Certicom's Bulletin of Security and Cryptography

CØDE&CIPHER

This issue of Code and Cipher examines different types of certificates, their uses and benefits.

An Introduction to the Uses of ECC-based Certificates

Increased network use to communicate sensitive data and complete critical transactions has created a need for greater confidence in the identities of the person, computer, or services involved in the communication. Digital certificates and public-key cryptography provide an enhanced level of authentication and privacy to digital communications that cannot be achieved by passwords alone.

There are three key types of algorithms that are used for digital certificates: RSA, DSA and ECDSA (Elliptic Curve Digital Signature Algorithm). Other signature algorithms include ECPVS (Elliptic Curve Pintsov Vanstone Signatures), which provides partial message recovery, ECQV (Elliptic Curve Qu Vanstone): an implicit certificate type and ECNR (Elliptic Curve Nyberg Rueppel): which provides full message recovery.

To date, RSA and DSA have enjoyed wide use in applications ranging from anti-cloning to secure firmware updates. However, changing security and performance requirements, as well as the shift to smaller, more mobile devices, are pointing towards a need for smaller and faster signatures. Moreover, the National Security Agency (NSA) requires that signatures used today last for the next 50 years. These requirements pave the way for digital certificates that use Elliptic Curve Cryptography (ECC)-based signatures.

Using ECC-based signatures with digital certificates provides added size and performance advantages. This article provides an overview of what certificates are and examines how digital certificates are used in applications–applications that have wider adoption possibilities beyond the markets outlined here.

What are Digital Certificates?

Digital certificates are electronic credentials that are used to verify the identities of individuals and devices. They have one primary function: to attest that the ID data is bound to the public key. Like any physical identification, they can be created, expired, renewed, revoked or suspended. Digital certificates are typically issued by recognized organizations that can verify the certificate requestor's identity–a business, a government agency, or a bank. This ensures that once the certificate is issued, the identity of the requestor can be trusted by all who trust the authority.

Digital certificates are made up of three parts. The first part of a certificate is the identification information for the user or device. Second is a public key, associated with a private key that is kept

continues on page 3

Crypto Column by Dr. Scott Vanstone

The Practical Side of Public-Key Authentication

The authentication of the public keys used in a security protocol is one of the essential steps to ensuring the authenticity of the participants.

You may ask, "Why do you need to authenticate the public keys in a security protocol?" Without authentication, it may be possible for someone to insert themselves in a protocol and appear as a valid participant. This is the traditional man-in-the-middle attack. For example, assume Alice and Bob are using Diffie-Hellman (DH). If Eve inserts herself into the communications path and impersonates both Alice and Bob in the DH protocol, Alice and Bob will discover this only if they authenticate the public keys they receive from each other. This is because each will have received Eve's public key during the exchange rather than each other's authentic keys.

Public-key authentication schemes range from manual to fully automatic methods. Manual methods work well when a limited number of parties are involved; automated methods are generally more secure and cost effective for large distributed networks. Manual authentication usually involves an out-of-band verification process, such as telephonic verification, the use of pre-shared secrets, or public key signing "events" such as those held at IETF meetings.

As the number of users increases, it is no longer always practical to use a manual method – users may not know each other or secure communications must be established "on-the-fly". A VPN device or a secure telephone may need to establish a secure session with a previously unknown (but authorized) party. For these applications where a manual setup process is too costly or time intensive, an automated method must be used. Automated methods will rely upon certificates; and as we've seen in this issue of *Code and Cipher*, there are many forms of certificates ranging from implicit to proprietary and X.509 explicit formats.

When certificates are used, a Certificate Authority (CA) is required. However, it need not be complex or standalone. Again, depending on system criteria, the CA could be built into one device and issue certificates to the other communicating devices. For instance, a CA in a manufacturing environment may create and load a manufacturer's certificate into each device before it is shipped. At the customer site, another device with a built-in CA will communicate with all other deployed devices, validate their manufacturer's certificates, and then issue new customer-specific certificates to them. These certificates can be proprietary, if it's a closed network, or X.509 for an open network. Both proprietary explicit and implicit certificates can be very lightweight. A public CA really only needs to be used when secure communications must be established between parties who have little to no other a priori knowledge of each other.

Obviously, this column is not a complete discussion of authentication methods or certificates. There are many alternatives available for performing the critical task of public key authentication. Security requirements, level of expertise and resources available all influence the final product. The use of a simple embedded CA issuing either implicit or explicit certificates can provide high security and great efficiency. Which authentication solution you select will depend on your specific application and its projected usage requirements.

An Introduction to the Uses of ECC-based Certificates

continued from page 1

by the user. The third part of the digital certificate is the digital signature.

Size and Performance Advantages of ECC Signature Algorithms

The advantages of ECC-based certificates are two-sided: ECC-based signatures on a certificate are smaller and faster to create; and the public key that the certificate holds is smaller and more agile as well. Verification is also faster using ECC-based certificates, especially at higher key strengths. The reason can be found in the basic math behind elliptic curves. As highlighted in other issues of *Code and Cipher*, the security of ECC systems is based on the elliptic curve discrete logarithm problem, rather than the integer factorization problem. This difference allows ECC systems to start out smaller, and scale more efficiently as the bit size of the matching symmetric key increases. Ultimately, this allows for faster computations and smaller key sizes for comparable security.

Table 1 demonstrates the performance of ECDSA over RSA for signing and verifying. The ECC and RSA signatures compared at each level are of equivalent strength, despite the difference in signature size. The chart measures the number of digital signatures applied to a data block per minute.

At all levels, which are NIST (National Institute of Standards in Technology) recommended key sizes, using ECDSA provides incredible performance benefits to both signing and verification operations. The benefits of ECC-based digital certificates become even more apparent when applied to some of today's applications.

Certificates and Check 21

The *Check Clearing Act for the 21st Century* (Check 21) allows financial institutions to securely exchange image replacement documents, eliminating the paper trail associated with clearing checks and reducing overall operation and administrative costs. A more complete discussion of Check 21 can be found in *Code and Cipher* volume 1, issue 3.

Check 21 poses a security risk if images are altered at any point in the check exchange process. Digital certificates can mitigate this risk.

Each bank obtains its own certificate, which contains the corresponding public/private key pair. The private key generated during the certificate request process is securely stored and encrypted within the banks' trusted database. This private key is used to generate the signature of an image. The signature and the certificate containing the certified public key are appended to the corresponding check image. This protects the image from tampering and fraud as it progresses through the payment system.

RSA, DSA, and ECDSA are all approved as signature methods in ANSI X9.37.** Because of its relative small size, however, ECDSA provides greater efficiency and performance (especially at higher security levels). The size of ECDSA signatures also benefits the archiving process. An image captured today would need to be archived until 2012. To ensure adequate security, a 2048-bit RSA or equivalent 224-bit ECDSA signature would need to be applied. The smaller ECDSA signatures minimize overall footprint on each image, resulting in cost savings for storage.

Certificates and Digital Postal Marks

Message-recovery schemes provide the most bandwidth-efficient signatures possible. With digital postal marks signature size is very important, because the size of the signature—which is attached to the message to ensure the authentication of the source—affects the size of the mark. ECPVS is ideal for use with DPMs. As documented in volume 1, issue 3 of *Code and Cipher*, ECPVS adds as little as 20 bytes to the original message length, six times less than RSA—making ECPVS signatures more efficient.

ECNR provides the foundation for ECPVS. ECPVS improves on the efficiency of the ECNR scheme by providing partial message recovery. In this case, all or part of the message is

Table 1: Signing/Verifying Performance of ECDSA versus RSA*

ECC key size	RSA key size	Symmetric Key	ECDSA sign (sigs/min)	RSA Sign (sigs/min)	ECC Benefit (sign)	ECDSA verify (sigs/min)	RSA verify (sigs/min)	ECC Benefit (verify)
224 bit	2048 bit	3-DES (112-bit)	105840	2940	3600%	47520	26880	177%
256 bit	3072 bit	AES-128	54000	480	11250%	22800	11280	202%
384 bit	7680 bit	AES-192	30960	60	51600%	11040	2160	511%
521 bit	15360 bit	AES 256	14400	60	24000%	5280	480	1100%

** The performance was measured on Windows XP with an Intel 3.00 GHz Pentium 4 processor, and 512KB of memory, using Security Builder by Certicom toolkits.*
*** ANSI (American National Standards Institute) X9.37-2003 Specifications for an Electronic Exchange of Check & Image Data standard provides the guidance for conformance for Check 21 images.*

embedded in—and recovered from—the signature. With ECNR, the message itself can be calculated from its signature and the signing public key.

ECPVS and ECNR signatures are ideal in systems where all or part of the message needs to remain confidential as well as being small and secure.

ECC-based signature RSA-based signature

Figure 1: A comparison of physical DPM size on a piece of mail.

On February 16, 2005, the NSA unveiled their strategy and recommendations for securing U.S. government sensitive and unclassified communications. The strategy included a recommended set of advanced cryptography algorithms known as Suite B for securing sensitive but unclassified data. ECDSA was the only public key algorithm specified for authentication. ECDSA is already specified in NIST publication FIPS 186-2) and is the first ECC-based algorithm to have FIPS Validated status.

Certificates and ID cards

The new Personal Identity Verification (PIV) program defines a standard smart-card ID for all US government employees. It was put forward by NIST as FIPS 201 and includes several supporting Special Publications: SP 800-73 (PIV specification), SP 800-76 (biometrics) and SP 800-78 (cryptography). This card will carry a private key and will be able to provide the corresponding public key embedded in a digital certificate. The certificate can be signed with ECDSA or RSA for use with current systems. It is clear, however, that ECDSA will be the agencies' choice in future due to recommendations from NIST and the NSA.

This key pair will be used for physical access to government facilities; admittance will require a signed response to a challenge, and in some cases a biometric validation. Biometric data will be stored in a field in the ID information portion of the certificate and validated as part of the certificate.

The PIV will also be used for logical access to computing facilities: login to PCs, workstations, and network access. The certificate will enable single sign-on capabilities: once the certificate is validated, network resources will make use of the validation to provide access to resources such as web-services databases, secure e-mail and online forms.

Certificates and Trusted Computing

The Trusted Computing Group (TCG) initiative was established to develop a common secure computing framework. Certificates play a key role in that framework, providing the root of trust for a device. This root is used for attestation: As the hardware boots, a cryptographic hash of the system state is made, matched against known values, and then digitally signed to attest to the trusted state. From this boot process upwards, the device uses hashes to validate the hardware environment and any modifiable software or firmware, including boot code and configuration files; each is signed using the root private key.

Using these stored signed values, one system can prove to another that it has successfully passed its internal checks. The signed hashes can be sent over a network, along with the system's digital certificate, as proof that the state has been validated by a trusted secure module.

Currently under consideration within the specification, elliptic curve cryptography could be used to provide better performance to the sign and verify operations and appends less overhead to the certificate.

The future of ECC-based digital certificates

The technical merits of ECC-based digital certificates make them an excellent choice for applications, now and in the future. The strength and small size provides numerous performance and security benefits. Additionally, the longevity of the security is assured. In addition to the applications listed above, they also have a wide range of possible uses in other markets such as consumer and healthcare—for applications such as 2D bar codes, digital imaging and digital rights management. As the full effect of the U.S. Government's strategy for Crypto Modernization spreads throughout the government and into other industries, ECC-based digital certificate use will become more widespread. ■

Explaining Implicit Certificates

As devices proliferate and become more mobile, managing relationships between them will be critical. Devices will move in and out of range of each other and can interact using wireless protocols and peer-to-peer connections, taking advantage of temporary or semi-permanent secure connections to share information and access services.

Using digital certificates is considered the best-known method of establishing identity in network communications. A certificate provides a binding between identity information and a public key; a key pair can subsequently be used for key exchange to set up secured communications and for digital signatures, to validate transactions.

However, digital certificates can represent a substantial investment, both in infrastructure (to protect the keys used), memory (to store and manipulate the certificate), and bandwidth (in repeatedly transferring the certificate to various entities). Implicit certificates, known in the cryptographic community but not widely used, are smaller and faster than those in common use. Implicit or "bullet certificates", can enable a low-resource trust model for resource-constrained settings, ad-hoc networks and applications requiring printed certificates. They can also enable applications in these special situations that will not work using conventional certificates.

Consider the following applications which are well-suited to implicit certificates:

- **As the public-key scheme in ad-hoc sensor networks, which may have frequent new entrants and continually changing trust relationships. Implicit certificates provide a lightweight certificate scheme for use in these constrained environments.**
- **As "optimal mail certificates" or postage stamps. For example, the small two-dimensional machine-readable postage marks in use today could use a form of implicit certificate. This could also apply to any situation in which a very small certificate is required.**
- **As a trust scheme for peripheral devices: if a borrowed USB drive is connected to a secured PC, the PC may use a security scheme which quickly certifies the drive for temporary use. In this scenario, enrolment generates an implicit certificate between the PC and the connected device.**
- **In any situation where certificates are generally used and a lightweight certificate is preferable: e-mail systems, logical and physical access, secure online transactions; and for transactions using mobile devices. In these situations, the benefits of using an implicit certificate scheme are simple: Smaller certificates, lower bandwidth usage, optimized processing.**

What is an implicit certificate?

As with conventional or "explicit" certificates, implicit certificates are made up of three parts: identification data, a public key and a digital signature which binds the public key to the user's ID data and verifies that this binding is accepted by an authority (or trusted-third-party).

Within a conventional certificate, the public key and digital signature are distinct data elements. In contrast, the public key and digital signature are 'super imposed' in implicit certificates and allow the recipient to extract and verify the public key of the other party from the signature portion. This substantially reduces the bandwidth required as there is no need to transmit both the certificate and the verification key.

Advantage #1: Smaller

Explicit certificates are relatively large. An ECDSA-signed certificate, at an 112-bit security-strength, requires 680 bits plus the user's ID data (which is taken to be the same size in all cases). Using RSA keys at the same strength, the certificate is 4096 bits plus the ID data. A comparable implicit certificate is only 232 bits plus the ID data. *Table 2* shows certificate sizes at a few security strengths.

Advantage #2: Faster

Implicit certificates are faster than explicit certificates because deriving the public key is faster than verifying a digital signature (which is necessary in an explicit certificate). In addition, some calculations can be combined during protocol operations. There is no explicit check of the signature in using an implicit certificate; all verification is performed as part of the use of the certificate.

Table 2: Size comparison between ECC and RSA public key and certificates.

Security Level	Public key size[1] (bits)		Ratio ECC/RSA public keys	Certificate size[2] (bits)			Ratio ECQV/RSA certificates
	ECC	RSA		ECQV	ECDSA	RSA	
80	192	1024	5x smaller	193	577	2048	10x smaller
112	224	2048	9x smaller	225	673	4096	18x smaller
128	256	3072	12x smaller	257	769	6144	23x smaller
192	384	7680	20x smaller	385	1153	15360	39x smaller
256	521	15360	29x smaller	522	1564	30720	57x smaller

Data show that ECC certificates are 1-2 orders of magnitude smaller than RSA certificates, depending on security level. While ECDSA certificates are a factor 4-20 smaller than RSA certificates, ECQV implicit certificates realize another factor 3 of size reduction.

[1] *NIST-recommended key sizes.*
[2] *Data based on size of public keys and digital signatures, excluding (fixed) overhead of identification data.*

Table 3: Operational Cost Comparisons: *Conventional versus Implicit Certificates*

	Conventional Certificate		Implicit Certificate	
Action	Operation	Cost	Operation	Cost
1. Deriving the public key.	public key extraction (key included in cert)	**0**	compute public key from signature	**Elliptic-curve point multiplication**
2. Check authenticity of public keys (binding between the entity and the public key).	signature verification	**public-key operation**	no operation (delegated to Step 3)	**0**
3. Check authenticity of public keys in operation (binding between the entity and the private key).	evidenced by proper execution of protocol	**relatively expensive private-key operation** (as part of protocol)	evidenced by proper execution of protocol	**EC private-key operation (as part of protocol)**

Table 3 provides a comparison of the operational costs of a conventional certificate versus an implicit certificate.

Using an ordinary X.509 certificate, evidence of the binding between an entity and its public key is obtained during processing. Using an implicit certificate, evidence that the given (extracted) public key is genuinely bound to this entity is only corroborated by subsequent use of the corresponding private key (for example, completion of an authenticated key agreement scheme or a signing operation).

It is important to note that with an implicit certificate scheme, steps 1, 2 and 3 above can be combined during protocol operation, resulting in fewer operations than shown in the table. For example, executing the MQV protocol (described in *Code & Cipher* vol. 1, no. 2), when used with implicit certificates, requires only one computation involving elliptic curve points, during MQV computation; steps 1 and 2 are folded into step 3.

Uses For Implicit Certificates

In this time of mobile devices and wireless "motes", flat business organizations, and ad-hoc project teams, fluid trust models will become increasingly important. Interactions between mobile devices will provide valuable services; secure networks will enable interaction while maintaining logical barriers. Implicit certificates enable users and devices to establish trust relationships for ad-hoc or peer-to-peer network services.

Because implicit certificates can be issued frequently, certificate revocation is unnecessary; the CA will simply no longer refresh a user's certificate. This makes security policy more flexible. For example, if a traveling user's laptop or smart-phone is stolen, the private key may be stolen with it; but if the private key is associated with a short-lived implicit certificate (say, active for four days), the keys are only temporarily compromised.

Another advantage of using implicit certificates is flexibility in delegating authority. If different members of a team have different responsibilities, implicit certificates can be issued to these members corresponding to these different responsibilities. Only a team member that has responsibility for a specific topic can decrypt messages or documents on that topic.

Coupled with its flexibility advantages, the ECQV implicit certificate scheme detailed below provides the functional benefits of ordinary certificates, but with lower processing and storage costs, especially when used in an authenticated key agreement scheme, such as ECMQV, or in a signature scheme such as ECDSA. This makes it ideal in the development of flexible and efficient secure systems. ■

Figure 2: The ECQV Implicit Certificate Generation Scheme

The certificate requester ***A*** executes this scheme with the certificate authority ***CA*** to interactively compute an elliptic curve public-key pair with ***CA*** and obtain an implicit certificate for this public key provided by ***CA***. The ECQV scheme uses a fixed elliptic curve with generator ***G*** of order ***n***.

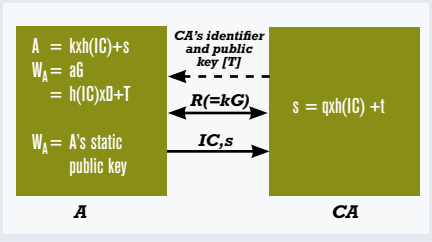
To obtain an implicit certificate:

- *A* generates a random positive integer $k < n$ and computes a point R on the curve ($R = kG$). *A* sends this point R to the certificate authority *CA*.
- Upon receipt of the point R, the certificate authority *CA* checks the authenticity of the request received from *A*.
- *CA* generates a random positive integer $q < n$ and computes a point Q on the curve ($Q = qG$). *CA* then computes the elliptic curve point D, which is the sum of the value R received from *A* and the value Q generated by *CA* itself (thus, $D = Q + R$). *CA* constructs the to-be-signed certificate data Text, which contains, as a minimum, identifying information according to established procedures of the public key infrastructure and may also contain other information, such as the intended use of the public key, the serial number of the implicit certificate, and the validity period of the implicit certificate.
- *CA* constructs the implicit certificate *IC*, which is *A*'s implicit certificate, which contains the public-key reconstruction data D and the certificate data Text in such a way that this data can be uniquely reconstructed.
- *CA* signs the implicit certificate *IC*, by first computing a hash value $h(IC)$ hereof and then calculating the implicit signature $s = h(IC)\, q + t$, where t is the *CA*'s private key.
- *CA* sends *A* both the implicit certificate *IC* and the signature s.
- *A* uses the implicit certificate *IC* and signature s to compute its private key:

  $a = s + k \times h(IC)$ (**mod** n).

- *A* may validate whether the implicit certificate *IC* indeed originated from *CA* by performing an elliptic curve computation involving the implicit certificate *IC* and signature s purportedly received from *CA*, the public-key reconstruction data D derived here from, and the public-key pair (k, R) previously generated by the device itself:

  $h(IC)\, R + s\, G = h(IC)\, D + T$.

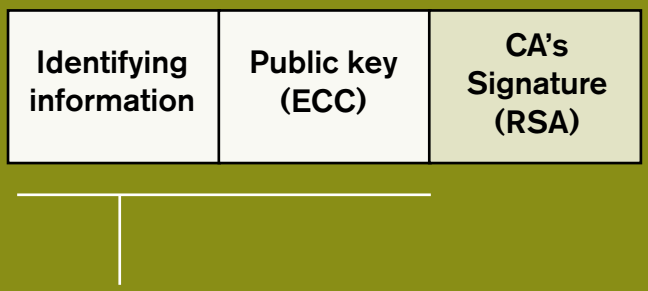


Key Distribution and Hybrid Certificates

The usual method of distributing public keys in networks of any scale is a certificate authority (CA)—a central server with a well-known public key which generates certificates for securely determining the proper public key of other elements in the network. However, existing CAs typically use RSA keys for signing and they do not necessarily have the capability to sign material using an ECC key pair—though there are some CAs that provide support for ECC. In the near future, however, CAs will need ECC certificates to implement a number of ECC-based protocols that are being submitted to the IETF (Internet Engineering Task Force) and IEEE (Institute of Electrical and Electronics Engineers).

Since CAs are such critical components in an organization's security infrastructure—tightly integrated into existing security practices and protocols—and since the distribution of the CA's public key is a significant part of making that CA useful, merely rolling out a new CA with an ECC key pair provides significant logistical hurdles.

Figure 3: Hybrid Certificates



There is a way to manage these hurdles. Traditional CAs with non-ECC public keys can still be used to distribute ECC public keys. The technology that permits this is the hybrid certificate.

A hybrid certificate is a certificate which binds an ECC public key to a communicating party, but which is itself signed using non-ECC signature algorithms, and the CA's private key. Schematically, it looks like *Figure 3*.

Hybrid certificates can be verified by any end entity which already has secure knowledge of the CA's existing public key—say, a browser using certificates to verify the identities of web servers—without the need to distribute a new public key for the CA. Using this technology, you can associate ECC public keys with any communicating party *without* having to replace your CA.

From a performance perspective, hybrid certificates do not present any disadvantages as compared to an ECC-only solution. Even though the user of a hybrid certificate has to implement and use both ECC and RSA, the only RSA operation used is signature verification, which is much faster than RSA signature generation, relatively speaking.

If the CA can be upgraded to handle ECC signatures itself, hybrid certificates can also be used to roll out the new, ECC public key for an existing CA—a technique offering a route by which the existing, non-ECC CA key may be 'grandfathered' out of use. The CA can issue certificates signed by both technologies in an interim period, and clients capable of verifying the CA's ECC signature can use the ECC signature exclusively. Eventually, if clients incapable of doing ECC verification are no longer in use, the non-ECC key can be allowed to expire, and the CA need only issue ECC-signed certificates thereafter. The benefit here is that the ECC technology can be integrated into the network gradually, as needed.

The Benefits of Hybrid Certificates

Hybrid certificates may be found to offer such potential benefits in any number of applications—a way to employ ECC and reap its benefits without entirely replacing the CA. A likely initial use is in deployments using TLS-based protocols.

Specifically, portable and constrained devices such as web-enabled phones using ECC for the handshake with a secured web server could see significant performance benefits, but they could just as well prove useful in any network using protocol families offering both ECC and non-ECC based methods of authentication and verification. ■

Crypto News

SECG Launches New Working Groups

The Standards for Efficient Cryptography Group (SECG), an industry consortium founded to develop commercial standards that facilitate the adoption of efficient cryptography and interoperability across a wide range of computing platforms, has updated some of its specifications and has recently launched two new working groups:

ECC Protocol Reference Implementations: tasked with the specification and implementation of a server hosting reference ECC protocols. This server may be used by the development community to vet their protocol implementations;

ECC Certification Authority: tasked with the specification and implementation of an X.509 certificate authority (CA) for the issuance of ECC test certificates. This CA may be used by developers in obtaining test certificates for verifying the correct operation of their products.

For more information, visit: www.secg.org ■

Weaknesses found in SHA-1

In February 2005, researchers Xiaoyun Wang, Yiqun Yin, and Hongbo Yu reported a new collision attack on SHA-1, a widely used hash function: improved collision search techniques make it possible to find collisions of SHA-1 with a reported complexity of roughly 2^{69} hash operations, considerably less than the 2^{80} theoretical bound assumed valid before. The result may have implications for the use of SHA-1 in applications that depend on its collision resistance and where an attacker is able to carry out a so-called chosen message attack. Although the new collision attack is about 2,000 times faster than the brute force method for finding collisions, it is still too slow for today's computers. Moreover, most digital signature applications include contextual information that will likely foil the attack in practice.

For a summary of the research results, visit:
http://theory.csail.mit.edu/~yiqun/shanote.pdf

NIST's comments on these development can be found here:
http://csrc.nist.gov/news-highlights/NIST-Brief-Comments-on-SHA1-attack.pdf ■

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™