SKEME: A Versatile Secure Key Exchange Mechanism for Internet

Hugo Krawczyk

IBM – T. J. Watson Research Center

– ISOC SNDSS'96 –

Key Exchange

- key exchange as the key component of a key management solution
- enabler of shared-key cryptography (efficiency!)
- needed all the way from link security to application layer security
- SKEME: motivated by needs of IP layer security (ipsec WG)
- originally intended to broaden and strengthen Karn's Photuris
- applicable to other layers (e.g., secure telnet), to other standards (e.g., IEEE framework), etc.

SKEME Highlights

- Challenge: Accommodate the heterogenous scenarios and requirements of Internet into a simple and compact protocol
- support a diversity of security models
- wide range of requirements
- tradeoffs security-performance
- well founded security (analysis yet to be formalized)
- true algorithm independence (implem'n and analysis)
- small set of well-defined options
- uniform message format

Security Models

- **Public Key Model:** authenticated public keys, certificates (manual distribution, PGP, PKI), scalable trust model
- **Key Distribution Centers:** trusted third party, Kerberos, widely deployed, important to support, important to improve (PFS)
- **Manual Key Installation:** rudimentary but useful, first stage, the mobile-to-workstation example

(starting with a shared key vs. starting with a public key, the key refreshment case)

Requirements (ipsec and beyond)

- **Secrecy and Authenticity:** secrecy of exchanged key (no partial leakage), other party's identity, freshness; cryptographic analysis

- **Key Refreshment:** automatic refresh mechanisms short-lived vs. long-lived keys, from inexpensive operations to full-security, independence of keys, limit exposure and cryptanalysis (e.g., weak algorithms)

- **Perfect Forward Secrecy:** "disclosure of long-term secret keying material does not compromise the secrecy of exchanged keys from earlier runs." [DvOW] – the Diffie-Hellman algorithm and more...

- **Key Separation:** cryptographically independent keys for different cryptographic functions, no re-use of keys (keyed-MD5 vs DES), session key as seed, algorithm identifiers

Requirements (cont.)

- **Privacy and Anonymity:** hiding identities from eavesdroppers (and from active attackers), e.g., mobile users and temporary IP addresses, digital signatures and "proofs of communication"

- **Clogging Attacks:** alleviate denial of service attacks, public key operations as a clogging opportunity, Photuris: awareness and solution (the "cookies" technique)

- **Performance:** flexible tradeoffs: public key operations, symmetric key operations, Diffie-Hellman; amount of communication

- **Multiple Security Models:** PK, KDC, manual (presented before)

Requirements (cont.)

- **Algorithm Independence:** functional specification vs. implementation choices, replacement of algorithms (new algorithms, cryptanalysis, performance), negotiation; generic analysis

- **Exportability:** avoid weaknesses from regulated cryptography, e.g., message authentication vs encryption, mandatory symmetric key encryption

- **Minimize Protocol Complexity:** small set of well-defined options, uniform message format for the different options, compact representation

Selective Perfect Forward Secrecy

- exposure of long-lived keys does not expose short-lived keys (private keys, master keys, sessions keys)
- Diffie-Hellman as the only practical solution: computationally expensive
- Yes PFS: confidential traffic, long-lived keys (e.g., corporate firewall, mobile to workstation), the KDC case
- No PFS: authentication only (e.g., AH, NFS server), low secrecy (eg, timely financial information, weak algorithms)
- SELECTIVE PFS as a requirement (cf. Photuris and SKIP)!

Signatures and proof of communication

- non-repudiation as a disadvantage
- signing other party's identity
- proof of communication: a privacy concern
- more severe than identity disclosure
- signing of identity can be forced: nonces, DH exponents, etc.
- best solution: avoid signatures in the key exchange protocol

The SKEME Protocol

Cryptographic functions:

- public key encryption function (e.g., RSA, ElGamal)
  Notation: $PKE_A(\mathrm{info})$ (semantic security)

- Diffie-Hellman exchange (prime modulus, elliptic curves, etc.); Notation: g^x, g^y

- pseudorandom functions [GGM]
  - keyed functions: F_K
  - values in different inputs are "cryptographically independent"
  - serves for MAC and key generation
  - e.g., keyed hash: keyed-MD5
  - e.g., block cipher's MAC: DES-CBC-MAC, IDEA, etc.

- cryptographic hash function (e.g., MD5, SHA)
  Notation: H ("good randomizer")

- notation: SK (session key), id_A, id_B (identities)

Basic Phases

SHARE:

$$A \rightarrow B: \quad PKE_B(K_A)$$
$$B \rightarrow A: \quad PKE_A(K_B)$$
$$K_0 = H(K_A, K_B)$$

$A \rightarrow B: \quad PKE_B(id_A, K_A)$: anonymity

EXCH:

$$A \rightarrow B: \quad g^x \bmod p$$
$$B \rightarrow A: \quad g^y \bmod p$$

AUTH:

$$A \rightarrow B: \quad F_{K_0}(g^y, g^x, id_A, id_B)$$
$$B \rightarrow A: \quad F_{K_0}(g^x, g^y, id_B, id_A)$$

$SK := H(g^{xy} \bmod p)$

(cookies omitted)

Basic Phases: Rationale

- key authentication: via public keys and K_0 (notice: no digital signatures!)
- freshness (via DH exponents $=$ challenges)
- PFS: Diffie-Hellman
- *implicit* key confirmation: avoids computation delays
- use of H: extracting better randomness

- Pre-shared key: just omit the SHARE phase (e.g., KDC!)
- Design idea: reduce public key-based exchange to symmetric-key techniques (great for flexibility and analysis)

SKEME – Basic Protocol

A $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ B

$$\xrightarrow{\quad PKE_B(id_A, K_A), \quad g^x \quad}$$

$$K_0 := H(K_A, K_B)$$

$$\xleftarrow{\quad PKE_A(K_B), \quad g^y, \quad F_{K_0}(g^x, g^y, id_B, id_A) \quad}$$

$$K_0 := H(K_A, K_B)$$

$$\xrightarrow{\quad F_{K_0}(g^y, g^x, id_A, id_B) \quad}$$

$$SK := H(g^{xy} \bmod p)$$

Phases Without Diffie-Hellman

SHARE: $A \to B$: $PKE_B(id_A, K_A)$
$B \to A$: $PKE_A(K_B)$
$K_0 = H(K_A, K_B)$

EXCH: $A \to B$: $nonce_A$
$B \to A$: $nonce_B$

AUTH: $A \to B$: $F_{K_0}(nonce_B, nonce_A, id_A, id_B)$
$B \to A$: $F_{K_0}(nonce_A, nonce_B, id_B, id_A)$

- no PFS: but needs to break *both* private keys
- pre-shared K_0: key refreshment (very deficient!)
- well-studied shared key techniques: [NS, Bird*, BR]

SKEME– **Combined Protocol**

A $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ B

$$\xrightarrow{\quad [ENC_B(id_A, K_A),]\ \ value_A \quad}$$

$$\xleftarrow{\quad [ENC_A(K_B),]\ \ value_B,\ \ F_{K_0}(value_A, value_B, id_B, id_A) \quad}$$

$$\xrightarrow{\quad \overbrace{F_{K_0}(value_B, value_A, id_A, id_B)}^{arg} \quad}$$

$Mode$	PK	DH	$value_A$	$value_B$	K_0	SK	Comments
basic protocol (full strength)	Y	Y	g^x	g^y	$H(K_A, K_B)$	$H(g^{xy})$	PK+PFS
pre-shared key	N	Y	g^x	g^y	pre-shared (input to protocol)	$H(g^{xy})$	omit PK, e.g., Kerberos+PFS
share only (no PFS)	Y	N	$nonce_A$	$nonce_B$	$H(K_A, K_B)$	$F_{K_0}(arg)$	Effic (no DH) No PFS
fast re-key	N	N	$nonce_A$	$nonce_B$	pre-shared (input to protocol)	$F_{K_0}(arg)$	MD5-only no PK, no DH

Security Analysis

- pre-shared key mode: well analyzed (Bellare-Rogaway, Crypto'93)
- adding Diffie-Hellman: exponents as nonces plus DH conjecture
- adding public key derivation (of K_0): direct from semantic security of public key encryption
- yet to be formalized: especially missing definitions (work in progress)

Details and Implementation

- many incomplete details

- exact message formats, re-transmission, error handling, fault tolerance, SPI's (key identifiers)

- mode and parameter negotiations: yes/no PFS, specific algorithms, etc.

- cryptographic implementation: some guidance but details omitted; e.g., generation of DH exponents (off-line/on-line computation), key derivation (for different algorithms), formatting of encrypted data (e.g., OAEP)

- implementation in progress: some details (e.g., negotiation) borrowed from Photuris – future report

Comparison with Photuris: SKEME Advantages

- selective PFS (Diffie-Hellman)
- broader functionality: well-designed support for shared key modes
- key refreshment through efficient hand-shakes
- avoidance of digital signatures (better privacy and no symmetric encryption)
- anonymity against active attackers
- DH computation can be completed after the communication (avoid delays)
- use of generic cryptographic functions (e.g., not tailored to MD5)
- better basis for cryptographic analysis (reduction to symmetric-key techniques)

- we borrowed cookies, negotiation, and other details from Photuris. *Shouldn't Photuris borrow some more of our design...?*

And what about SKIP?

- replace SKEME's public key encryption with static DH ($K_0 = g^{x_A x_B}$) and voilà...

SKIP with selective PFS