Elliptic Curve Certificates and Signatures for NFC Signature Records

Research In Motion, Certicom Research

Tony Rosati, Greg Zaverucha

Abstract:

The Near Field Communication (NFC) Forum finalized its Signature Record Type Definition (RTD) to protect against manipulation of NFC Data Exchange Format (NDEF) data. The choice of digital certificate and signature type has a major impact on tag memory usage, cost and device performance. The Smart Poster RTD, gives example NDEF message sizes ranging from 23 to 69 bytes. With digital signatures and certificates this can balloon to over 1000 bytes, depending on the type of signature and certificate(s) forcing the use of larger and more expensive tags. This paper proposes further use of elliptic curve cryptography; specifically ECQV certificates and ECPVS signatures in addition to the ECDSA signature scheme. These technologies were designed with efficiency as a primary goal, and are well adapted to the constraints of NFC tags. For the same level of security, ECQV+ECPVS provides a 10 fold reduction in storage overhead compared to RSA signatures and certificates (from about 1000 to 100 bytes). Both ECQV and ECPVS are standards based, compatible with the NFC Forum Signature RTD and the ITU X.509 standard for Public Key Infrastructure (PKI). ECPVS can provide an additional confidentiality feature that allows portions of the data to be encrypted under a separate key. We introduce the reader to an NFC PKI architecture, scenarios for tag issuers, memory utilization and performance data for the various schemes specified in the Signature RTD.

Introduction

Near Field Communications (NFC) is a short-range wireless communication technology that enables data exchange between devices at a distance of a few centimeters up to a rate of approximately 400 kbps. NFC builds on the (13.56Mhz) RFID standard ISO/IEC 14443 where devices can emulate ISO/IEC 14443 smartcards, is able to power and read ISO/IEC 14443 smartcards, and can exchange information between peers. NFC enabled smart phones are now available for trial in contactless payment and ticketing applications.

The *NFC Data Exchange Format* (NDEF) [1] defines the data structures (called *NDEF Records*) for the exchange of information. The integrity and authenticity of NDEF records are critical to the success of Near Field Communications because ad hoc connections are made between untrusted devices. In the simplest case an NFC-enabled smart phone reads an NDEF record stored on an NFC tag (an RFID tag).

The *NFC Signature RTD* [10] introduces digital signatures for integrity and authenticity to NDEF records. The specification gives implementers choices for digital signatures and certificate types. These choices have a major impact on tag memory, performance and tag issuer infrastructure.

Our analysis shows that Elliptic Curve signatures (ECDSA, ECPVS) and certificates (ECQV) provide the most efficient memory utilization for constrained environments on tags and mobile devices by a wide margin over equivalent RSA based solutions. With keyed ECPVS it is also possible to provide confidentiality of the signed message between the signer and a chosen recipient.

A certificate infrastructure will have to be deployed to enable the use of signatures similar to that used for secure web browsing. As signatures are optional, an indication should be given to the user when they connect to authenticated tags or peers similar to secure web browsing. Phishing attacks can only be prevented if the user has some indication that NFC proposed actions come from an authenticated source.

Near Field Communications (NFC) on Mobile Devices

The greatest potential for NFC is the new paradigm of using proximity as context for immediately triggering an appropriate action or event without having to type in other parameters such as URLs. Simply touching an NFC-enabled object with a mobile phone can immediately trigger an action. The use of NFC promises to be more convenient and more secure than alternatives for applications including payment, ticketing, gaining access to an event, building or network. Dodson et al. [3] describes novel applications such as using an NFC enabled phone at a sporting event to gain access, run a location based application associated with the event to receive play-by-play and statistics. With the same application it is possible to order, then pay for concessions delivered to your seat.

An NFC-enabled mobile device may be in one of three possible modes of operation, as defined by the NFC Forum [4].

1. **Card Emulation mode**. The mobile device is in card emulation mode where it behaves exactly like a contactless smart card (e.g., credit and debit cards). Mobile device vendors are incorporating similar smart card chips and protocols as those used for payments to ensure at least the same security levels as existing chip based payment cards.
2. **Read/Write mode**. The mobile device is in reader/writer mode where it can, for example, read a tag (a small chip) embedded into a smart poster or magazine ad and optionally trigger an action like loading a web site for further information. A more sophisticated supply chain application could track inventory with NFC tags and possibly write updates back to the tracking tag.
3. **Peer-to-peer mode**. When the mobile device is in peer-to-peer mode it has the ability to make a connection using a different communications protocol such as Bluetooth or WiFi. For example, touching an NFC enabled phone to a printer to transfer an image to be printed over WiFi. The NFC handover protocol takes care of the configuration transparently.

There are a number of NFC Forum specifications to make this type of communication possible and highly interoperable[4].

NFC Forum Specifications

The *NFC Data Exchange Format* (NDEF) [1] defines the data structures (called *NDEF Records*) for the exchange of information. It consists of header fields and a payload field for the actual data. The payload data is formatted according to the type of information. The NFC Forum publishes a number of *Record Type Definitions* (RTDs) for well known use cases [2]. The RTD architecture is very flexible and designed to facilitate any NFC interaction. To illustrate we provide example NFC forum specifications, and the size of the NDEF records.

- The **Text RTD** [5] is for text data, and includes language information. The string "Hello, World!" is encoded in 8 bytes in the NDEF format.

- The **Uniform Resource Identifiers (URI) RTD** [6] includes web based identifiers including Internet address, Uniform Resource Locator (URL), e-mail address, telephone number, SMS Message, etc. The URL "http://www.nfc.org" is encoded in 12 bytes, a 12 digit telephone number is encoded in 17 bytes and a proprietary URI "mms://example.com/download.wmv" is encoded in 35 bytes in the NDEF format.

- The **Smart Poster RTD** [7] builds on the URI and plain text RTDs to build a complete informative record that is able to take an action such as open a web page. Two example NDEF messages are given in the specification. The first is a simple URI (http://www.nfc.org) with a total length of 23 bytes, and the second is a URI, title and an action to launch a browser, with a total length of 69 bytes.

- The **Generic Control RTD** [8] provides a way to request an action from an NFC-enabled device from another NFC device or tag. Four examples are given in the specification. The first is a generic control record which adds 500 bonus points to a customer loyalty application, encoded in 79 bytes. The second generic control record sets a mobile phone to silent mode. It is encoded in 89 bytes. The third example has multiple control records, one to turn the silent mode off and one to play music via the music player reached via a specified URI. These are encoded in 179 bytes. The final example sets the vibrate mode on with an AT command understood by most phones. It is encoded in 54 bytes. Note that the destination device is responsible for security and privacy such as prompting the user for a decision to proceed.

- The **Connection Handover RTD** [9] allows for the negotiation of an alternate communication carrier through an NFC connection. This enables some novel applications such as being able to print a document stored on a phone by simply touching it to an NFC-enabled printer. The document may be transferred using the negotiated WiFi or Bluetooth connection. The Handover RTD does not specify security for network parameters, credentials etc. The Signature RTD can be used to guarantee integrity and authenticity. Additional security access controls could be implemented but are not part of the NFC Forum standards.

- The **Signature RTD** [10] provides integrity and authenticity to NDEF messages [10]. The specification lists signature algorithms and certificate types that can be used. As we have seen, typical NDEF messages are short (8-179 bytes). Adding a signature record can dramatically increase the size of the NDEF record to hundreds and even thousands of bytes, depending on the signature algorithm and certificate type selected. We will go into much greater detail later in this paper. The use of signature records is optional. There is currently no way to provide confidentiality to NDEF messages, although this could be added in future versions of the specification, as we will see. Roland et al. [11] provides an analysis of what parts of NDEF messages should be signed and the implications from a system and security standpoint.

As shown in the preceding examples NDEF records are designed to be relatively small, on the order of 10's to 100's of bytes to enable many of the contemplated applications and to fit on low cost NFC tags.

NFC Tags

NFC tags are typically passive storage devices for NDEF records that can be read by an NFC-enabled device. The NFC Forum defines four tag types [12, 13, 14, 15]. Type 1 and 2 tags are based on ISO/IEC 14443A. These tags are read/write capable and can be configured to become read-only. Type 3 and 4 tags are more sophisticated and more costly. The Type 3 tag is based on the Japanese Industrial Standard (JIS) X 6319-4 also known as FeliCa. The Type 4 tags are fully compatible with ISO/IEC 14443A and B standards. Type 4 tags have the most memory but are also quite expensive. Table 1 gives a summary of NFC tag types, compatible products, memory sizes and relative pricing.

Table 1: NFC Tag Types

	Type 1 Tag	Type 2 Tags	Type 3 Tag	Type 4 Tag
Sample of Compatible Products	Innovision Topaz	NXP Mifare UL NXP Mifare UL-C Infineon SLE 66RxxP	Sony Felica	NXP DESFire/NXP SmartMX-JOCP Calypso B
Memory Size	96 Bytes 512 Bytes	64 Bytes 192 Bytes 2048 Bytes	1,4, 9 KB	2KB, 4KB, 8KB, 32KB
Pricing*	Low	Low	High	High

*NFC chips can be incorporated in any tag form factor such as a sticker or key fob. Prices vary dramatically depending on volumes and the final form factor. "Low" here means less than $0.50 in volume, "High" means more than $1 in volume (and in some cases more that $2 in volume) [16].

As we can see from the table above, NFC tags are very limited in memory and still quite expensive. Based on the examples shown in NFC Forun RTDs where NDEF messages are at most a few hundred bytes, signature records may only fit on larger, more expensive tags. In addition tags have low reading speed (e.g. for type A tags which is the case for NFC-Forum Type1, Type2 and Type4A) of 106 Kbit/s

gross. The net throughput may be half or less. In addition polling period of reader needs to be added. If we consider usability for reading tags should be less than 0.5 second then we should put maximum ~1,5KB information on the tag including the signature record. Before discussing signatures in greater depth it is useful to see how signatures might be deployed in the ecosystem.

NFC Signature Ecosystem

Deploying signatures for NFC is based upon a Public Key Infrastructure (PKI) where a Root Certificate Authority (CA) issues certificates for the ecosystem (Application providers and Tag issuers). The Tag issuers are enabled to sign tags for the intended application such as smart posters. Figure 1 below shows how the infrastructure might be deployed. In order for the phone to verify signature records (3) it must retrieve the root certificate (1). As in the web browser CA market there will likely be a small number of root certificates and they will most likely be installed on the phone during manufacturing or distributed by an application/service provider. Given that signatures are optional, a visual indication should be given to the user when a signature has been verified similar to web browsers. Alternatively, the user could be forced to acknowledge a warning that the tag is unsigned. Phishing attacks can only be prevented if the user has some indication that NFC proposed actions come from an authenticated source.

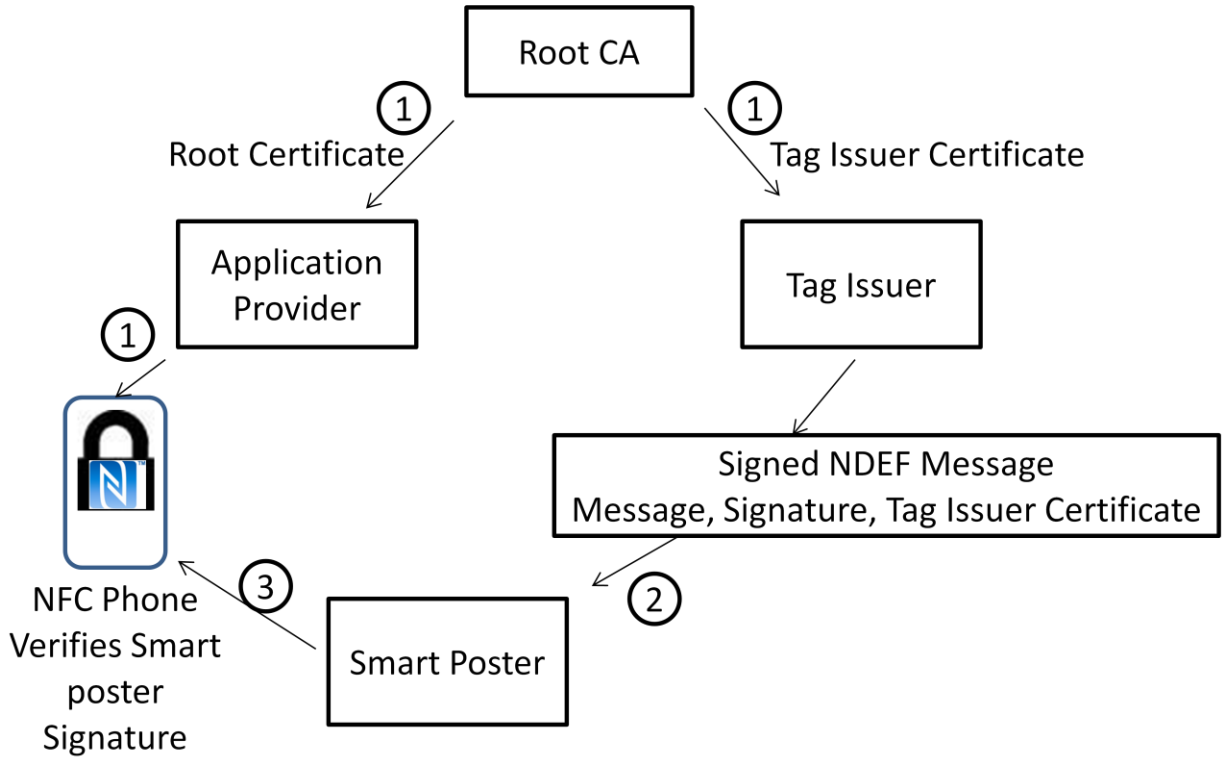

Figure 1: Signatures in Smart Posters

In more sophisticated application we can use signatures to buy tickets online. Secure (signed) tickets are sent directly to the phone and verified upon use. Figure 2 below shows how this might be deployed. The user purchases the ticket online (2). Signed tickets are stored within the NFC-enabled application on the phone and verified at ticket kiosks where they are to be used (4). Note that there is no need to store signed tickets in a secure element because they cannot be tampered with. There is a threat that the ticket could be shared(copied) but it contains identifying information of the original purchaser. Duplicates are easily detected and thus could not be easily used.

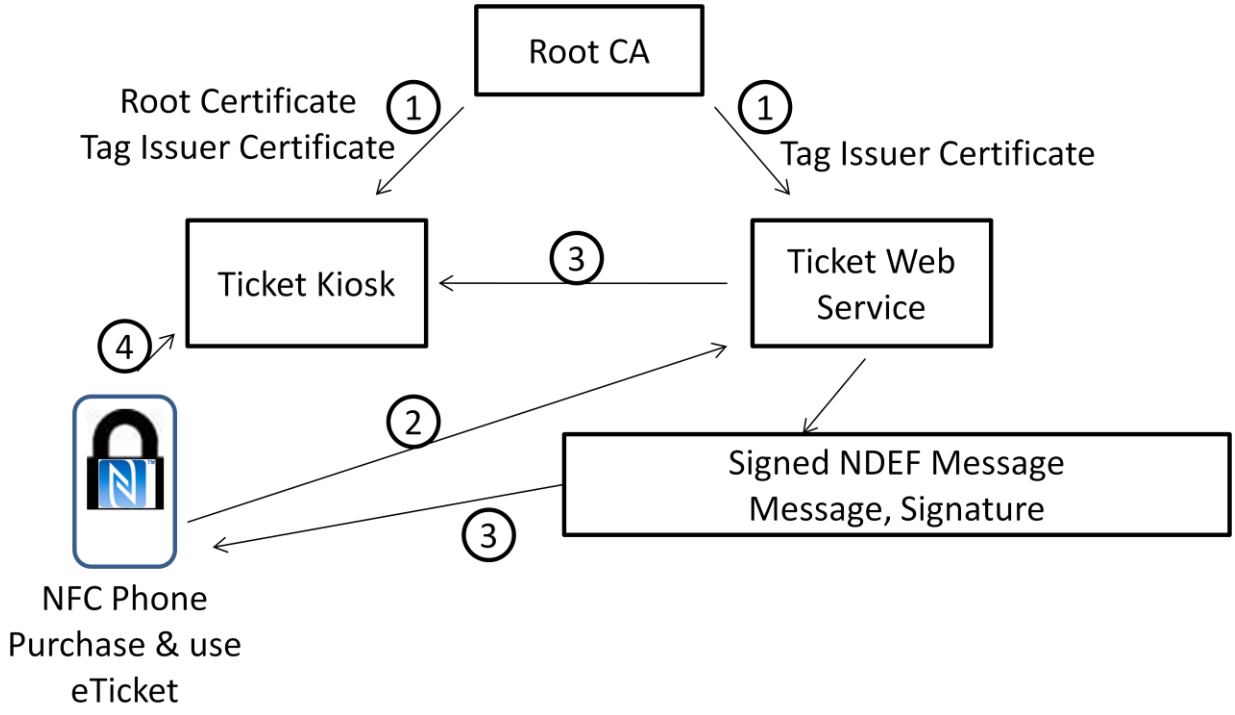

Figure 2: Signatures for Ticketing

NFC Threat Model

An NFC transaction occurs over a wireless channel between a pair of NFC endpoints (which can be readers, tags or peers). The two endpoints typically do not have any pre-shared information such as keys or certificates. The security goal is to protect this transaction from an adversary who may read, modify, delay or drop the messages sent on this channel. The adversary may also inject new messages into the transaction. Finally, the adversary may mount relay attacks to remove the need for proximity to devices he communicates with [23, 24].

In this paper, we first consider the existing mechanisms to provide message integrity and authenticity for NFC transactions, and present more efficient alternatives. This will protect communications on the wireless channel from being modified by the adversary, and also prevents the adversary from creating fake messages from one of the protocol endpoints. We then consider an extension to the Signature RTD to provide confidentiality to communication over NFC channels. This is to prevent an adversary

from being able to read messages sent on an NFC channel. Confidentiality is made possible by using a signature scheme which has additional features over the signature schemes in the existing standard.

It is important that we be able to trust the data received, especially when it may be used to trigger subsequent actions. The NFC forum specifications described above are very powerful and thus have the potential to cause significant damage without some form of protection. If we cannot verify the validity of the data, a number of attacks are possible (by malicious tags, for example). Mulliner [17] demonstrates spoofing, phishing and denial of service attacks. He also shows how NFC worms are possible when the user accepts the contents of a malicious tag. Relying on the user to approve actions may be insufficient for strong security, and create an undesirable user experience. Moreover, users can be tricked into a phishing attack. The Signature RTD was standardized to deal with these kinds of threats. Signatures can be placed on any NDEF record in any mode of operation. This also extends to subsequent events since events may be triggered automatically based on the trust established by the signature.

Confidentiality is also important if NFC is used in applications such as electronic payments and ticketing. Signatures with confidentiality may be used to protect private information in a transaction. Currently these security concerns are handled in each specific application. For example American Express, JCB, MasterCard and VISA use the EMV protocol for payments [18]. We will also give other use cases where signatures with confidentiality are desirable.

Digital Signatures for NFC

A digital signature provides integrity and authenticity for NDEF messages, and is the only security service specified by the NFC Forum. We do not want anyone to tamper with a message (integrity) and we want some sort of proof that the message was authenticated or "digitally signed" by a trusted third party (authenticity). As we will see later, some signatures can also provide confidentiality for all or part of the message, ensuring that the message may only be read by the intended recipient. Given that digital signatures can provide integrity, authenticity and confidentiality, they are extremely versatile and can accommodate almost any application scenario.

Public key cryptography is employed to implement digital signatures. Public key cryptography is based on the difficulty of solving hard mathematical problems. There are two well known schemes that have withstood the test of time. RSA is based on the problem of factoring larger integers, and elliptic curve cryptography (ECC) is based on the problem of computing discrete logarithms in elliptic curve groups. Computing elliptic curve discrete logarithms is computationally more expensive than factoring (given the current state of the art) so ECC has the advantage of smaller key sizes over RSA, for the equivalent security level.

The US government National Institute for Standards and Technology (NIST) Special Publication SP800-78-3 gives key size recommendations for the RSA and ECDSA signature algorithms [19]. Table 2 compares NIST signature algorithm requirements for Personal Identity Verification (PIV) card information to the NFC Signature RTD. The use of 1024-bit RSA for digital signatures was phased out in 2008 for Government use, and the SHA-1 hash algorithm was phased out in 2010 as attacks have shown

that SHA-1 is much weaker than its initial design [20]. The Signature RTD uses 1024-bit RSA, presumably because keys and certificates are smaller, and more widely available. However, the Signature RTD is designed to accommodate different algorithms and key sizes in the future.

Table 2: Signature Algorithm and Key Size Requirements for PIV Information and Signature RTD

Signature Generation Date	Public Key Algorithms and Key Sizes	Hash Algorithms	Padding Scheme
PIV After 12/31/2010	RSA (2048 or 3072)	SHA-256	PKCS #1 v1.5
	RSA (2048 or 3072)	SHA-256	PSS
	ECDSA (Curve P-256)	SHA-256	N/A
	ECDSA (Curve P-384)	SHA-384	N/A
NFC Forum Signature RTD (Present)	RSA (1024)	SHA-1	PKCS #1 v1.5
	RSA (1024)	SHA-1	PSS
	ECDSA (Curve P-192)	SHA-1	N/A

For signatures in the public key setting, a public and private key is generated. The private key is also called the signing key, and must be kept secret by the signer. The public key is made known to all. The public key can be used by anyone to verify that a message was signed by its corresponding private key.

A digital certificate (often simply referred to as a "certificate") binds a public key to an identity with a signature of a third party called a Certificate Authority (CA). The CA is trusted to behave honestly, in particular, to issue certificates with the correct identifying information, and only to parties who possess the private key associated to the public key in the certificate.

Given a message, signature and certificate we can verify that the message was signed by another party. The identity of the signer is established by verifying the validity of the certificate. To check the validity of a certificate, we must verify the CA's signature, using the CA's root certificate: a certificate containing the CA's public key and identity. Modern web browsers all have this capability. They have a list of root certificates for various CAs, who issue certificates to secure web sites. Any browser can verify the secure web server via a root certificate. This is part of the Transport Layer Security (TLS) protocol used by every web browser [21].

The Signature RTD defines a Signature record containing the version, signature and certificate chain.

The Signature is calculated using the following inputs:

- The message to be signed
- The private key of the signer

The Certificate contains the following values:

- Identity of the signer
- Public key of the signer
- (Optional) additional information about the signer, and certificate metadata
- Digital signature of the Certificate Authority (CA) on the above information

The X.509 certificate format (Table 3) was initially standardized by the ITU and further enhanced by the IETF and ANSI X9 is widely used in industry today[22]. The precise structure and format is defined in the RFC3280 standard (Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List Profile)[22].

Table 3: X.590 General Certificate Format

Version
Certificate Serial Number
Certificate Algorithm Identifier for Certificate Issuer's Signature
Issuer Identifier
Validity Period
Subject Identifier
Subject Public Key Information
Optional + Extension Fields
Certificate Authority Digital Signature

A certificate chain (or certificate path) is a list of certificates that is needed to verify a signature. The certificate chain terminates with the root CA certificate. For example, in the NFC ecosystem, a root CA could issue a certificate to a tag issuer who then signs NFC tags to be deployed on smart posters. In this case the verifier (NFC mobile device) needs the tag issuer certificate and root certificate in order to verify a signature on the smart poster tag. In practice root certificates are distributed to mobile devices by some alternate path. Therefore, the tag's signature record contains the tag issuer signature and certificate. It never contains the root certificate. The signature record contains the message (e.g., a URI), signature and certificate chain. Instead the signature record could contain a reference to the certificate chain but this will directly impact user experience because the mobile device will have to retrieve the certificate chain over the network. Longer certificate chains are possible, and occur when there are more levels in the certification hierarchy.

Signature Record Size

Table 4 compares RSA and ECDSA signature record sizes (excluding the overhead of 7 bytes) for different key lengths. The assumption is that the tag contains a certificate chain of length one. With RSA, the corresponding signatures and X.509 certificates are quite large. Using RSA forces implementers to use the more expensive Type 3 and Type 4 tags. In contrast, ECDSA signatures and certificates can fit on smaller tags.

Table 4: Signature Record Size Comparision (bytes)

Signature Algorithm	Signature	**Certificate Chain	Total Size
***RSA 1024/SHA-1**	128	774	902
RSA 2048/SHA-1	256	1180	1436
RSA 4096/SHA-1	512	1437	1949
***DSA 1024/SHA-1**	128	693	821
***ECDSA P-192/SHA-1**	48	388	436
ECDSA P-256/SHA-256	64	412	476

* Specified by the signature RTD

** DER Encoded X.509 Certificates (sample RSA Certificates from Verisign, ECC Certificates from Zigbee Smart Energy 2.0 Certificate Profile specification)

Signature records are typically much larger than the NDEF messages they are intended to protect. We now describe alternative techniques which provide the same security properties, but reduce the size of the signature and the certificate.

Implicit Certificates

We propose the use of the ECC-based Elliptic Curve Qu-Vanstone (ECQV) implicit certificates. Implicit certificates replace the public key (of the signer) and CA signature with a single value (called a reconstruction point in the ECQV implicit certificate scheme). They are standardized in SECG SEC4 [25] and ANSI X9.123 as a new work item. The CA's public key, and the signer's certificate (reconstruction point) are used to compute the the signer's public key, thereby authenticating it implicitly. If we use ECQV certificates to certify ECDSA-P192 signatures, then our certificate is only 56 bytes long, a substantial reduction from the 388 bytes required by an ECDSA certificate. Table 5 below specifies the fields of an ECQV certificate using fixed length encoding defined in SEC4 [25]. The certificate can be inflated in order to be compatible with X.509 ASN.1 encoding. The size would be approximately 214 bytes. ECQV certificates are fully compatible with the NFC Signature RTD.

Table 5: X.590 ECQV Certificate Format for P192

Field	Size (bytes)
Version	1
Certificate Serial Number	8

Certificate Algorithm Identifier for Certificate Issuer's Signature	1
Hash	1
Issuer Identifier	8
Validity Period	4
Subject Identifier	8
Public Key Reconstruction Value	25
Total Certificate Size	**56**

Table 6 shows the signature record size using ECDSA with ECQV implicit certificates. Using ECDSA P192 we can reduce the signature record to 104 bytes compared to RSA 1024 at 902 bytes.

Table 6: Signature Record Size ECDSA with Implicit Certificates

Signature Algorithm	Signature	Certificate Chain	Total Size
***RSA 1024/SHA-1**	128	774	902
***ECDSA P-192/SHA-1**	48	388	436
****ECDSA P-192/SHA-1/ECQV**	48	56	104
ECDSA P-256/SHA-256/ECQV	64	64	128

* Specified by the signature RTD, ** compatible with the signature RTD

Signatures Providing Confidentiality

The objective for the Signature RTD is to provide integrity and authenticity of NDEF messages, but there are signature schemes that can additionally provide confidentiality. We focus on the PV signature scheme (Pintsov-Vanstone). The PV scheme comes in two variants. The first is standard PV, or regular PV signatures. The second variant is *keyed PV*, where the signer chooses a recipient and encrypts and signs the message (or part of the message) for the recipient. This type of sign-and-encrypt operation is sometimes called signcryption. A requirement of keyed PV is that the recipient have an elliptic curve key pair, compatible with the domain parameters of the signer's key pair.

Both of the PV signature schemes have a feature called message recovery, where part of the message is embedded in one of the signature values to reduce the size of the (signature, message) pair. The message is recovered from the signature during verification using the public key of the signer. Message recovery may also be partial. In partial message recovery the message is divided into two parts, called

the hidden and visible part. The hidden part is embedded in the signature, while the visible part is sent as is, along with the signature. During verification, the hidden part is recovered from the signature. This type of signature schemes are not part of the Signature RTD in its first release, but could easily be added in the future.

In keyed PV signatures, confidentiality is provided to the hidden portion of the message. Any party can verify that the hidden and visible parts of the message were signed, but only the intended recipient may recover the hidden portion of the message, using his secret key. The scheme allows complete flexibility over which part(s) of the message are hidden, and which part(s) are visible. This is useful for applications which may require that all, some or none of the message be kept private. PV signatures are standardized in ANSI X9.92 [27], IEEE 1363a-2004 [28], ISO/IEC 9796-3-2006 [29].

There are many compelling use cases for the addition of confidentiality to NDEF messages, we list some here.

1. The military uses smart dog tags with RFID to track wounded patients solving the problem of lost or misplaced medical records [30]. The visible portion of the message could contain basic, public information about the soldier (such as name and rank), while the hidden portion contains sensitive health information that can only be accessed by authorized personnel.
2. Purchasing electronic event tickets on line, where a ticket consists of a signed message sent to the user's phone. The visible part of the message contains information the user needs (such as the date and time of the event), while the hidden part of the message is needed by the kiosk when she uses the ticket at the kiosk (such as payment information).
3. Making payments from a mobile phone. The phone produces a signature for a recipient containing payment information. The signed payment message can be cleared by a payment gateway. The hidden portion of the signature contains the user's transaction information that can only be viewed by the payment gateway.

Table 7 describes at a high level how ECPVS signatures are generated, verified and recovered. In our description we assume both parties have a common set of domain parameters, i.e., an elliptic curve group of order n, generated by a point G, a suitable key derivation function, denoted *KDF*, and hash function, denoted *HASH*, and a symmetric key encryption function, denoted ENC_k, with associated decryption function DEC_k. Complete details are left to the standards referenced above. The signer generates a key pair by choosing at random a private key d from [1, n-1], and computing the public key as $Q = dG$.

Table 7 ECPVS signature Generation, Verification and Recovery

ECPVS Signature Generation	ECPVS Verify and Recover
Inputs: • Private key of the signer: *d* • The visible message part: *V* • The recoverable message part *M* (with intrinsic redundancy) • Optional padding added to *M* Actions: • Generate a random value *k* in [1, *n*-1] • Compute *R = kG* • Compute *K = KDF(R)* • Compute $r = ENC_K(PAD(M))$ • Compute *s = k+HASH(r||V)d* (mod *n*) Outputs: • Recovery part *r* • Signature part *s* • Visible part *V*	Inputs: • Public key of the signer: *Q* • The visible message part: *V* • The recovery part: *r* • The signature part: *s* • Optional amount of padding Actions: • Compute *R = sG – HASH(r||V)Q* • Compute *K = KDF(R)* • Compute $M = UNPAD(DEC_K(r))$ ○ Includes check for padding correctness • Check intrinsic redundancy of *M* Outputs: • Recovered message part: *M* • Authenticity of the signature

The padding and redundancy checks in the verification step are required for security, since it should be difficult to create a ciphertext that decrypts to a chosen message, as an attacker can use this to create a forgery. By requiring the padded message to have sufficient redundancy, it should be infeasible to find such a ciphertext.

The amount of padding depends on the message. For instance, if we know the message is a URL that starts with "http://www.", where each character is encoded with 8 bits, we have 11*8 = 88 bits of redundancy. This is 8 more bits than enough at the 80-bit security level, but 40 bits short at the 128-bit level (so we'd need to pad with 40 bits if 128 bits of security is desired). In general the number of bits of redundancy should be equal to the security level b. So if the message has r bits of redundancy we need to pad the message with b-r bits.

For a complete security analysis of the PV signature scheme, including details of the redundancy requirement, see [32].

The size of an ECPVS signature is a function of the padded recoverable message part *r* plus the key size (since the signature value *s* is as long as the key). The signature expands depending on the size of the

recoverable part of the message *M*. For example at ECPVS P192 with a padded recoverable message part of 50 bytes has total signature length of 74 bytes.

In the keyed version of ECPVS, recall that there is a third part to the message, which may only be recovered by a chosen party, say Bob. The signer derives two symmetric keys. The first is computed in the same way above, in unkeyed ECPVS. The second key is computed $K_2 = KDF(kB)$, where k is the ephemeral random value chosen by the signer, and B is Bob's public key. This is essentially a non-interactive Diffie-Hellman key agreement between Bob and the signer, with the signer using the ephemeral key pair (k, R). The second key K_2 is used to encrypt the confidential part of the message.

The size of a keyed ECPVS signature is a function of the padded recoverable message part plus the encrypted part plus the key size. The signature expands depending on the size of the recoverable part of the message M and the encrypted part. For example, ECPVS P192 with a padded recoverable message part of 50 bytes and an additional 20 bytes for the encrypted part. The total signature is 94 bytes.

The ECPV signature schemes are straightforward to implement. They leverage the same primitives used in ECDSA.

Performance of RSA and ECC based signature schemes

Kilas [31] evaluated the performance of signatures on the Nokia 6131 NFC-enabled mobile phone for RSA, ECDSA and DSA. For the most part the results are unacceptable in terms of usability (on the order of many seconds). We ran the signature algorithms on a Blackberry Bold 9700 from standard API calls available to third party developers. The results are shown in table 8. In all cases both for signing and verification we have numbers that are very acceptable: below 10ms. It is safe to assume that NFC smart phones will be more than capable of verifying NFC digital signatures with no user experience impact.

Table 8: Signing and Verifying Digital Signatures on Blackberry Bold 9700

Signature Algorithm	Sign (ms)	Verify (ms)
***RSA 1024/SHA-1 (PKCSv15)**	9.16	0.816
RSA 2048/SHA-256	66.65	2.66
RSA 3072/SHA-256	208.74	6.20
***ECDSA P-192/SHA-1**	0.76	2.52
ECDSA P-256/SHA-256	1.34	5.63
ECDSA P-384/SHA-384	2.93	13.07

Conclusion

Digital Signatures are necessary in providing trust in the NFC ecosystem where users are expected to make wireless connections to unknown readers, tags and peers. They provide the user with a level of comfort that the data they receive has been signed by a trusted third party and more importantly, prevent a bad user experience with a malicious tag. They can also accommodate almost any application scenario including coupons and tickets.

The signature RTD gives implementers choices for digital signature and certificate types. With modern processors found on smart phones the choice of signature type does not impact performance as signing and verifying is less than 10ms. However, the choice does have a major impact on memory utilization. ECDSA uses approximately 50% less memory than RSA and can fit on most tag types. If we utilize ECDSA with ECQV certificates we use 90% less memory than RSA.

Signatures with message recovery such as ECPVS and keyed ECPVS can be used for message confidentiality where needed. If there is enough demand for confidentiality then the NFC forum can easily add this signature type to the Signature RTD given its extensible design.

Future Work

Signatures are optional for NFC. In order to prevent phishing attacks the user experience must include some indication that a tag was authenticated (or not). In this context visual indication and usability should be investigated further.

A certificate infrastructure prototype should be deployed and tested in real world applications.

References

[1]NFC Data Exchange Format (NDEF), NFC Forum Technical Specification, Rev. 1.0, Jul. 2006. http://www.nfc-forum.org/specs/spec_list/

[2] NFC Record Type Definition (RTD), NFC Forum Technical Specification, Rev. 1.0, Jul. 2006. http://www.nfc-forum.org/specs/spec_list/

[3] B. Dodson, H. Bojinov, M. S. Lam, Touch and Run with Near Field Communication (NFC), Computer Science Department Stanford University, October 2010, http://mobisocial.stanford.edu/papers/nfc.pdf

[4] http://www.nfc-forum.org/home/

[5] Text Record Type Definition, NFC Forum Technical Specification, Rev. 1.0, Jul. 2006. http://www.nfc-forum.org/specs/spec_list/

[6] URI Record Type Definition, NFC Forum Technical Specification, Rev. 1.0, Jul. 2006. http://www.nfc-forum.org/specs/spec_list/

[7] Smart Poster Record Type Definition, NFC Forum Technical Specification, Rev. 1.0, Jul. 2006. http://www.nfc-forum.org/specs/spec_list/

[8] Generic Control Record Type Definition, NFC Forum Technical Specification, Rev. 1.0, Mar. 2008. http://www.nfc-forum.org/specs/spec_list/

[9] Connection Handover, NFC Forum Technical Specification, Rev. 1.2, Jul. 2010, http://www.nfc-forum.org/specs/spec_list/

[10] Signature Record Type Definition, NFC Forum Technical Specification, Rev. 1.0, Nov. 2010, http://www.nfc-forum.org/specs/spec_list/

[11]M. Roland, J. Langer , Digital Signature Records for the NFC Data Exchange Format, Proceedings from the Second International Workshop on Near Field Communication, NFC 2010.

[12] Type 1 Tag operation Specification, NFC Forum Technical Specification, Rev. 1.0, Jul. 2007, http://www.nfc-forum.org/specs/spec_list/

[13] Type 2 Tag operation Specification, NFC Forum Technical Specification, Rev. 1.0, Jul. 2007, http://www.nfc-forum.org/specs/spec_list/

[14] Type 3 Tag operation Specification, NFC Forum Technical Specification, Rev. 1.0, Aug. 2007, http://www.nfc-forum.org/specs/spec_list/

[15] Type 4 Tag operation Specification, NFC Forum Technical Specification, Rev. 2.0, Nov. 2010, http://www.nfc-forum.org/specs/spec_list/

[16]http://www.idcardmarket.com/home.html Sample pricing

[17]C. Mulliner, "Vulnerability Analysis and Attacks on NFC-enabled Mobile Phones," in Proceedings of the International Conference on Availability, Reliability and Security, Mar. 2009

[18] EMVCo, LLC. EMV Contactless Communication Protocol Specification, July 2009. Version 2.0.1. http://www.emvco.com/specifications.aspx

[19] T. Polk, D. Dodson, W. Burr, D. Cooper, NIST Special Publication 800-78-3, Cryptographic Algorithms and Key Sizes for Personal Identity Verification, November 2010

[20] Dealing with weakness in SHA-1, http://lwn.net/Articles/337745/

[21] T. Dierks, E. Rescorla, IETF RFC5246: The Transport Layer Security (TLS) prptocol, Version .12, http://tools.ietf.org/html/rfc5246

[22] R. Housley,W. Polk,W. Ford, and D. Solo. RFC3280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. IETF, April 2002. http://www.ietf.org/rfc/rfc3280.txt.

[23] L. Francis, G. P. Hancke, K. Mayes, and K. Markantonakis. Practical NFC Peer-to-Peer Relay Attack using Mobile Phones. In Workshop on RFID Security – RFIDSec'10, Istanbul, Turkey, June 2010.

[24] M. Weib, Performing Relay Attacks on ISO 14443 Contactless Smart Cards using NFC Mobile Equipment, Master's Thesis in Computer Science, University of Munich, May 2010.

[25] Standards for efficient cryptography group (SECG), SEC4 Elliptic Curve Qu-Vanstone implicit Certificate Scheme (ECQV) v0.91, http://www.secg.org/index.php?action=secg,docs_draft

[26] Standards for efficient cryptography group (SECG), SEC1 Elliptic Curve Cryptography version 2.0, Sections 2.3.3 and 2.3.4, http://www.secg.org/index.php?action=secg,docs_secg

[27] ANSI Draft X9.92-2007-02-21: Public Key Cryptography for the Financial Services Industry, Digital Signature Algorithms Giving Partial Message Recovery Part 1: Elliptic Curve Pintsov-Vanstone Signatures (ECPVS), Accredited Standards Committee X9, Inc., 2007.
[28] IEEE P1363a / D2, Standard Specifications for Public Key Cryptography: Pintsov-Vanstone Signatures with Message Recovery, 10 January 2000.

[29] ISO/IEC 9796-3:2006: Information technology -- Security techniques -- Digital signature schemes giving message recovery -- Part 3: Discrete logarithm based mechanisms, 2006

[30]David C. Wyld, "The Bottom-Line is Survivability: Enhancing Military Medicine With Automatic Identification of Casualties", http://www.bukisa.com/articles/372603_the-bottom-line-is-survivability-enhancing-military-medicine-with-automatic-identification-of-casualties#ixzz1ArBSMxAT

[31] M. Kilas, Digital Signatures on NFC Tags, Masters of Science Thesis, 2009, web.it.kth.se/~johanmon/theses/**kil**as.pdf

[32] D.R.L. Brown and D.B. Johnson. Formal Security Proofs for a Signature Scheme with Partial Message Recovery. Proceedings of CT-RSA 2001, LNCS 2020, 126-142.