The Secure Socket API

TLS as an Operating System Service

Mark O'Neill, Scott Heidbrink, Jordan Whitehead, Tanner Perdue, Luke Dickinson, Torstein Collett, Nick Bonner, Kent Seamons, and Daniel Zappala

BYU
BRIGHAM YOUNG
UNIVERSITY

your apps are vulnerable


PayPal


amazon


CHASE

why?

“The root cause of most of these vulnerabilities is the **terrible design** of the APIs to the underlying SSL libraries”

--The most dangerous code in the world: validating SSL certificates in non-browser software. Martin Georgiev et al., 2012. ACM CCS.

using TLS is hard

Symbols in libssl: 504

SSL_CTX_set_verify()

X509_verify_cert()

SSL_CTX_set_cert_verify_callback()


Activities
Terminal
Thu 14:42
mark@localhost:~/bene-master/examples/openssl_examples/simplessl
File Edit View Search Terminal Help
317 Lines

can we do better?

can we use the POSIX socket API?

can we use the POSIX socket API?

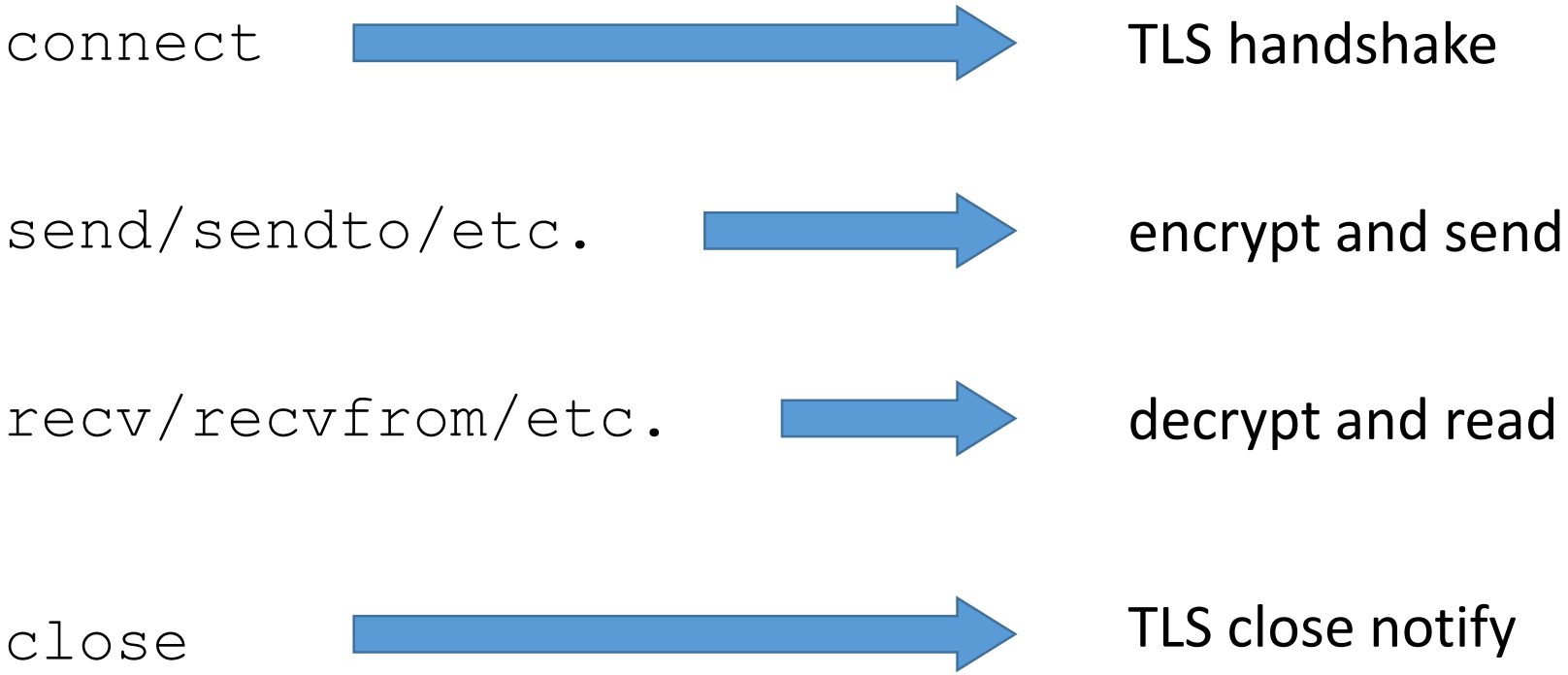
```
int socket = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
```

can we use the POSIX socket API?

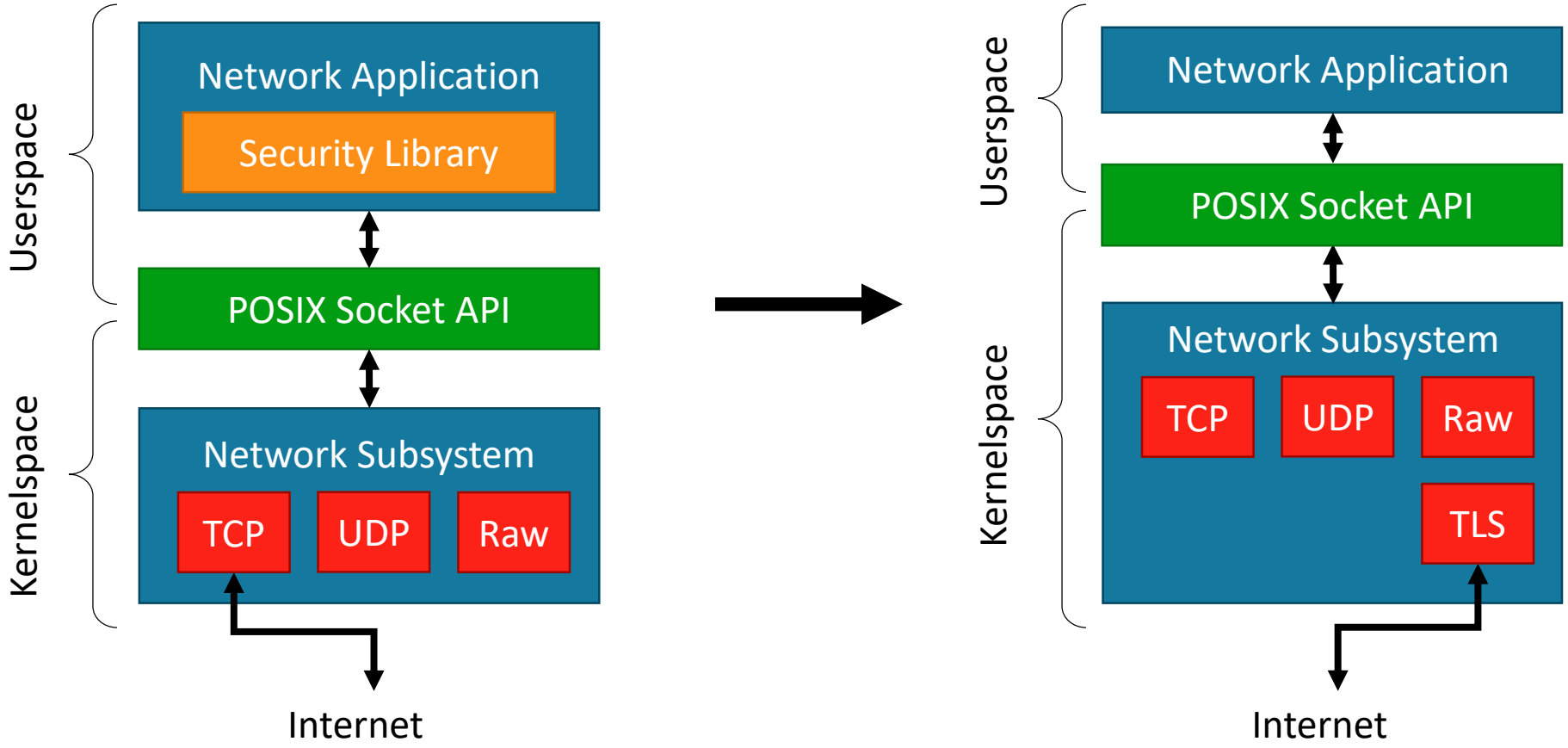
```
int socket = socket(PF_INET, SOCK_STREAM, IPPROTO_TLS);
```

the Secure Socket API (SSA)

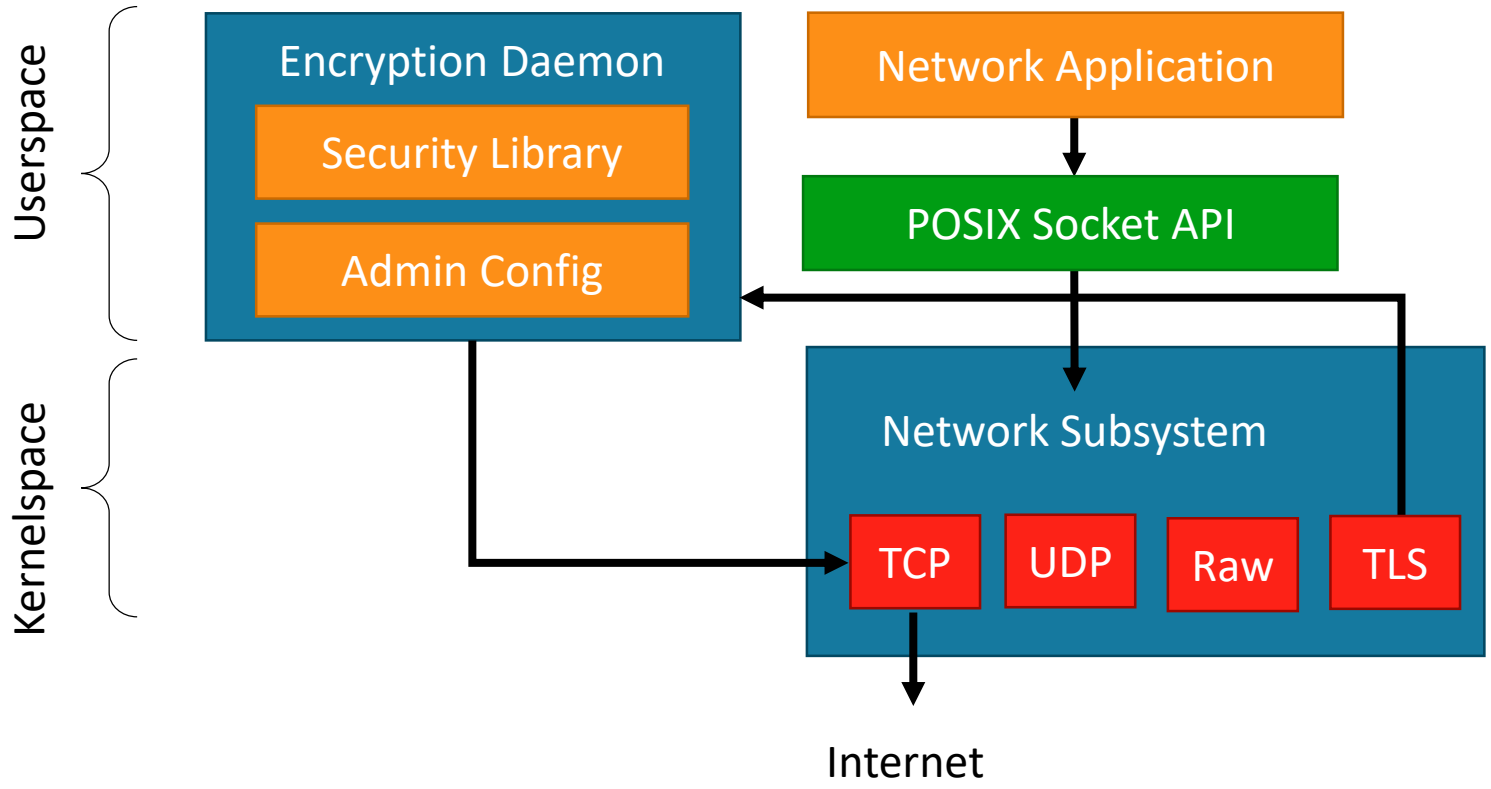
the Secure Socket API (SSA)



TLS via the POSIX socket API



Userspace Encryption Daemon



TLS API reduction

OpenSSL
SSL_CTX_new
SSL_CTX_set_verify
SSL_new
SSL_set_fd
TLS_method
SSL_exts_set_hostname
SSL_do_handshake
SSL_set_verify_callback
SSL_get_peer_certificate
And 495 more…

Symbol Count

504 ⟶ **14**

Secure Socket API
socket
bind
listen
connect
setsockopt
getsockopt
close
recv/recvfrom/recvmsg
send/sendto/sendmsg
getaddrinfo

```c
int main() {
        struct sockaddr_host addr;
        addr.sin_family = AF_HOSTNAME;
        strcpy(addr.sin_addr.name, "www.google.com");
        addr.sin_port = htons(443);

        int sock_fd = socket(PF_INET, SOCK_STREAM, IPPROTO_TLS);
        connect(sock_fd, (struct sockaddr*)&addr, sizeof(addr));

        char http_request[] = "GET / HTTP/1.1\r\nHost: www.google.com\r\n\r\n";
        char http_response[2048];
        memset(http_response, 0, 2048);
        send(sock_fd, http_request, sizeof(http_request)-1, 0);
        recv(sock_fd, http_response, 2047, 0);
        close(sock_fd);
        printf("Received:\n%s", http_response);
        return 0;
}
```

reconnaissance

Features	Symbols
version selection	29
cipher suite selection	39
extension management	68
certificate/key management	73
certificate/key validation	51
session management	61
configuration	19
allocation	33
connection management	41
miscellaneous	64
instrumentation	26

analyzed 410 Ubuntu packages that depended on libssl

used developer behavior to guide our design

developer options

setsockopt

getsockopt

developer options

```
...
fd = socket (PF_INET, SOCK_STREAM, IPPROTO_TLS);
/* Bind to local address and port */
bind (fd, &addr, sizeof(addr));
/* Assign certificate chain */
setsockopt(fd, IPPROTO_TLS, TLS_CERTIFICATE_CHAIN, CERT_FILE, sizeof(CERT_FILE));
/* Assign private key */
setsockopt(fd, IPPROTO_TLS, TLS_PRIVATE_KEY, KEY_FILE, sizeof(KEY_FILE));
...
```

developer options

`setsockopt`

`getsockopt`

Option
TLS_REMOTE_HOSTNAME
TLS_HOSTNAME
TLS_TRUSTED_PEER_CERTIFICATES
TLS_CERTIFICATE_CHAIN
TLS_PRIVATE_KEY
TLS_ALPN
TLS_SESSION_TTL
TLS_DISABLE_CIPHER
TLS_PEER_IDENTITY
TLS_PEER_CERTIFICATE_CHAIN

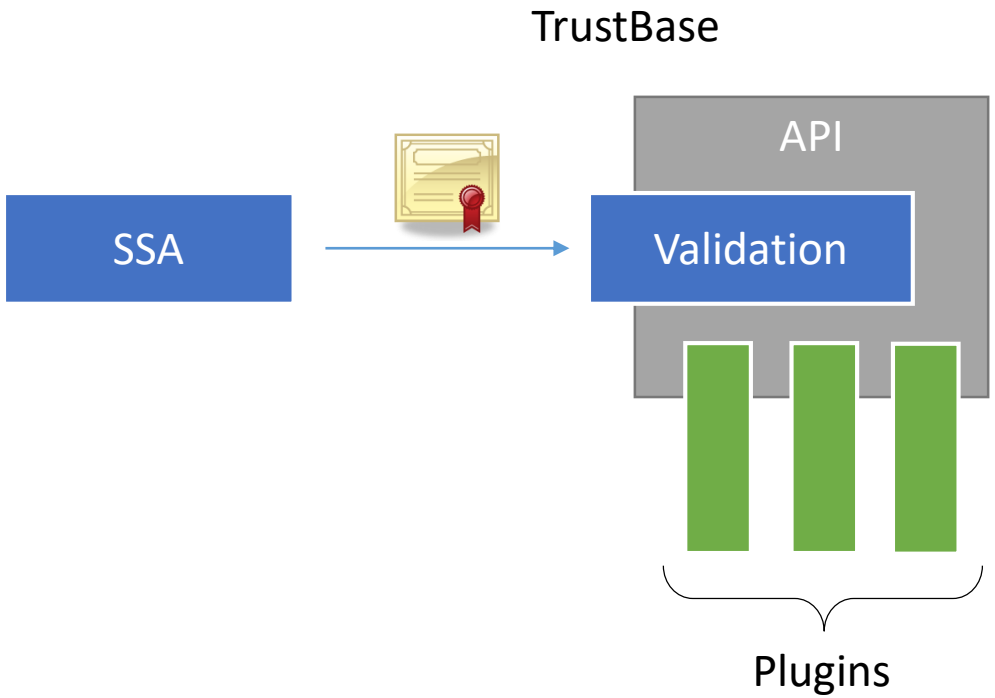
administrator options

- global configuration file assigns TLS defaults
- per-application profiles can further customize settings

Option	Description
TLS Version	Enabled TLS versions, in order of preference
Cipher Suites	Allowed cipher suites, in order of preference
Certificate Validation	Specified root store for certificate validation, or custom validation engine like TrustBase
Enabled Extensions	Specified TLS extensions to use (e.g., ALPN)
Session Caching	Specified session cache parameters
Default cert/key paths	Specify location of certificates and keys to use when application does not specify

certificate validation

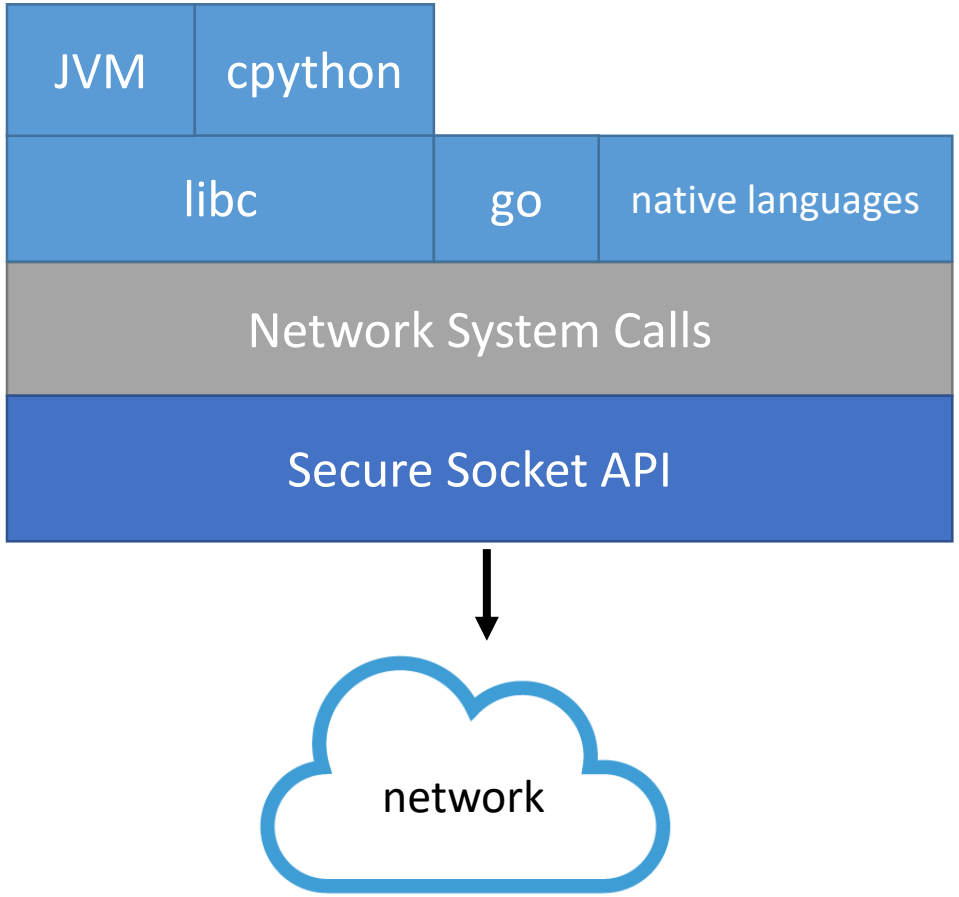
- admin's choice
  - standard validation
  - TrustBase
- TrustBase is an OS service that validates certificates according to admin config
- can enable multiple services (OSCP, CRLsets, custom root stores, Convergence, etc.)



using the SSA

Application	LOC Modified	LOC Removed	Familiar with Code?	Time Taken
Already using TLS				
wget	15	1,020	No	5 Hrs.
lighttpd	8	2,063	No	5 Hrs.
Not using TLS				
in-house webserver	5	0	Yes	5 Min.
netcat	5	0	No	10 Min.

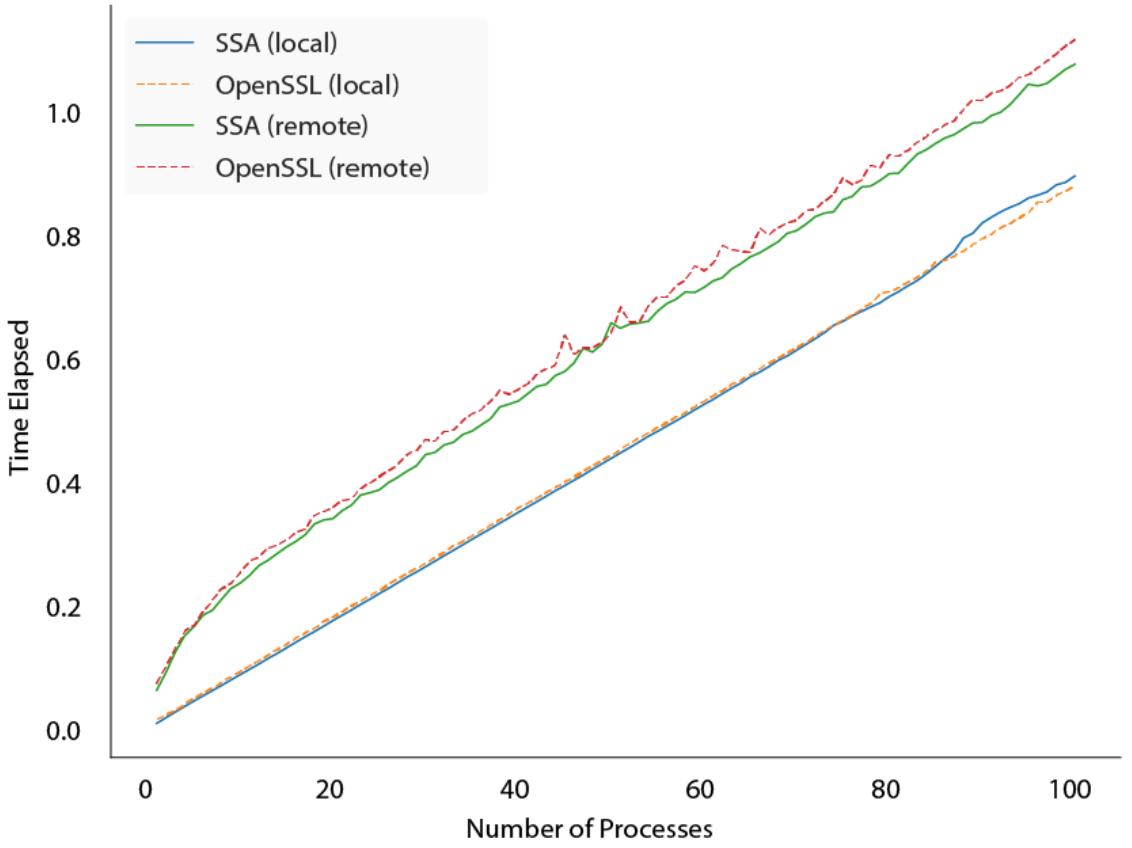
language support



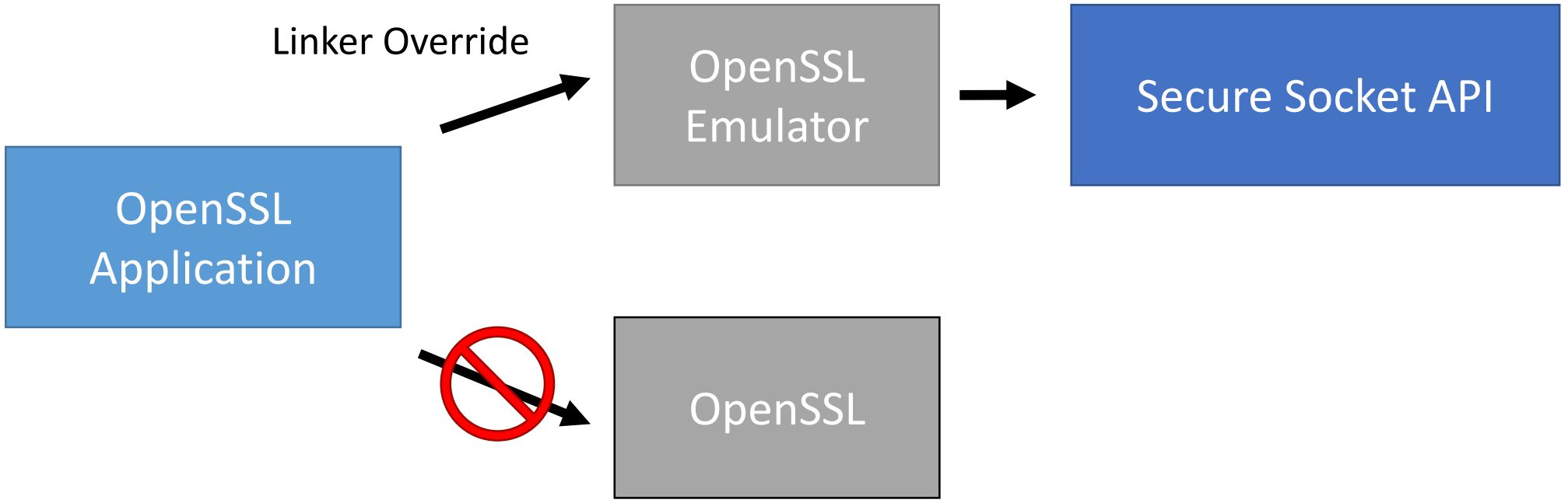
- any language that uses the network uses network system calls (directly or indirectly)
- the SSA is implemented behind the system call layer
- adding SSA support to a language is trivial
  - **Go**: < 50 lines of code (syscall wrappers)
  - **Python**: new constants only
  - **PHP**: new constants only
  - **C/C++**: new constants only

performance vs OpenSSL

- no discernable time overhead for 0 – 100 concurrent TLS-using processes



broadening coverage



dynamically ported ncat, wget, lighttpd, irssi

outcomes

- general benefits
  - TLS through a known API
  - admin control of TLS settings
- implementation benefits
  - easy language support
  - natural privilege separation
  - alternative implementations supported

the Secure Socket API:

enabling developers to **secure connections**

using a **known API**

in ways **you can control**

Thank You

- kernel module: https://github.com/markoneill/ssa
- encryption daemon: https://github.com/markoneill/ssa-daemon
- pull requests welcome!
- project website: https://owntrust.org
- contact me: mto@byu.edu

- thanks to our sponsors

Homeland Security

Image Attributions

- Productivity by Gregor Cresnar from the Noun Project
- confused by Gregor Cresnar from the Noun Project