Post-Quantum Algorithms in cURL

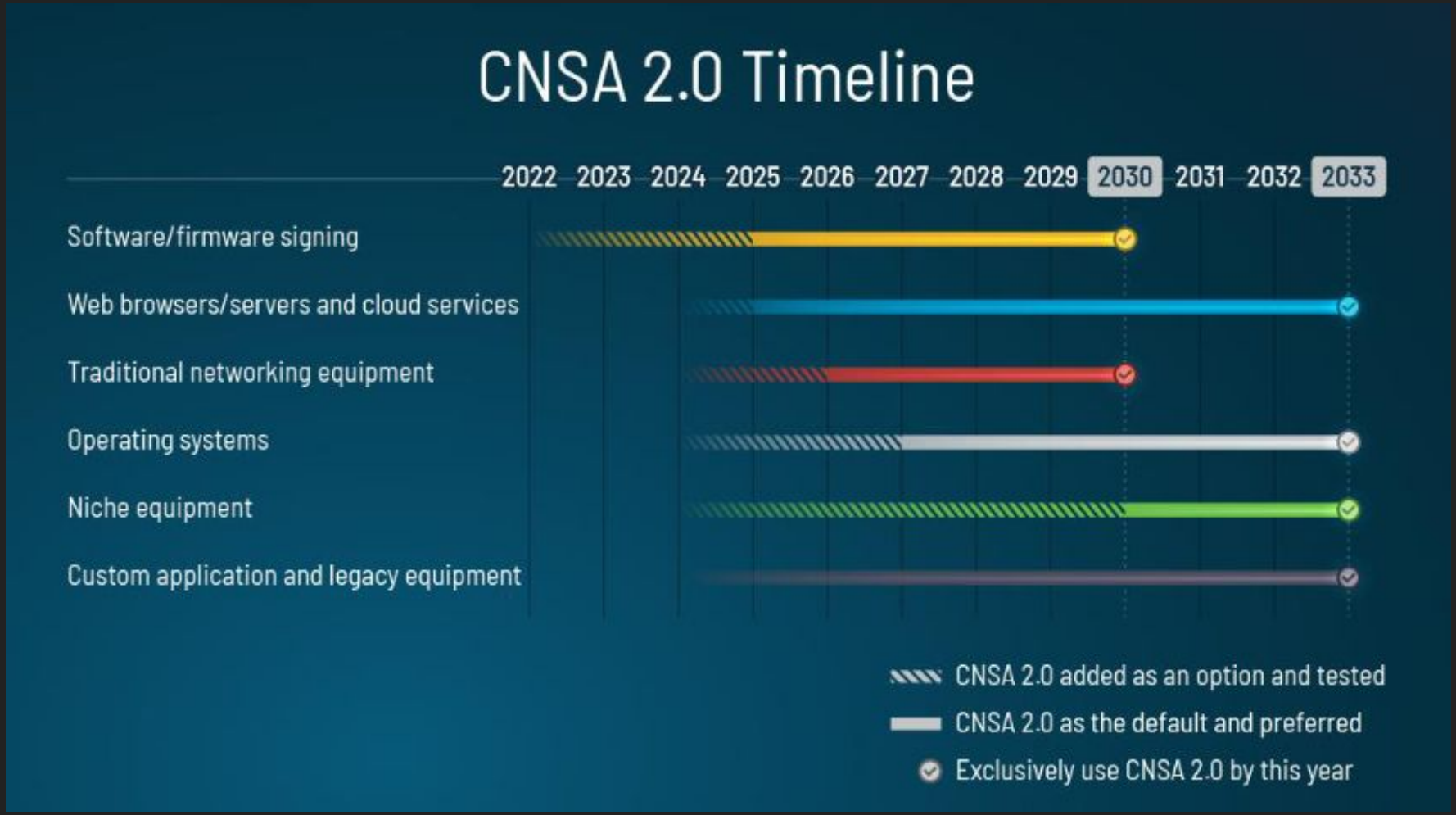
Anthony Hu

Open Source
Internet Security

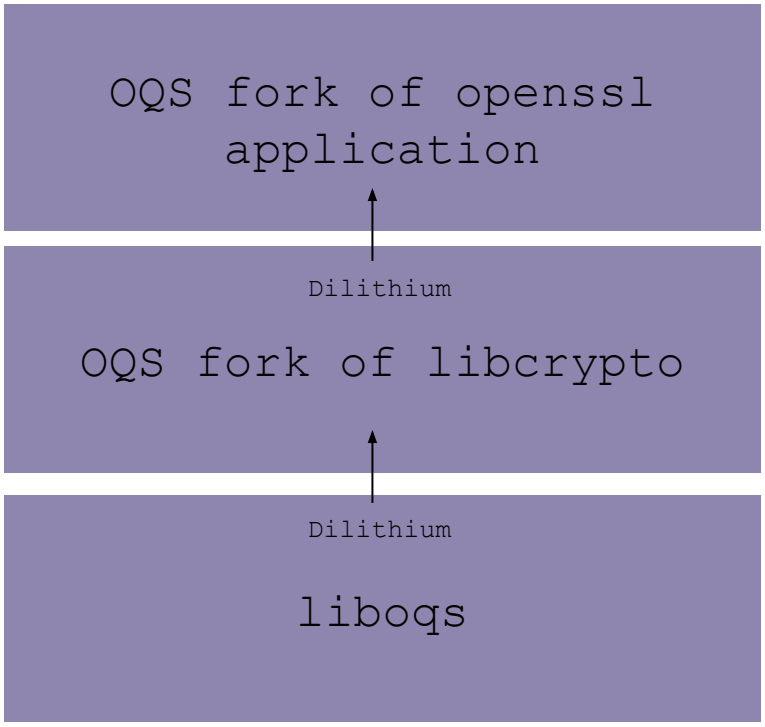
- **Dual Licensed GPLv2 and Commercial TLS implementation**
- **Securing over 2 billion connections world wide**
- **Meeting high standards of security (FIPS certificate, DO-178, extensive testing)**
- **Progressive cryptography leading TLS 1.3 adoption**
- **Resource conscious for use in embedded IoT scaled all the way up to large server farms**

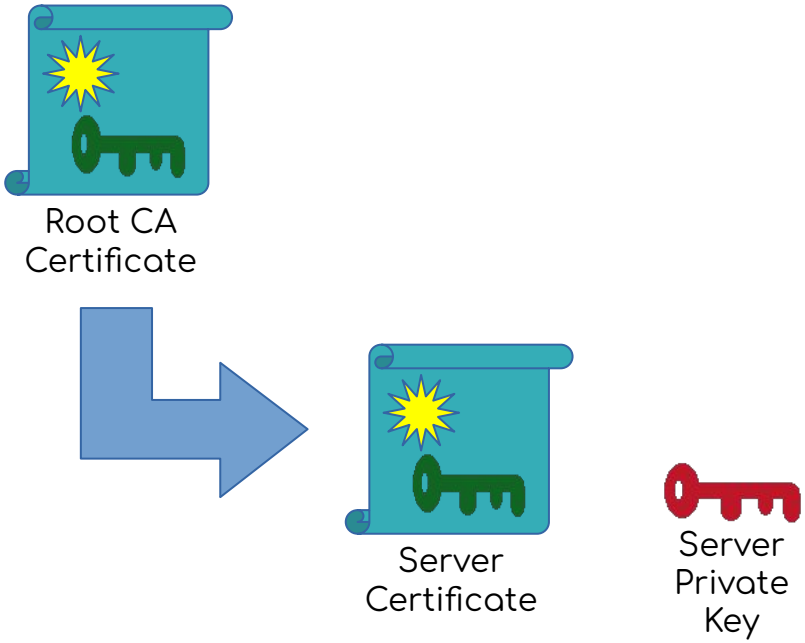
SYSGO EMBEDDING INNOVATIONS
RENESAS
Mentor Graphics
arm
Texas Instruments
Green Hills SOFTWARE
0x5 HEX-Five
MICROCHIP
NXP
life.augmented
XILINX
intel
DDC-I

Motivation: CNSA 2.0



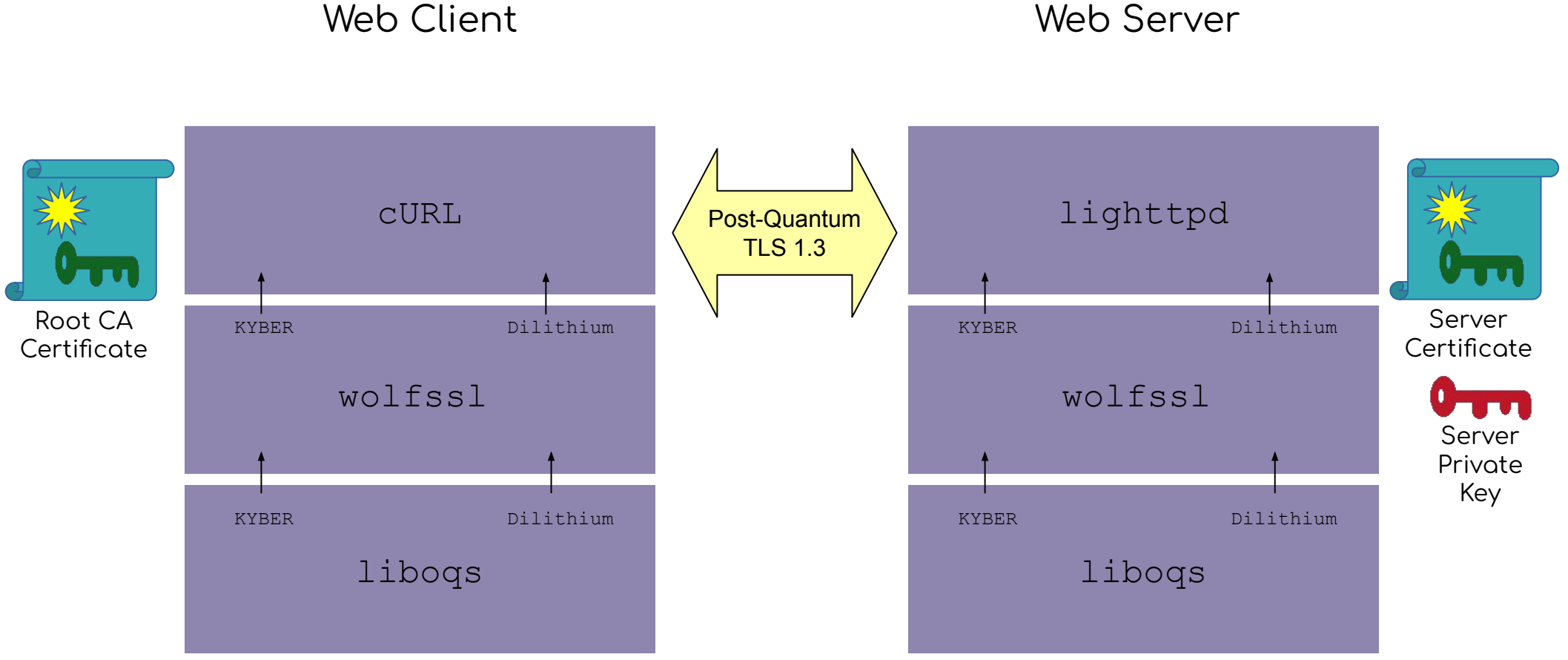
Demo Architecture: Preparations





The software stack generates the certificate chain.

Demo Architecture: Connection



Getting the Code

Project	Links/Commands
liboqs	https://github.com/open-quantum-safe/liboqs.git
OQS OpenSSL	https://github.com/open-quantum-safe/openssl/archive/refs/tags/OQS-OpenSSL_1_1_1-stable-snapshot-2021-08.tar.gz
wolfSSL	git clone https://github.com/wolfSSL/wolfssl.git
lighttpd	git clone https://github.com/anhu/lighttpd1.4
cURL	git clone https://github.com/curl/curl.git

Software Versions

Project	Version
liboqs	GIT Tag: `0.8.0`
OQS Project's OpenSSL Fork	GIT Tag: `OQS-OpenSSL_1_1_1-stable-snapshot-2021-08`
wolfSSL	GIT Tag: `v5.7.0-stable`
cURL	GIT Tag: `curl-8_8_0`
wolfSSL OSP	GIT Hash: `04215e201957c2613c039f639d855dcff2a6ab1d`
lighttpd	GIT Hash: 7b04c47d1178ebc5b48cc0ade4b2791fb9fa2aea GIT Branch: pq

Build liboqs

```
$ mkdir ~/oqs
$ cd ~/oqs
$ git clone --single-branch https://github.com/open-quantum-safe/liboqs.git
$ cd liboqs/
$ git checkout 0.8.0
$ mkdir build
$ cd build
$ cmake -DOQS_USE_OPENSSL=0 ..
$ make all
$ sudo make install
```

Build OQS's OpenSSL Fork and Generate Certificates

```
$ git clone --single-branch --branch=OQS-OpenSSL_1_1_1-stable
https://github.com/open-quantum-safe/openssl.git

$ cd openssl

$ git checkout  OQS-OpenSSL_1_1_1-stable-snapshot-2021-08

$ ./config no-shared

$ make all

$ cp /path/to/osp/oqs/generate_dilithium_chains.sh ./

$ ./generate_dilithium_chains.sh
```

Build wolfssl

```
$ git clone https://github.com/wolfssl/wolfssl

$ cd wolfssl

$ git checkout v5.7.0-stable

$ ./autogen.sh

$ ./configure --enable-lighty --enable-curl --enable-experimental
--with-liboqs

$ make all

$ sudo make install
```

Get OSP Repo

```
$ git clone https://github.com/wolfssl/osp
```

Build curl

```
$ autoreconf -fi

$ ./configure --with-wolfssl

$ make all

$ sudo make install
```

Build lighttpd

```
$ git clone https://github.com/anhu/lighttpd1.4 lighttpd

$ cd lighttpd

$ ./autogen.sh && ./configure --with-wolfssl --without-zlib --without-pcre

$ make all
```

Setup Certificates and Keys

```
$ cat /path/to/oqs/openssl/dilithium_level5_entity_key.pem \
      /path/to/oqs/openssl/dilithium_level5_entity_cert.pem > server.pem

$ cp /path/to/oqs/openssl/dilithium_level5_root_cert.pem ca-cert.pem
<optional>
```

index.html File

```
Congratulations!!!  You got Lighttpd with wolfSSL to run.
```

lighttpd.conf File

```
server.document-root = " /path/to/lighttpd "

server.modules += ( "mod_wolfssl" )

server.errorlog = " /path/to/lighttpd/ server_err.log"

server.port = 443

ssl.engine = "enable"

ssl.pemfile = " /path/to/lighttpd/ server.pem"

index-file.names = ( "index.html" )
```

Demo Time!

Execute lighttpd

```
$ cd /path/to/lighttpd/

$ sudo /usr/local/sbin/lighttpd -D -f lighttpd.conf
```

Quantum-Safe Connection

```
$ LD_LIBRARY_PATH=/usr/local/lib  /usr/local/bin/curl \
    --curves P521_KYBER_LEVEL5 \
    --cacert /path/to/oqs/openssl/dilithium_level5_root_cert.pem \
    --ciphers TLS_AES_256_GCM_SHA384 \
    https://127.0.0.1
```

Expected Output:

Congratulations!!! You got Lighttpd with wolfSSL to run.

Demo is Done

Hopefully nothing went wrong!

Quantum-Safe Connection Explained

Security Component	Algorithm	Quantum Threat Averted
Authentication	Dilithium Level 5	Shor's Algorithm
Key Establishment	Kyber Level 5	Shor's Algorithm
Symmetric Encryption	AES-256	Grover's Algorithm

Bonus: Hybrid Key Establishment with ECC P521 and symmetric encryption with AES-256 are required by FIPS 140-3!

Supported Post-Quantum Algorithms and Variants

Signature Schemes

- DILITHIUM_LEVEL2
- DILITHIUM_LEVEL3
- DILITHIUM_LEVEL5
- FALCON_LEVEL1
- FALCON_LEVEL5
- LMS/HSS (not appropriate for TLS)
- XMSS/XMSS^MT (not appropriate for TLS)
- SPHINCS+ (Many variants; not appropriate for TLS)

KEM Groups

- KYBER_LEVEL1
- KYBER_LEVEL3
- KYBER_LEVEL5

Hybrid Groups

- P256_KYBER_LEVEL1
- P384_KYBER_LEVEL3
- P521_KYBER_LEVEL5

NOTE: Our codepoints, OIDs and cryptographic artifacts are interoperable with OQS Project's OpenSSL fork.

Questions?

Email: facts@wolfssl.com