Certicom's Bulletin of Security and Cryptography

C0DE&CIPHER

This issue of Code and Cipher focusses on the subject of key establishment in current security standards and highlights the importance of this often overlooked but central area of concern.

Key Establishment Schemes

an overview

The rapid growth of information technology that has resulted in significant advances in cryptography to protect the integrity and confidentiality of data is astounding. New algorithms have been introduced such as the Advanced Encryption Standard (AES) as defined in the Federal Information Processing Standard (FIPS) 197 to offer three security strengths: 128 bits, 192 bits and 256 bits.

The use of AES requires the establishment of shared keying material in advance. Manual distribution methods such as trusted couriers are inefficient and complex. They simply do not scale as the system grows. Key establishment schemes are required to distribute keys in today's communication systems. Protocols such as S/MIME, SSL and IPSec all use key establishment schemes.

Key establishment is *so fundamental* to security that the American National Standards Institute (ANSI) and the National Institute of Standards and Technology (NIST) are producing standards and recommendations for key establishment. For example, security products intended for government use will require FIPS 140-2 Validated modules with an approved key establishment method.

Cryptographic Elements

Before using key establishment schemes, there are a number of cryptographic elements that must be generated. Some of these elements must be kept private and some are made public. These elements include, among others, the generation, validation and authentication of static and ephemeral public-key key-pairs.

An entity's static key-pair, as the name implies, is a long-term key-pair for the user and must be generated in a trusted manner. Static public-keys and user credentials may be digitally signed by a trusted third-party (i.e. a Certification Authority) to provide a strong level of assurance to recipients that the public-key can be trusted.

An entity's ephemeral key pair is intended for exactly one use. The keys are created, used once in the calculation of a *key establishment primitive* and then destroyed immediately after the shared secret is computed. ANSI X9.42 and X9.63 describe the use of primitives for the calculation of the shared secret.

A primitive is a cryptographic building block that is used to facilitate the implementation of more complicated schemes. Each key establishment scheme, as outlined by NIST, requires the use of one primitive that is based on either the Diffie-Hellman (DH) or the

continues on page 4

Crypto News

JOINT VENTURE TO BRING ECC TECHNOLOGY TO THE MARKET PLACE
Hitachi, Mitsubishi, And NTT Develop New Encryption Technology

In July, Hitachi, Mitsubishi Electric, and NTT Corporation announced their success in mutually researching and developing an implementation technology of an elliptic curve cryptosystem, named "CRESERC." To establish this technology, Hitachi, Mitsubishi Electric and NTT founded a project team to research and develop a secure and efficient implementation for ECDSA signatures. According to the release: "This is the world's first case of well-established leaders in the field of cryptography collaborating in the development of implementation technology by integrating their advanced skills and technologies."

Visit http://www.ntt.co.jp/news/news03e/0307/030728.html for more details.

Crypto Column by Dr. Scott Vanstone

Key Establishment and ECMQV

The importance of key establishment schemes and the underlying cryptographic primitives cannot be understated. Most standards groups equate good security with the use of the Advanced Encryption Standard (AES). For example, AES is seen as the fix for wireless LAN 802.11i security (Counter mode with CBC-MAC protocol CCMP) to replace RC4. While AES is a good choice for bulk encryption, the flaws in WEP are largely related to key establishment. Plus, this move to AES will also impose a full-blown hardware upgrade for the manufacturers.

For 802.11i security, key establishment is derived from 802.1x, a port level authentication protocol. However, the security of this architecture is not well understood. Standards groups would be well advised to follow the key establishment recommendations proposed by NIST.

NIST Special Publication 800-56: Recommendation on Key Establishment Schemes (Draft) presents many flexible options for key establishment to address different industry scenarios. NIST clearly understands the many security attributes and bases their recommendations on well-studied cryptographic protocols and primitives.

NIST is not alone in these recommendations. Many open, security standards, such as ANSI X9, IEEE 1363-2000, IETF TLS/SSL, IETF IPSec and S/MIME, also propose use of these same key establishment schemes that have proven to be quite robust in real world implementations.

These key establishment schemes include Diffie-Hellman (DH) key agreement and its improvement, Menezes-Qu-Vanstone (MQV). While DH offers a very simple way of creating a shared secret between two entities, there are some major security weaknesses that must be overcome.

The major flaw that MQV addresses relates to a malicious user stealing a private key and using it to masquerade as a third party to the user whose key was compromised. Security protocols such as IPSec overcome this weakness by adding another step in the protocol. However, this makes the protocol less efficient. If you could find a key establishment scheme that offers the required security attributes with significant performance advantages then why not use it?

…the major flaw that MQV addresses relates to a malicious user stealing a private key and using it to masquerade as a third party…

The ECMQV scheme offers the performance advantages because elliptic curve groups have smaller key sizes and faster computations than other types groups of similar security while offering equivalent security. This makes MQV extremely well suited for very constrained environments such as smart cards, mobile devices and RFIDs.

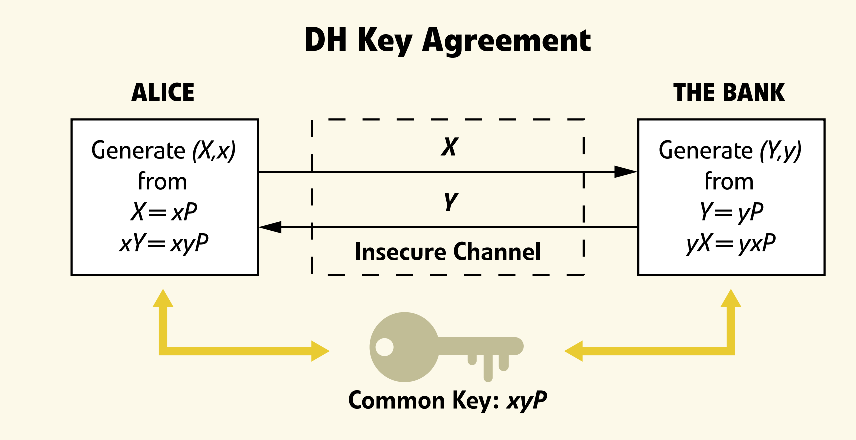
When we take these attributes into account and the fact that NIST and ANSI have already standardized MQV, it only makes sense that security protocols such as 802.11i should at least consider it as the primitive for key establishment.

MQV: Efficient and Authenticated Key Agreement

Many systems need to use well-established cryptography to protect the integrity and confidentiality of the data. Symmetric algorithms such as the 128-bit Advanced Encryption Standard (AES) provide strong protection beyond the year 2036. However, the use of AES requires the establishment of shared keying material in advance.

While a courier could be used to manually distribute the keying material, this is not practical when the number of people using a system grows because the distribution work grows quadratically. Therefore, it's essential to support symmetric algorithms with key agreement schemes.

Key agreement is a technique for parties who wish to engage in secure communications in which both parties contribute to the establishment of a shared secret. The shared secret is used to derive a symmetric key, which in turn is used to establish a secure channel. The method by which the shared secret is generated is known as key agreement and it is useful for applications in which parties are exchanging data in real time. A critical requirement for key agreement is the assurance that eavesdroppers are unable to obtain the shared secret.


DH Key Agreement
ALICE
THE BANK
Generate (X,x) from X=xP xY=xyP
X
Y
Insecure Channel
Generate (Y,y) from Y=yP yX=yxP
Common Key: xyP

Alice and the bank agree to use DH to obtain a shared secret. Alice generates a sessional key pair (X,x) by randomly generating x and calculating $X = xP$ where x is an integer and X is a point on an elliptic curve. The bank generates a sessional key pair (Y,y) by randomly generating y and calculating $Y = yP$. P is a generating point on the elliptic curve.

Alice sends X to the bank; the bank sends Y to Alice. Alice receives Y from the bank and calculates $xY = xyP$. The bank receives X from Alice and calculates $yX = yxP$.

They now have the same shared secret xyP.

Figure 1: DH Key Agreement

The article on Key Establishment Schemes (found in this issue of *Code and Cipher*) outlines a number of different options for key establishment and the desirable security attributes. This article focuses on two schemes for key agreement: the Elliptic Curve Diffie-Hellman (ECDH) and Elliptic Curve Menezes-Qu-Vanstone (MQV).

ECDH Key Agreement

ECDH is the elliptic curve analog of the traditional Diffie-Hellman key agreement algorithm. The Diffie-Hellman method requires no prior contact between the two parties. Each party generates a dynamic, or ephemeral, public key and private key. They exchange these public keys. Each party then combines its private key with the other party's public key to form the shared secret. This method is also known as carrying out an ECDH key agreement.

An example of ECDH key agreement is a home-banking subscriber, Alice, setting up a secure communication channel with her bank. Alice generates a public key and a private key; she sends the public key to her bank. Independently, her bank generates a public key and a private key; the bank's public key is sent to Alice. Alice combines her private key and the bank's public key to form a shared secret. Her bank combines its private key and Alice's public key to arrive at the same shared secret. The shared secret may now be used to generate a key for encrypting and decrypting the communication session. *Figure 1* illustrates the use of DH for this session.

The problem with DH is that Alice may be setting up a secure session with someone impersonating her bank. Because the private and public keys are generated on the fly, there is no way to prove you have a secure session with the intended party unless you add a method for user authentication. MQV addresses this issue.

MQV Key Agreement

The MQV elliptic curve key agreement method is used to establish a shared secret between parties who already possess trusted copies of each other's static public keys. Both parties still generate dynamic public and private keys and then exchange public keys. However, upon receipt of the other party's public key, each party calculates a quantity called an implicit signature using its own private key and the other party's public key. The shared secret is then generated from the implicit signature. The term implicit signature is used to indicate that the shared secrets do not agree if the other party's public key is not employed, thus giving implicit verification that the public secret is generated by the public party. An attempt at interception will fail as the shared secrets will not be the same shared secrets because the adversary's private key is not linked to the trusted public key.

To return to the example of Alice communicating with her bank. If Alice has the bank's public key and the bank has Alice's public key then the MQV key exchange may be used. Anyone intercepting the transmissions and substituting the public key is unable to communicate because the resulting shared secrets differ.

continues on page 6

Key Establishment Schemes

continued from page 1

Menezes-Qu-Vanstone (MQV) algorithm. These algorithms can be computed using Discrete-Log Cryptosystem (DLC) over finite-fields as in ANSI X9.42 or using Elliptic Curve Cryptography (ECC) as in ANSI X9.63.

As a result there are a number of possible primitive combinations to choose from for your key establishment scheme. For example, if you chose the MQV algorithm as the primitive to calculate the shared secret, you would also need to decide if MQV is calculated using DLC over a finite field or ECC. Similarly, you could chose DH as the primitive but again you would need to decide how it's calculated.

Although all schemes are approved, there are a number of drawbacks of using DLC over finite-fields and of using DH. One drawback is that, with DLC over finite fields, the system needs to process very large keys for new symmetric ciphers such as AES. In a system using 256-bit AES, this would require a massive public-key pair *on the order of 15,000 bits*, which is not be feasible for most applications.

Relevant Standards & Recommendations

American National Standards Institute (ANSI) X9, Inc.:

- **ANSI X9.42 Agreement of Symmetric Keys using Discrete Logarithm Cryptography**
- **ANSI X9.44 Key Agreement and Key Transport using Factoring-Based Cryptography**
- **ANSI X9.63 Key Agreement and Key Transport using Elliptic Curve Cryptography**

National Institute of Standards and Technology (NIST)

- **NIST Special Publication 800-57, Guidelines for Key Management.**
- **NIST Special Publication 800-56: Recommendation on Key Establishment Schemes**

Key establishment schemes using the MQV primitives provide assurance to each entity that if a malicious entity compromises their static private key, the malicious entity cannot masquerade as a third party to the entity whose key was compromised. For example, if a malicious entity, E, compromises entity U's static private key, then E cannot masquerade as any other party to U. Key establishment schemes using the DH primitive do not provide this assurance.

Key Establishment Schemes

With the primitives and computations selected, key establishment can begin once the system parameters are generated and the public-keys distributed throughout the system.

Key establishment is the process by which two (or more) entities establish a shared secret key. Essentially, two methods are used to establish cryptographic keying material between parties: key agreement and key transport.

With a key agreement scheme, all parties contribute to the derived keying material with information that allows each party to derive the shared keying material. In a key transport scheme, one party determines the keying material that is wrapped (i.e. encrypted) and transported to the intended receivers.

With key transport schemes, the sender determines the key to be transported, wraps (i.e. encrypts) the key and sends the wrapped key to the receiver, who then unwraps (i.e., decrypts) the key. If the scheme is a symmetric key only system, the sender and receiver have manually established a symmetric key to be used as the key-wrapping key between the two parties. The keying material is wrapped using a NIST-approved key-wrapping algorithm (such as the AES key wrap algorithm).

Public-key based key agreement schemes can be transformed into a key transport scheme using an approved key-wrapping scheme, as recommended in NIST SP 800-56. An example is S/MIME, where key agreement is combined with key wrapping to achieve the effect of key transport. This is useful because it allows the efficient encryption of large emails to multiple recipients. The efficiency is that the email content is encrypted just once, with the content encryption key encrypted (wrapped) multiple times, once for each recipient.

There are three major categories of key agreement schemes defined in the standards with two of these categories having multiple cases:

- Two-Party Participation: an interactive, two-way method where each party generates an ephemeral key pair. This method is used in the most widely deployed security protocols (for example IPSec).
- One-Party Participation: a store-and-forward, one-way method where only the initiator generates an ephemeral key pair. This method is ideally suited to email and is used in the S/MIME protocol. It can also be used in SSL if the server has a static DH public-key.
- Static Keys Only: a static (passive) method where each party has only a static key pair, no ephemeral keys are used. This method can be used in S/MIME and SSL but the absence of ephemeral keys diminishes its security. In this method, the shared symmetric keys are only assured to be distinct from previous by adding unencrypted (public) nonces to the derivation of the shared keys.

With all these options available for key establishment, the question becomes which scheme to use and when? In general, you need to determine which attributes are important for the level of security.

continues on page 5

CATEGORY	CASES	SECURITY ATTRIBUTES
Two Party Participation	U and V generate an ephemeral key pair and have a static key pair	• Known-key security • Forward secrecy • Key-compromise impersonation resilience (MQV primitive only) • Unknown key-share resilience (MQV primitive only) • Key control • Identity assurance
	U and V generate an ephemeral key pair; no static keys are used	• Known-key security • Forward secrecy • Key control
One Party Participation	Initiator U has a static key pair and generates an ephemeral key pair; Responder V has only a static key pair	• Known-key security • Forward secrecy • Key-compromise impersonation resilience (MQV primitive only) • Unknown key-share resilience (MQV primitive only) • Key control • Identity assurance
	Initiator U generates an ephemeral key pair; the Responder V has only a static key pair	• Known-key security • Forward secrecy for initiator only • Key control • Identity assurance for responder only
Static Keys Only	U and V have a static key pair	• Key-compromise impersonation resilience (MQV primitive only) • Unknown key-share resilience (MQV primitive only) • Identity assurance • Known-key security and forward secrecy are possible if variability is introduced in the shared secret

Table 1: Key Agreement Schemes

Security Attributes of Key Agreements Schemes

A secure protocol should be able to withstand both passive attacks and active attacks. In a passive attack, an adversary attempts to prevent a protocol from achieving its goals by merely observing honest entities carrying out a protocol. In an active attack, an adversary subverts the communications by injecting, deleting, altering or replaying messages.

The following are the primary security attributes that key establishment schemes can possess (explained by way of example between entities U and V):

- Known-key security: If one session key is compromised then neither the private keys nor session keys (both past and future) are compromised as a result.
- Forward secrecy: If private keys are compromised, the secrecy of previous session keys should not be affected.
- Key-compromise impersonation resilience: If U's private key is exposed, it does not enable an adversary to impersonate other entities to U.
- Unknown key-share resilience: Entity U cannot be coerced into sharing a key with entity V without U's knowledge.
- Key control: Neither U not V can predetermine any portion of the shared secret key being established.
- Identity assurance: Parties have the assurance as to the identity of the provider by bonding the identifier to the static key.

In addition to these security attributes, key establishment schemes can be compared by their performance attributes, such as a minimal number of passes, low communication overhead and low computation overhead. These performance attributes are extremely important for constrained devices such as smart phones, PDAs and other wireless devices with limited resources.

Implementing the correct scheme

In some cases, the key establishment scheme will be chosen as part of the security protocol being implemented. In other cases, you need to carefully weigh the different options and choose based on which security attributes are most desirable. The main consideration then is the primitive, either DH or the MQV algorithm, selected for key establishment and how it's computed, DLC over finite fields or ECC.

The bottom line is that you need to carefully consider key establishment schemes whenever you are designing a protocol or implementing a security system. As this article shows, there are many different key establishment schemes endorsed by the standards. So your choice comes down to carefully weighing the tradeoffs between the different security attributes and the performance attributes. Generally, key establishment using ECC and MQV yield the best results in terms of the security performance tradeoff.

MQV

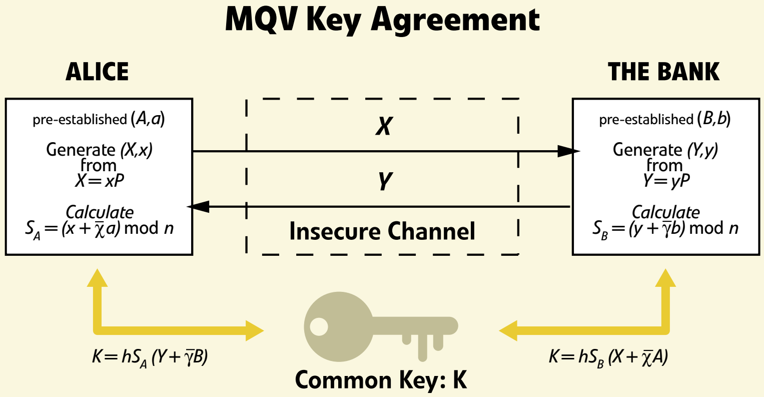
continued from page 3

MQV: Security and Performance

While based on DH, MQV offers attributes – such as key-compromise impersonation resilience and unknown key-share resilience – that are not found with DH. This allows protocols that use MQV for key agreement to offer stronger authentication and ensure malicious entities cannot masquerade as a third party to the entity whose key was compromised.

MQV also has many desirable performance attributes, including the fact that the dominant computational steps are not intensive while the protocol also has low communication overhead, is role-symmetric, non-interactive and does not use encryption or time-stamping.

The result is that MQV has all the desirable security attributes of key establishment as outlined in the article on Key Establishment Schemes and if computed using Elliptic Curve Cryptography offers significant performance advantages over other key establishment schemes. This makes it ideal in the development of security protocols and systems that require efficient and authenticated key agreement.


MQV Key Agreement
ALICE
THE BANK
pre-established (A,a)
Generate (X,x) from X = xP
Calculate $S_A = (x + \bar{\chi}a) \bmod n$
X
Y
Insecure Channel
pre-established (B,b)
Generate (Y,y) from Y = yP
Calculate $S_B = (y + \bar{\gamma}b) \bmod n$
$K = hS_A (Y + \bar{\gamma}B)$
Common Key: K
$K = hS_B (X + \bar{\chi}A)$

Alice possesses a key pair (A,a) with A being her public key and a being her private key. Similarly, the bank possesses a key pair (B,b) with B being his public key and b being its private key. Alice generates a sessional key pair (X,x) by randomly generating x and calculating $X = xP$ where x is an integer and X is a point on an elliptic curve. The bank generates a sessional key pair (Y,y) by randomly generating y and calculating $Y = yP$. P is a generating point on the elliptic curve.

Alice sends X to the bank and the bank sends Y to Alice. It is assumed that Alice already has the bank's public key B and that the bank already has Alice's public key A. The public keys have been received in some trusted manner.

Alice calculates S_A (called the implicit signature): $S_A = (x + \bar{\chi}a) \bmod n$

where n is the field size.

The bank calculates S_B in a similar manner: $S_B = (y + \bar{\gamma}b) \bmod n$.

Both Alice and the bank calculate a shared secret K: $K = hS_A (Y + \bar{\gamma}B) = hS_B (X + \bar{\chi}A)$.

where h is a co-factor defined in P1363.

Here $\bar{\chi}$ (or $\bar{\gamma}$) represents the first L bits of the first component of the point X (or Y) where $L = \left[\frac{([\log_2 n] + 1)}{2} \right]$

Figure 2: MQV Key Agreement

Code and Cipher

Code and Cipher, published quarterly by Certicom Corp., is an educational newsletter that covers the security and cryptography industry. In each issue we will examine security issues and cryptography trends in an objective manner. We welcome your thoughts, opinions and comments on anything that affects the industry. Please send us your feedback on this issue and what you'd like to see in upcoming ones:

5520 Explorer Drive, 4th Floor
Mississauga, Ontario, L4W 5L1
Canada

T (905) 507-4220
F (905) 507-4230
E codeandcipher@certicom.com

certicom