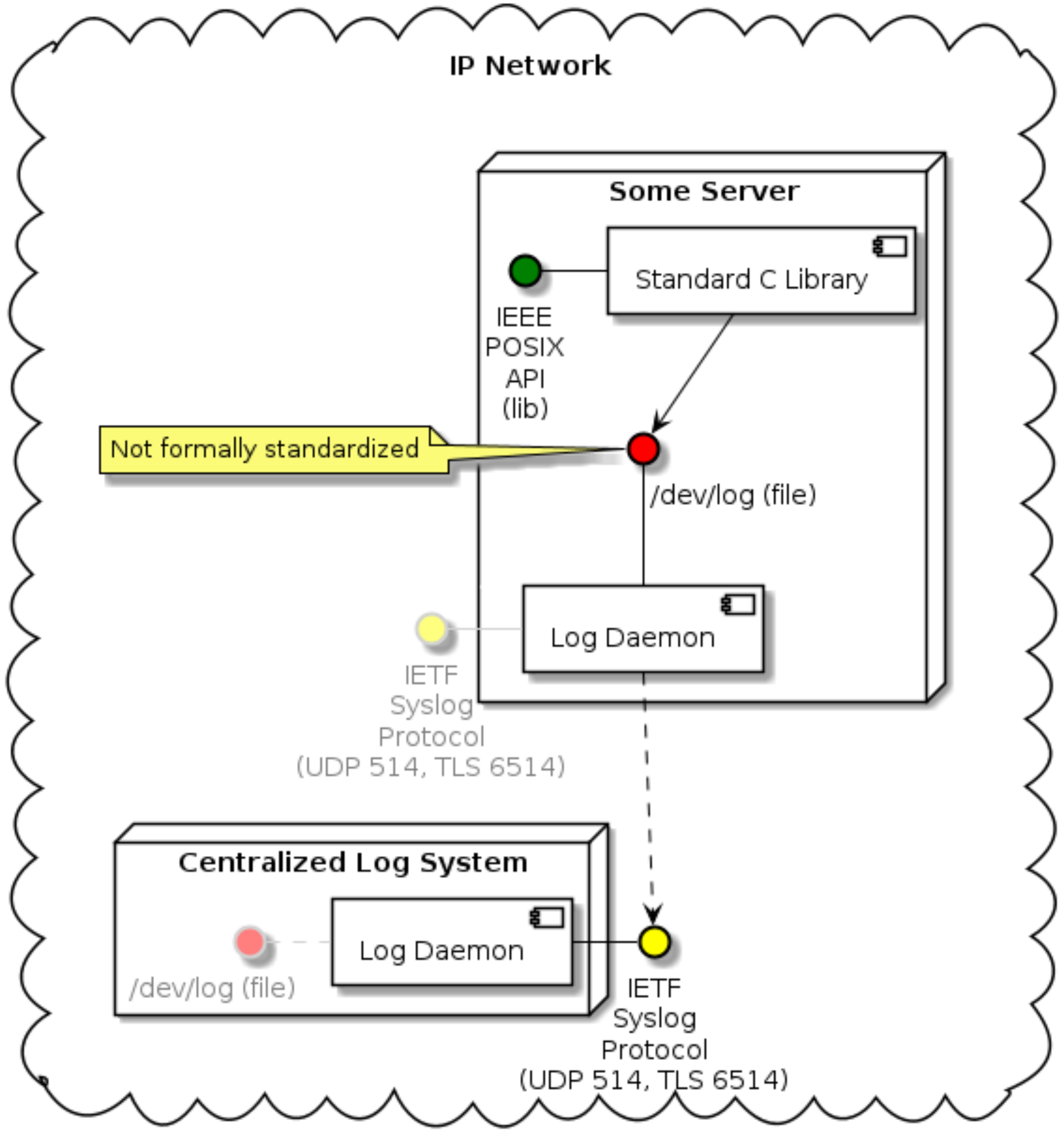
Very simple. Essentially three C functions:

openlog()
syslog()
closelog()


hp UX
AIX
z/OS
NetBSD
HAIKU
syslog-ng
rsyslog
NXLog
graylog
KIWI
WinSyslog

HP UX
AIX
z/OS
HAIKU
NetBSD
syslog-ng
rsyslog
NXLog
graylog
KIWI
WinSyslog


IP Network
Some Server
Standard C Library
IEEE POSIX API (lib)
Not formally standardized
/dev/log (file)
Log Daemon
IETF Syslog Protocol (UDP 514, TLS 6514)
Centralized Log System
Log Daemon
/dev/log (file)
IETF Syslog Protocol (UDP 514, TLS 6514)


IP Network
Some Server
Standard C Library
IEEE POSIX API (lib)
Not formally standardized
/dev/log (file)
Log Daemon
IETF Syslog Protocol (UDP 514, TLS 6514)
Centralized Log System
Log Daemon
/dev/log (file)
IETF Syslog Protocol (UDP 514, TLS 6514)

bindgen

Automatically generates Rust FFI bindings to C and C++ libraries

https://lib.rs/crates/bindgen

libsyslog-sys

Creating libsyslog-sys Crate

```
% cargo init --lib
     Created library package
% cargo add bindgen
    Updating crates.io index
      Adding syslog-rs v0.65.1 to dependencies.
             Features:
             + log
             + logging
✂         ✂         ✂         ✂         ✂         ✂         ✂
             - testing_only_libclang_5
             - testing_only_libclang_9
```

Creating libsyslog-sys Crate

src/lib.rs

```rust
include!(concat!(env!("OUT_DIR"), "/bindings.rs"));
```

wrapper.h

```c
#include <syslog.h>
```

Creating libsyslog-sys Crate

build.rs

```rust
use bindgen::{Builder, CargoCallbacks, MacroTypeVariation},
    std::{env,path::PathBuf}};
fn main() {
    let bindings = Builder::default()
        .header("wrapper.h")
        .parse_callbacks(Box::new(CargoCallbacks))
        .default_macro_constant_type(MacroTypeVariation::Signed)
        .generate().unwrap()
    let out_path = PathBuf::from(env::var("OUT_DIR").unwrap());
    bindings.write_to_file(out_path.join("bindings.rs")).unwrap()
}
```

libsyslog

Creating libsyslog Crate

```
% cargo init --lib
     Created library package
% cargo add bitflags libsyslog-sys log
    Updating crates.io index
      Adding bitflags v2.3.1 to dependencies.
✂         ✂         ✂         ✂         ✂         ✂         ✂
      Adding libsyslog-sys v0.1.0 to dependencies.
      Adding log v0.4.17 to dependencies.
             Features
✂         ✂         ✂         ✂         ✂         ✂         ✂
             - value-bag
```

Creating libsyslog Crate

src/lib.rs

```rust
mod builder;
mod facility;
mod logopt;
mod syslog;
pub use {
    builder::*,
    facility::*,
    logopt::*,
    syslog::*,
};
```

Creating libsyslog Crate

src/builder.rs

Creating libsyslog Crate

src/facility.rs

```
use libsyslog_sys::*;

pub enum Facility {
    Kern     = LOG_KERN     as isize,
    Mail     = LOG_MAIL     as isize,
    Daemon   = LOG_DAEMON   as isize,
    User     = LOG_USER     as isize,
✂        ✂        ✂        ✂        ✂        ✂
}

impl Default for Facility {
    fn default() -> Facility {
      Facility::User
    }
}
```

Creating libsyslog Crate

src/logopt.rs

```
use {bitflags::bitflags, libsyslog_sys::*,
    std::os::raw::c_int};

bitflags! {
    #[derive(Debug,Default)]
    pub struct Logopt: c_int {
        const Pid    = LOG_PID;
        const Cons   = LOG_CONS;
        const ODelay = LOG_ODELAY;
        const NDelay = LOG_NDELAY;
        const NoWait = LOG_NOWAIT;
        #[cfg(any(target_os="freebsd", target_os="netbsd"))]
✂        ✂        ✂        ✂        ✂        ✂        ✂
    }
}
```

Creating libsyslog Crate

src/syslog.rs
1/5

```
use {
✂         ✂         ✂         ✂         ✂         ✂         ✂         ✂
};

pub struct Syslog {
    pub(crate) facility: c_int,
    pub(crate) ident: CString,
    pub(crate) level: LevelFilter,
    pub(crate) logopt: c_int,
    pub(crate) module_levels: Vec<(String, LevelFilter)>,
}
```

Creating libsyslog Crate

src/syslog.rs
2/5

```
impl Syslog {
  pub fn builder() -> SyslogBuilder { SyslogBuilder::default() }
  pub fn init(mut self) -> Result<(), SetLoggerError> {
    unsafe { openlog(self.ident.as_ptr(), self.logopt, self.facility); }
    // This statement might be slightly simplified for the slide deck.
    set_max_level(LevelFilter::Info);
    set_boxed_logger(Box::new(self))
  }
}
```

Creating libsyslog Crate

```rust
impl Drop for Syslog {
    fn drop(&mut self) {
        unsafe { closelog(); }
    }
}

impl log::Log for Syslog {
    fn enabled(&self, metadata: &Metadata) -> bool {
        &metadata.level().to_level_filter() <= self.module_levels.iter()
            .find(|(modpath, _)| metadata.target().starts_with(modpath))
            .map(|(_, level)| level)
            .unwrap_or(&self.level)
    }
```

src/syslog.rs
3/5

Creating libsyslog Crate

```rust
fn log(&self, record: &Record) {
  if self.enabled(record.metadata()) {
    if let (Ok(fmt), Ok(msg)) = ( CString::new("%s"),
      CString::new(format!("{}", record.args())))
    {
      let fmt_ptr = fmt.as_ptr();
      let msg_ptr = msg.as_ptr();
      match record.level() {
        Level::Debug => unsafe { syslog(LOG_DEBUG,   fmt_ptr, msg_ptr); }
        Level::Error => unsafe { syslog(LOG_ERR,     fmt_ptr, msg_ptr); }
        Level::Info  => unsafe { syslog(LOG_INFO,    fmt_ptr, msg_ptr); }
        Level::Warn  => unsafe { syslog(LOG_WARNING, fmt_ptr, msg_ptr); }
        Level::Trace => unsafe { syslog(LOG_DEBUG,   fmt_ptr, msg_ptr); }
      }
    }
  }
}
```

src/syslog.rs
4/5

Creating libsyslog Crate

```
    fn flush(&self) {}
}
```

src/syslog.rs
5/5

hellolib

Creating hellolib Crate

```
% cargo init --lib
     Created library package
% cargo add log
    Updating crates.io index
      Adding log v0.4.17 to dependencies.
             Features
✂         ✂         ✂         ✂         ✂         ✂         ✂
             - value-bag
```

Code Example, library

src/lib.rs

```
use log::info;

pub fn say_it() {
    info!("About to output hello world.");
    println!("Hello world!");
}
```

helloapp

Creating helloapp Crate

```
% cargo init
     Created binary (application) package
% cargo add libsyslog log
    Updating crates.io index
      Adding libsyslog v0.1.0 to dependencies.
      Adding log v0.4.17 to dependencies.
✂        ✂        ✂        ✂        ✂        ✂        ✂
% cargo add --path ../hellolib
      Adding hellolib (local) to dependencies
```

Code Example, application

```rust
use log::info;
use hellolib::say_it;

fn main() {
    libsyslog::Syslog::builder()
        .build()
        .init().unwrap();
    info!("Delegating greeting to hellolib::say_it()");
    say_it();
}
```

Code Example, application

```
    Finished dev [unoptimized + debuginfo] target(s) in 0.08s
     Running `target/debug/helloapp`
Hello world!
```

Code Example, application

```
illumos, rsyslogd
May 25 14:36:21 localhost helloapp[2716]: [ID 570825
    user.info] Delegating greeting to hellolib::say_it()
May 25 14:36:21 localhost helloapp[2716]: [ID 722573
    user.info] About to output hello world.

Linux, journald
May 25 14:36:39 laxa64 helloapp[792633]: Delegating greeting
    to hellolib::say_it()
May 25 14:36:39 laxa64 helloapp[792633]: About to output
    hello world.
```

Verified Working Platforms

--freebsd

--haiku

--illumos

--linux-gnu

--netbsd

--openbsd

Untested Platforms

--dragonfly

--linux-musl

--linux-uclibc

--nto-qnx710

AIX powerpc64-ibm-aix

Available Alternatives

Available Alternatives

Options/complements to `log`:

- `slog`
- `tracing`
- `log4rs`
- others?

Crate: syslog-tracing

Using tracing-syslog Crate

In summary:

- Validates that my thinking is sane.
  - I.e. also uses the POSIX interface.
- Uses API from `tracing` rather than from `log`.

Crate: log4rs-syslog

Using log4rs-syslog Crate

In summary:

- Also validates that my thinking is sane.
  - I.e. uses the POSIX interface.
- Ties into log4rs framework, rather than the simplistic log.

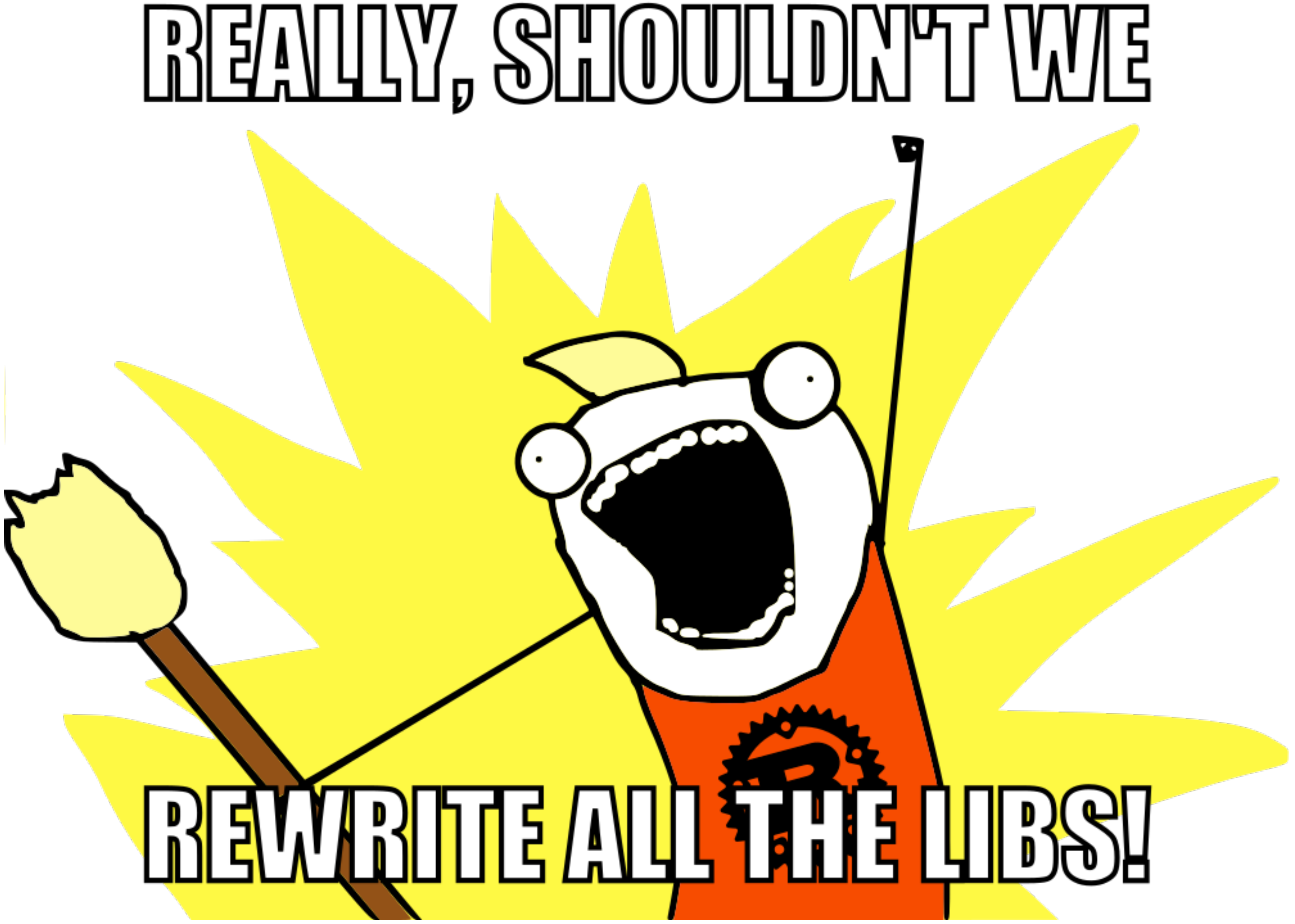
Future Work

Future, libsyslog(-sys)

Todo Tasks:

- Improve documentation.
- Wait for someone else to use it.
- Communicate with other crate owners.
- other things?
- Release 1.0.

Future, syslog

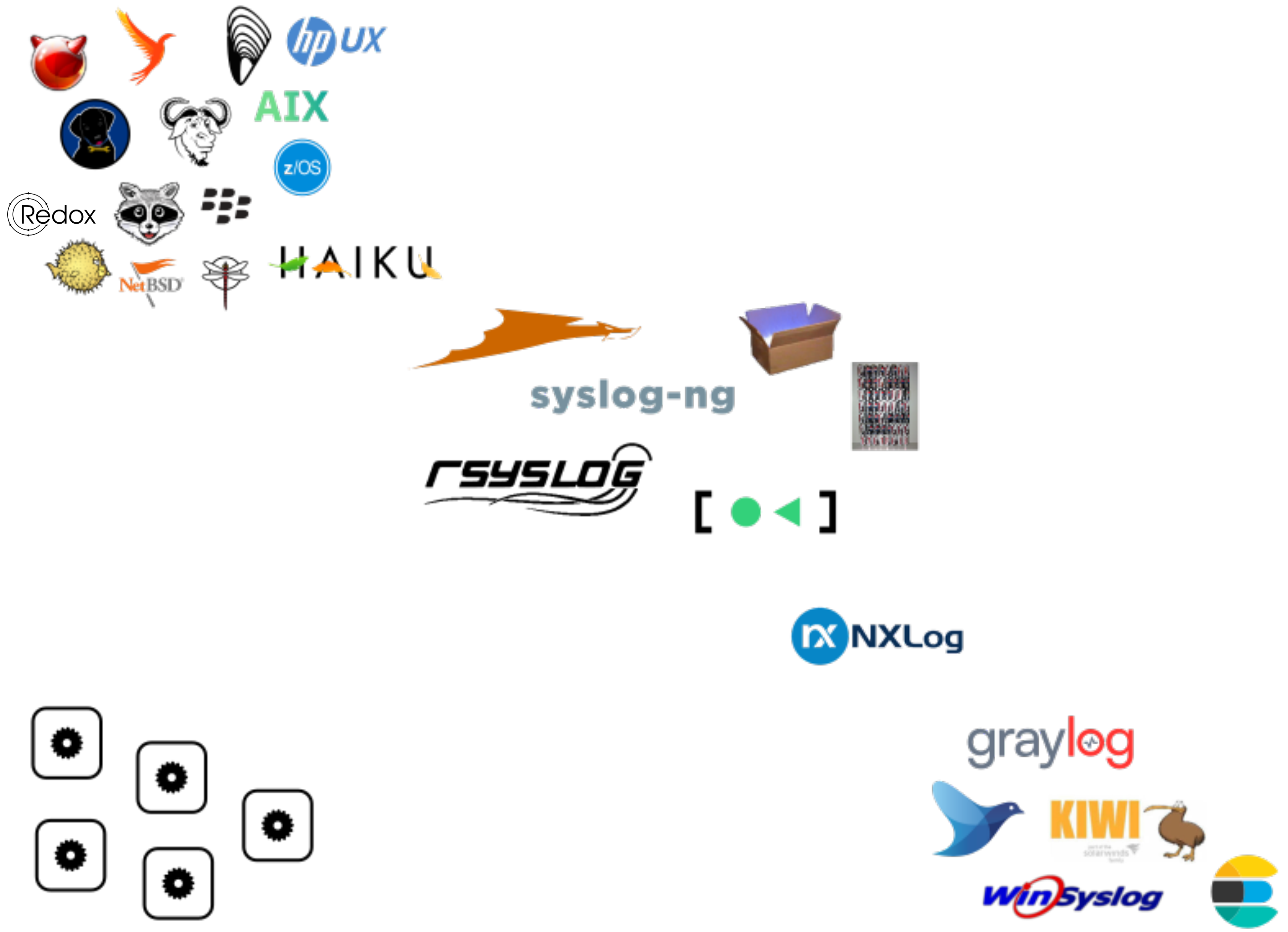

REALLY, SHOULDN'T WE
REWRITE ALL THE LIBS!

RIIR

Redox OS includes relibc, a libc implemented in Rust.

Seems to currently be lacking syslog() and friends.

Start here:
https://gitlab.redox-os.org/redox-os/relibc/-/issues/173


hp UX
AIX
z/OS
Redox
NetBSD
HAIKU
syslog-ng
rsyslog
NXLog
graylog
KIWI
WinSyslog

Case Studies

Case Study A

Ditch POSIX?

HelenOS … does not aspire to be a clone of any existing operating system and trades compatibility with legacy APIs for cleaner design.

http://www.helenos.org/wiki/Logging

Case Study B

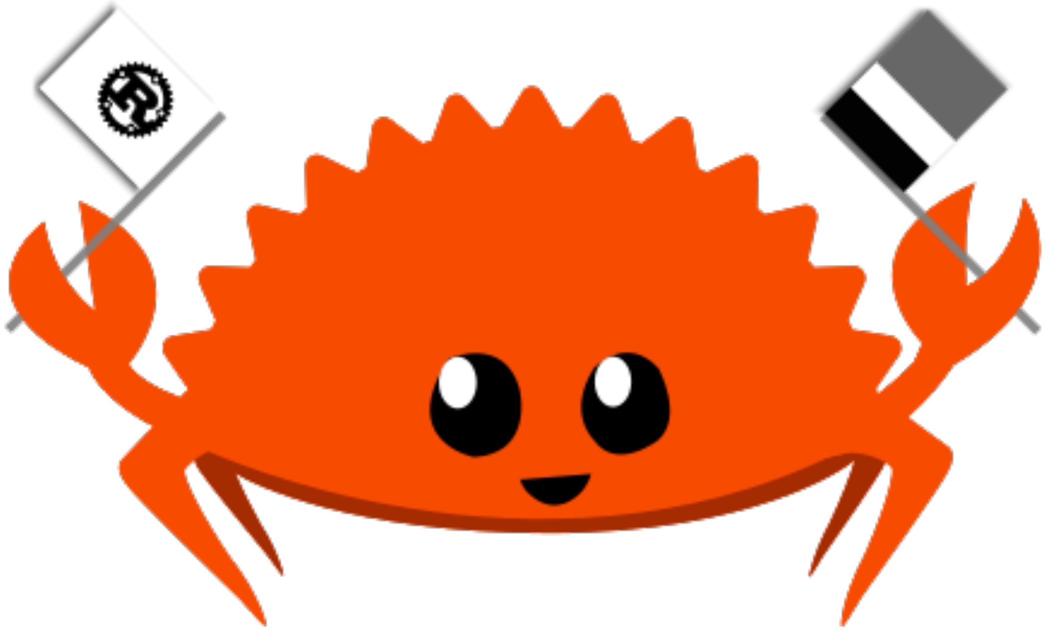
Break POSIX?

Calls to syslog() succeed, but the messages vanish...

Apple Inc. is advising to no longer use syslog on macOS 10.12 and later.

https://developer.apple.com/documentation/os/logging

Join the original Chat Room



[m] #rust-cph:matrix.org

Questions?

Contact Details

Martin "|cos|" Samuelsson

https://www.netizen.se/#contact