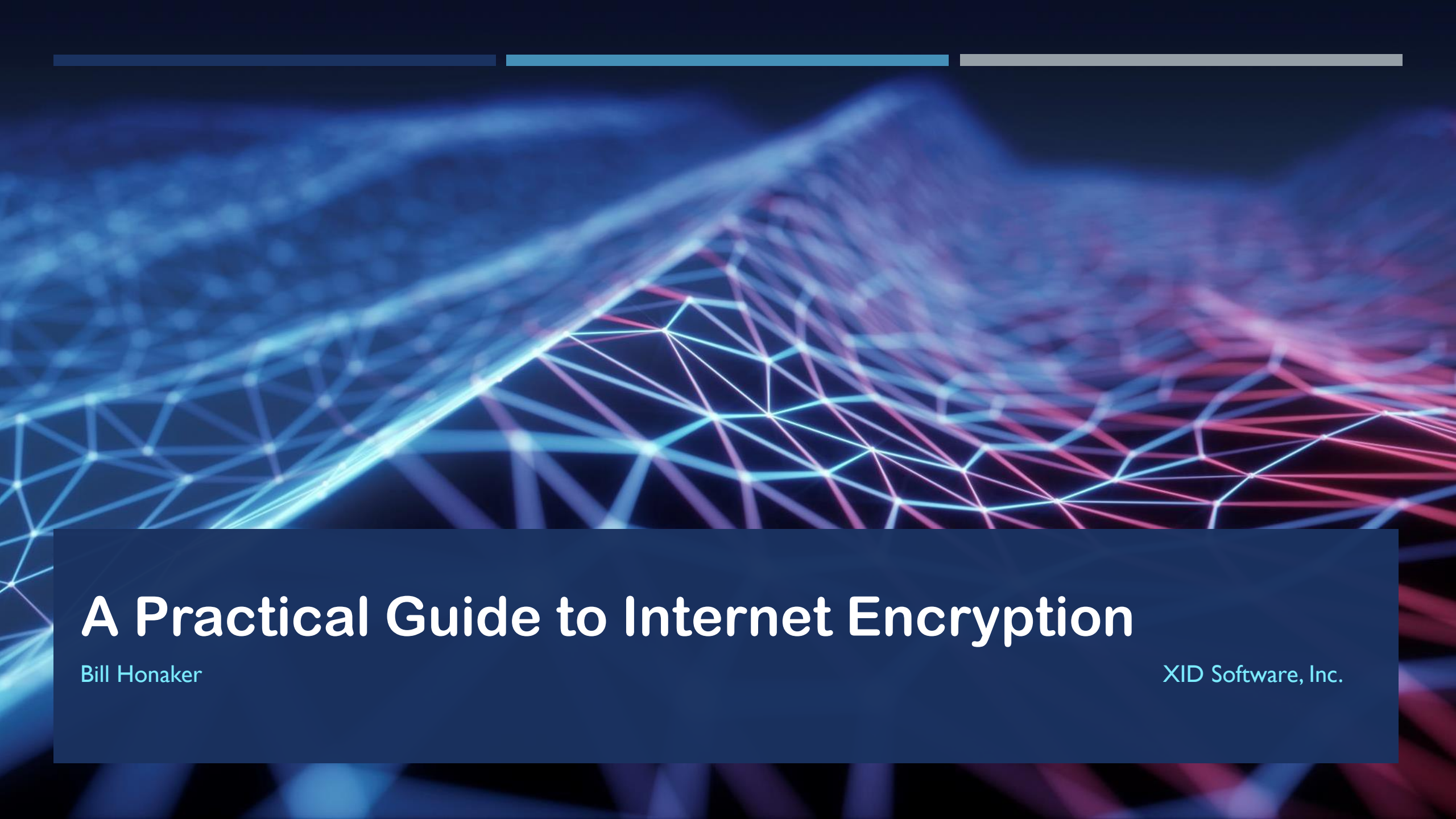

A Practical Guide to Internet Encryption

Bill Honaker

XID Software, Inc.

http://www.xid.com

Founded in 1989, XID Software, Inc. continues to be a leader in Technology Services throughout the United States. We have provided high quality services to an extensive array of industries.

The core values of our company are based upon the highest levels of integrity, reliability and total customer satisfaction. XID is headquartered in Euless, TX and is employee owned and operated. Our firm values are compelling to our many clients and alliances by helping them to achieve their business objectives.

Our diverse backgrounds have helped our clients manage through rapid changes in technology. XID has completed numerous projects for our clientele and has never wavered from our quality of work or our focus on their expectations.

We make it a point to understand your business and your technology needs before we offer solutions. Whether it is to create a new application, enhance an existing system or provide support for your application(s), we will accomplish it.

About XID

A Practical Guide to Internet Encryption
Agenda

- Encryption Basics
- Symmetric and Asymmetric Encryption
- SSH basics
- SSL/TLS basics
- NonStop specifics
- Futures
- Q&A

Encryption Basics

- Encryption keys are generated using common Ciphers, such as RSA, EAS, ECDSA
- Key generation starts with a (pseudo) Random Number
  - Generated by a 'Daemon' like PRNGD, or hardware such as the x86 DRNG
- Symmetric keys
  - One key used to encrypt and decrypt
  - Low CPU usage (fast)
  - Requires a secure means to share keys

Encryption Basics - Continued

- Asymmetric keys
  - A ‘public’ key (shareable) and a ‘private key’ (kept secure)
  - Primary cipher example: RSA
  - Encryption by one can only be decrypted by the other
  - Typically shared online, using secure methods such as Diffie-Hellman
  - Higher CPU usage (slow)

Asymmetric Encryption

- Typical Asymmetric Encryption-based session (over-simplified)
  - A client connects to a server requesting encryption, includes known cipher suites (ciphers and hash functions)
  - The server responds with information including a public key and a chosen cipher
  - The client generates a secret (a random number), encrypts with the public key, generates a 'shared secret' symmetric key, and sends encrypted secret to the server
  - The server decrypts the client secret with it's private key and the client secret, and generates what should be the same shared secret key
  - After receiving and verifying, the client sends the next message, encrypted with the shared secret key. The server checks that it can decrypt, and if so, sends back an acceptance message to the server
  - The rest of the session uses data encrypted with the shared secret symmetric key
  - Used by major protocols such as SSL/TLS and SSH

SSH

- SSH Key management is not usually automated; some employers generate for you
- A user requests a keypair to be generated
  - SSHCOM GENERATE KEY command
  - OpenSSH – ssh-keygen
    - Resulting file should use SSHCOM IMPORT KEY
    - Key can be shared on two clients – for example NonStop OSS (for git command line) and Eclipse
- Both will prompt you for Passphrase. This is useful in case your private key gets compromised (see below)
- Key generation creates 2 files. One contains only the public key, the other contains both keys
  - Users should make sure that only they have access to the private key file. Treat it like the keys to your locker
  - Both are simple text files, Usually Base64 encoded so they can be treated as text

Asymmetric Encryption in SSH -- Continued

- OpenSSH for NonStop (in ITUGLIB) is not certified for communications, only for key management tasks
- Passphrases
  - When created with the keypair, these must be provided when the key is used to connect
  - Usually, this is done via keyboard interface, but some clients (Eclipse for example) can save in an encrypted cache location
  - Similar to passwords, but verified locally (not sent to any servers)
  - Provides one method of two-factor authentication (2FA), since you must provide the private key but also the passphrase to authenticate

SSL/TLS

- SSL (Secure Socket Layer)
  - First version (1.0) released in 1994
  - Last version (3.0) released in 1996
  - All version of this are deprecated, and no longer considered secure
- TLS (Transport Layer Security)
  - First version (1.0) released in 1999
  - Latest version (1.3) released in 2018
  - Currently, versions 1.0 and 1.1 are no longer considered secure
- Used in many protocols, including HTTPS (web), SMTP/IMAP/POP3 (email), VPN

SSL/TLS Handshake

- Session setup is similar to other Asymmetric sessions, except:
  - Server sends PKI 'web server' certificate (containing the public key) instead of just the key
  - Client verifies trust chain of returned PKI certificate
  - There is an option of PKI certificates for the client, as well, typically server specific
- TLS 1.3 benefits
  - Older, obsolete cipher suites not shared, resulting in a shorter handshake message
  - Possible to establish sessions with fewer round trips
  - Session keys may be 'cached' for multiple sessions (for example, on web sites) to avoid handshakes
- Variant 'DTLS' is useful for UDP applications, such as streaming audio/video

SSL/TLS PKI Infrastructure

- Public Key Infrastructure
- Certificates contain public/private key plus other things, depending on need
  - A cryptographically protected signature to enhance authenticity
  - Web Server certificates include a 'Common Name' (CN), which is normally the Domain Name or wildcard of multiple domains. Clients verify that this matches the URL where the request was sent
  - Certificate Authority (CA) generates the certificate (including 'root' keys to provide a chain of trust)
    - Clients typically verify this chain against securely stored 'root' certificates
  - Most reputable CAs do background checks as part of the signature process and guarantee the authenticity

SSL/TLS PKI Infrastructure -- Continued

- Certificates are generated by a 'Certificate Authority (CA)'
  - Can be purchased/generated by a public CA such as Google, Digicert, Network Solutions.
    - Web servers generate a 'CSR' request, which contains a one-time-use key stored by the server. It also contains information such as the CN
    - Purchase is usually done first and then the Certificate Signing Request (CSR) is uploaded onto the CA's website
    - CA does verification prior to generating, then sends the key plus any trusted root and intermediate certificates
  - Enterprises can purchase a private CA certificate that itself is trusted by a public CA

SSL/TLS PKI Infrastructure -- Continued

- Trusted root certificate stores are available to clients and servers, varying by architecture
  - Java uses cacerts
  - OpenSSL uses digests (one file only) or directories (one file per certificate)
  - Other servers on NonStop (iTP, NS HTTP, Lightwave) specify server certificate (only) in config files
- Trusted root certificates are typically not used on Server connections
  - Only the server's web site certificate (including private and public keys) is used

SSL/TLS Client Certificates

- Some secure web services verify clients with PKI certificates, as well
  - Generated by an Enterprise CA, which must be trusted
  - Client must store these securely
  - Usually created with a PIN or Pass Phrase, which client must provide during handshake
  - Web servers on NonStop typically store these similarly to web site certificates
    - Client certificates are typically passed to CGI or NSJSP Servlets as well in HTTP headers

SSL/TLS PKI Infrastructure for Clients

- NonStop Java (NSJ) RVU includes the typical group of public Root certificates
  - Stored database file is /usr/tandem/java/jre/lib/security/cacerts
    - Can be overridden with -D option on JVM start
- OpenSSL doesn't come with a list of Root certificates
  - By default, looks in /usr/local[v]/ssl/certs
  - Application can choose another location
    - curl, for example uses -CAPath or -CAFile option on command line or with an environment variable
  - Can contain a 'Digest' file or a directory with certificates
  - Many Enterprises have a PKI team that can provide the certificate store files. This is strongly recommended

SSL/TLS PKI Infrastructure for Clients - Continued

- What if the certificate you need to trust is from a CA not in your trusted store?
  - For example, generated by an internal CA that isn't one trusted globally
  - However, these certificates should very likely be trusted by all internal client applications
- To overcome, you need 2 or more additional certificates:
  - The CA Certificate
  - One or more 'Intermediate' Certificates
  - This information is in the certificate chain in the certificate, but you will need to acquire them from the generating Certificate Authority
  - Java certificate store can be managed with the keytool utility, which is in the java/bin directory
  - OpenSSL certificates can be added multiple ways, Google is your best help here

SSL/TLS PKI in Your Enterprise

- Somewhere in your organization, there is most likely a team that manages PKI This may be under your CISO, or in your Networking group
- These teams manage PKI resources of all types, such as those stored in Active Directory, Azure or some other Enterprise CA
- To keep servers (and clients) in your private network in sync, they often provide certificate stores that include sites that they have verified
- Using those verified certificate stores is encouraged, as it can help you when audits occur
- Get to know this team, they are the ones that know your company's security rules

NonStop Specifics

- NonStop SSH as a Server
  - Terminal emulators, sftp clients, ssh clients can connect to your NonStop
  - User/password pairs are an authentication option, but this method is less secure, as the password must be 'shared'
  - Authentication with 'keypair' method is preferred. Client must generate a private RSA key pair. Each client looks in its own directories. CISO team may generate keys for clients
  - Authentication is controlled with SET CLIENTALLOWEDAUTHENTICATIONS
  - The public key must be added using SSHCOM in DAEMON mode by the ALTER USER, PUBLICKEY command
  - Both commands are typically available only to one single authorized user, often SUPER.SUPER

NonStop Specifics - Continued

- NonStop SSH as a Client
  - Keypairs are added in SSHCOM as previously mentioned (IMPORT or GENERATE)
  - TACL clients are SSH and SFTP
  - OSS clients are sshoss, sftposs, in /G/system/zssh
    - Perhaps use a symbolic link in the users' PATH
  - Open Source clients include git
    - To work, a 'wrapper' script must be configured – search HPE Scout with 'git sshoss' to find the Hotstuff describing how to do this
    - Note: github.com has been requiring stronger cipher suites to authenticate since April of this year

NonStop Specifics - Continued

- OpenSSL apps
  - SSL Toolkit (T2813)
    - This is used by the NonStop HTTP web server
    - The L22.09.00 release is based on OpenSSL 1.1.1
    - Later releases may be on 3.0.x or 3.1.x

NonStop Specifics - Continued

- OpenSSL apps - continued
  - OpenSSL from openssl.com are available at ITUGLIB in binary form
    - Current versions:3.1.x and 3.0.x.
      - Version 1.1.1 is out of support by OpenSSL as of 11 September 2023
      - Version 1.1.1w is the last free release and is available for download on ITUGLIB
      - Version 3.2 should be out in beta form soon – test your apps!
    - These versions support:
      - 64-bit and 32-bit applications. DLL (.so) and Static libraries (.a)
      - Thread neutral, SPT threads, or PUT threads
      - Future: KLT threads
    - OpenSSL is also available from openssl.com in source form. Download it and build it yourself
      - Some customers require this for auditor approval
  - Typical clients include curl, git, wget (also all in ITUGLIB)
    - At run time, export _RLD_LIB_PATH should include /usr/local[v]/lib. as well.

NonStop Specifics - Continued

- When building your own clients using OpenSSL:
  - Compile with -I /usr/local[v]/include
  - Link with –L to /usr/local[v]/lib
  - At run time, export _RLD_LIB_PATH should include /usr/local[v]/lib. as well.
  - For specialized trusted root support, alternate CA_FILE or CA_PATH can be provided
  - Pay attention to match up thread libraries if your application uses them
- When building clients using Java
  - javax.* packages contain very useful code supporting HTTPS

Planning for the Future – Post Quantum Cryptography (PQC)

- Many standards organizations recognize that Quantum Computers may someday have the ability to crack many cipher suites common in use today. Most of the weaker ones, which are deprecated, can already be cracked
- NIST, for example, has a list of proposed 'Symmetric Safe' ciphers. Some of these have been declared safe simply by doubling their key sizes. Some use more complex mathematical formulas, such as Elliptical Curve cryptography. Note that this list may contain ciphers that aren't safe from some **future** quantum capabilities, but that there is **no existing practical implementation** of a machine that is a danger to these ciphers.
- The Open Quantum Safe project has built an OpenSSL provider compatible with version 3.0+ of OpenSSL. This includes all candidates for validated PQC that are being investigated
- OpenSSL released a statement in October of last year that "no post-quantum algorithms have been selected for standardization by a national or international body". OpenSSL won't have built-in support for any until some are selected

Planning for the Future – Post Quantum Cryptography (PQC) -- Continued

- On August 24, NIST released draft standards of 3 algorithms that are proposed to make online communications secure against quantum systems. One of these algorithms deals with the handshake used to securely share a symmetric key between 2 computers
- The release article mentions one other previously-released algorithm, CRYSTALS-Kyber, which deals with secure web sites (and presumably SSH session partners). It details various Hash mechanisms and ciphers
- The other 2 algorithms are designed for digital signatures, such as those used in more advanced web server certificates as well as online commerce implementations

Questions and Answers

Thank You!

bhonaker@xid.com

XID TECHNOLOGY SOLUTIONS