FIPS 140-2 Compliance of Industry Protocols in 2015 and Beyond

Edward Morris

November 20, 2014

Gossamer Laboratories

Topics

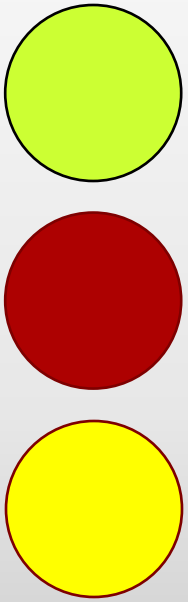
- Abstract
- Background
- The Protocols
  - IPsec
  - TLS
  - SSH
- Some Further Thoughts

Abstract

IPsec

TLS

SSH



Strengths of Some Algorithms

Bits of security	Symmetric key algs	FFC (e.g, DSA, DH)	IFC (e.g, RSA)	ECC (e.g., ECDSA)
56	DES	$L = 512$ $N = 112$	$k = 512$	$f = 112\text{-}159$
80	2TDEA, SKIPJACK	$L = 1024$ $N = 160$	$k = 1024$	$f = 160\text{-}223$
112	3TDEA	$L = 2048$ $N = 224$	$k = 2048$	$f = 224\text{-}255$
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

Background

- **History of SP 800-57, 800-131, 140-2 IG G.14 & G.15**
  - NIST withdrew 56-bit crypto in 2005 with 2 year transition.
  - NIST withdrew 80-bit crypto in 2011 with a 3/5 year transition
    - Allowed signature, sig. hashing, and key agreement/transport until 2014
    - Allowed encryption*, key wrap, and legacy RNGs until 2016.
  - ~ One year remaining now
  - Beyond 2016, the biggest disruption to industry protocols will likely come from SP 800-56 series.

The Protocols

- Internet Protocol Security (IPsec)
- Transport Layer Security (TLS)
- Secure Shell (SSH)

The Protocols

- Internet Protocol Security (IPsec)
- Transport Layer Security (TLS)
- Secure Shell (SSH)

Internet Protocol Security (IPsec)

- IPsec specified in many RFCs with many options.
- SP 800-57 Part 3 details IPsec and provides simplified guidance including guidance regarding 112-bit strength
- The three IPsec protocols allow a choice of algorithms
  - Peer Authentication and Key Exchange (KE) (e.g. IKE)
  - Subsequent protection of IP packets using either
    - Authenticated Header (AH) or
    - Encapsulated Security Payload

Internet Protocol Security (IPsec)

- Peer Authentication and Key Exchange typically accomplished by
  - Manual Keying or direct configuration of the IPsec Security Associations (SAs) - Acceptable in 2015 and beyond
  - Internet Key Exchange version 1 (IKEv1) - Unacceptable in 2014*
  - Internet Key Exchange version 2 (IKEv2) - Acceptable in 2015 if configured correctly

* 2014 compliant IKEv1 configuration may be possible; however, for several reasons (IETF and NDPP v1.1 recommendations) its simpler to avoid IKEv1

Internet Protocol Security (IPsec)

- IKEv2 RFCs defines negotiable elements for protocol SAs that should be chosen to be compliant for 2016.
- They include
  - Encryption:
  - Pseudo-random function:
  - Integrity:
  - Diffie-Hellman group:
  - Peer Authentication:

Internet Protocol Security (IPsec)

- Encryption:
  - 3 ENCR_3DES (TDES-~~168/~~192 in CBC mode),
  - 12 ENCR_AES_CBC (AES in CBC mode), or
  - 13-20 (other AES variants using CTR, CCM, and GCM).
- Pseudo-random function:
- Integrity:
- Diffie-Hellman group:
- Peer Authentication:

Internet Protocol Security (IPsec)

- Encryption:
- Pseudo-random function:
  - 2 PRF_HMAC_SHA1
  - 5 PRF_HMAC_SHA2_256
  - 6 PRF_HMAC_SHA2_384
  - 7 PRF_HMAC_SHA2_512
  - 8 PRF_AES128_CMAC
- Integrity:
- Diffie-Hellman group:
- Peer Authentication:

Internet Protocol Security (IPsec)

- Encryption:
- Pseudo-random function:
- Integrity:
  - 2 AUTH_HMAC_SHA1_96*
  - 7 AUTH_HMAC_SHA1_160
  - 8 AUTH_AES_CMAC_96*
  - 9-11 AUTH_AES_128|192|256_GMAC
  - 12-14 AUTH_HMAC_SHA2_256_128|384_192|512_256
- Diffie-Hellman group:
- Peer Authentication:

*see SP800-107 section 5.3.5

Internet Protocol Security (IPsec)

- Encryption:
- Pseudo-random function:
- Integrity:
- Diffie-Hellman group:
  - 14 2048-bit MODP Group
  - 15 3072-bit MODP Group
  - 16-30 excluding 22 (MODP, ECP and brainpool)
- Peer Authentication:

Internet Protocol Security (IPsec)

- Encryption:
- Pseudo-random function:
- Integrity:
- Diffie-Hellman group:
- Peer Authentication:
  - 1 RSA Digital Signature (2048-bit, e>=65537 w/ SHA-256)
  - 2 Shared Key Message Integrity Code (PSK >= 112-bits)
  - 9 ECDSA with SHA-256 on the P-256 curve
  - 10 ECDSA with SHA-384 on the P-384 curve
  - 11 ECDSA with SHA-512 on the P-512 curve

Internet Protocol Security (IPsec)

- In summary, IPsec well positioned for 112-bit strength
- NIST SP 800-57 part 3 provides a sensible recommendation for section of algorithms with a 112-bit security strength:
  - ESP Encryption: AES in CBC mode
  - ESP Integrity Protection: HMAC-SHA1
  - IKEv2 Encryption: AES in CBC mode
  - IKEv2 Pseudo-random function: HMAC-SHA1
  - IKEv2 Integrity: HMAC-SHA1
  - IKEv2 Diffie-Hellman group: 2048-bit MODP (DH14)
  - IKEv2 Peer Authentication: 2048-bit RSA with SHA-256

Internet Protocol Security (IPsec)

- Difficulties in practice
  - ESP & IKEv2 Encryption: TDES in CBC mode
    - Acceptable in 2016 provided use of 3 distinct keys (168-bits).
    - RFC 2451 section 2.2 specifies TDES keys to always be 168-bits
    - RFC 2451 section 2.3 requires implementations reject TDES keys where k1 == k2 or k2 == k3 (tantamount to single DES)
    - Logical to suggest that in 2016, implementations check to also reject k1 == k3.
  - May be simplest to avoid TDES all together and use AES instead, but not always possible

Internet Protocol Security (IPsec)

- Difficulties in practice
  - IKEv2 Peer Authentication: 2048-bit RSA with SHA-256
    - Generally we expect more (even complete) control of the peer certificate
    - While possible to configure 2048-bit certificates, it may not be possible to control the hashing algorithm used by the peer during authentication
    - While not part of IKE negotiations, some implementations allow configuration of the minimum strength for authentication and the certificate chain validation:
      - strongSwan 5.0.0+ allows one to specify in /etc/ipsec.conf:
      - `rightauth=rsa-2048-ecdsa-256-sha256-sha384-sha512`

The Protocols

- Internet Protocol Security (IPsec)
- Transport Layer Security (TLS)
- Secure Shell (SSH)

Transport Layer Security (TLS)

- Also detailed in SP 800-57 Part 3, which provides very good guidance (including guidance regarding 112-bit strength).
- TLS is not the same SSL v3.0, TLS is equivalent to SSL v3.1.
- The CMVP does not allow SSL, only TLS (due to the PRF)*
- The variable aspects of TLS include
  - TLS version
  - Certificates (Server and optional Client authentication)
  - Cryptographic algorithms comprising the cipher suite (key exchange, encryption, and message authentication algorithms)

*NIST SP 800-57 part 3 does not distinguish between SSL 3.0 and TLS 1.0

Transport Layer Security (TLS)

- All versions of TLS (1.0, 1.1, and 1.2), if paired with the appropriately cipher suite and public keys, can provide 112-bits of security.
- Certificates used by the server (and client, if used) must
  - Use SHA2 hashes (no SHA-1 or MD5)
  - Use keys of size 2048-bits or larger (for RSA, DSS, and DH)
  - Use ECDH/ECDSA curves with size 224 or larger
- If using Pre-shared Keys in lieu of Certificates, ensure the PSK >= 112-bits.

Transport Layer Security (TLS)

- Approved Key Exchange algorithms include:
  - RSA (RSA Key Transport)
  - DH_DSS*, DH_RSA
  - DHE_DSS*, DHE_RSA
  - PSK, DHE_PSK, RSA_PSK, ECDHE_PSK
  - ECDH_ECDSA, ECDHE_ECDSA, ECDH_RSA, ECDHE_RSA
- Non-Approved Key Exchange algorithms include any of
  - *_anon
  - SRP_*
  - KRB5

* unclear whether or not DSS keys are limited to 1024-bits

Transport Layer Security (TLS)

- Approved Encryption algorithms include:
  - 3DES_EDE_CBC
  - AES_128_CBC , AES_256_CBC
  - AES_128_GCM, AES_256_GCM
  - AES_128_CCM, AES_256_CCM
- Does not include any other encryption ciphers: NULL, DES, RC2, RC4, IDEA, etc.
- Approved Message Authentication Algorithms include:
  - SHA, SHA256, SHA384, CCM
- But not the MD5 Message Authentication Algorithm

Transport Layer Security (TLS)

- So summarizing TLS
- Ensure use of only TLS (and not SSL)
- Generate/Load public keys of appropriate size and hash
- Choose a compliant cipher suite (SP 800-57 Part 3 has a handy list)
- Example:
  - TLS 1.0
  - Generate a Server RSA 2048 with SHA-256 Certificate
  - TLS_RSA_WITH_AES_128_CBC_SHA

Transport Layer Security (TLS)

- Problems with TLS
- Generate a Server RSA 2048 Certificate that will sign with sign using SHA-256
  - Same problem as with IPsec, the server needs to support SHA-256
- Clients can't require RSA 2048 w/SHA-256 in < TLS 1.2
  - Only TLS v1.2 allows a client and server to fully specify the signatures strength, even then, only when using ECDSA
  - Implementation support for this not common yet.

"For TLS v1.2 you have to restrict the supported signature algorithms to exclude SHA1, allowing only SHA256 and above. There is no way to do that in OpenSSL 1.0.1 clients. Similarly the supported EC curves have to be restricted to exclude some which are of insufficient field size. In summary: it's a bloody mess."

- http://wiki.openssl.org/index.php/FIPS_mode_and_TLS

The Protocols

- Internet Protocol Security (IPsec)
- Transport Layer Security (TLS)
- Secure Shell (SSH)

Secure Shell (SSH)

- A future version of SP 800-57 Part 3 may cover SSH, but does not currently provide detail.
- Only SSHv2 can be used (known vulnerabilities in v1).
- Compliance of SSH requires selection of cryptographic algorithms. RFCs specifies options for
  - Ciphers
  - MACs
  - Key Exchange
  - Public Key

Secure Shell (SSH)

- Selection of 2016 compliant cryptographic algorithms.
  - Ciphers
    - 3des-cbc (REQUIRED)(168-bit key)
    - aes128-cbc (RECOMMENDED), aes192-cbc, aes256-cbc
    - aes128-ctr, aes192-ctr, aes256-ctr
    - AEAD_AES_128_GCM, AEAD_AES_256_GCM
  - MACs
  - Key Exchange
  - Public Key

Secure Shell (SSH)

- Selection of 2016 compliant cryptographic algorithms.
  - Ciphers
  - MACs
    - `hmac-sha1 (REQUIRED)`
    - `hmac-sha1-96 (RECOMMENDED)`
    - `AEAD_AES_128_GCM, AEAD_AES_256_GCM`
    - `hmac-sha2-256, hmac-sha2-512`
  - Key Exchange
  - Public Key

Secure Shell (SSH)

- Selection of 2016 compliant cryptographic algorithms.
  - Ciphers
  - MACs
  - Key Exchange
    - ~~diffie-hellman-group1-sha1 (REQUIRED)~~
    - diffie-hellman-group-exchange-sha1
    - diffie-hellman-group-exchange-sha256 (both above must use a suitably sized dh group)
    - diffie-hellman-group14-sha1 (REQUIRED)
    - ecdh-sha2-*
    - ecmqv-sha2
    - rsa2048-sha256
  - Public Key

Secure Shell (SSH)

- Selection of 2016 compliant cryptographic algorithms.
  - Ciphers
  - MACs
  - Key Exchange
  - Public Key
    - ~~ssd-dss (REQUIRED)~~
    - ~~ssh-rsa (RECOMMENDED)~~
    - ecdsa-sha2-*
    - x509v3-rsa2048-sha256
    - x509v3-ecdsa-sha2-*

Secure Shell (SSH)

- Summary Transport Layer Protocol Example Configuration:
  - Ciphers: `3des-cbc` (required) & `aes128-cbc` (recommended)
  - MACs: `hmac-sha1`(required) & `hmac-sha1-96` (recommended)
  - Key Exchange: `diffie-hellman-group14-sha1` (required) but exclude `diffie-hellman-group1-sha1` (required by RFC)
  - Public Key: very tricky, `ecdsa-sha2-nistp256` (or similar) and exclude RPC required `ssh-dss` and recommended `ssh-rsa` options.

Secure Shell (SSH)

- Problems with SSH
  - Very wide spread (large problem)
  - May be difficult to upgrade or replace the servers
- But SSH is not as bad off as TLS
  - SSH typically used for remote administrator configuration,
  - Clients typically pin the server's public key (some browsers starting to do this)
  - A given user may only SSH into a handful of devices

Some Further Thoughts

- Where FIPS 140-2 excels
  - Algorithm testing (CAVP)
  - Focus on cryptographic key management
- Where FIPS 140-2 can improve
  - Myopic with respect to boundary issues
  - Applied to both products and toolkits
  - Standard's requirements are divorced from intent/rationale

Questions?

Contacts:

- Ed Morris
  - EdMorris@gossamersec.com

www.gossamersec.com
www.facebook.com/gossamersec
@gossamersec