ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 1, *January 2015*

Signing of short messages

Suvendu Sekhar Dash[1], Sibo Prasad Patro[2], G.Sambasiba Rao[3]

Lecturer, Department of IT, G.I.E.T, Gunupur, India[1,2]

HOD, Department of IT, G.I.E.T, Gunupur, India[3]

Abstract: Now a day in the present scenario in network security, mainly we are facing problem in signing for short messages. To overcome this problem we are proposing a scheme that minimizes the total length of the original message and also the appended signature. This research was motivated by several Govt. services interested by stamping machines capable of producing digital signatures. Although we have so many message recovery schemes, but their security measurements are questionable. Our paper proposes several variants of DSA AND ECDSA allowing partial recovery. Mainly the signature part is appended to a truncated message and the discarded bytes are recorded by the verification algorithm up to the signature authenticates the whole messages. Our scheme has some form of security based on random oracle model. By using further optimization techniques we can lower the schemes overhead to 26 bytes for a 2^{-80} security level where we can compare to both 40 bytes for DSA or 128 bytes for RSA.

Keywords: Cryptography: Encryption, Decryption

I. INTRODUCTION

Thirty years so far the discovers of public key cryptography and digital signatures, the world appears ready for large scale usage. So far several signature schemes have been designed by the researches. But all are based on RSA algorithm or discrete algorithm. But in our scheme we are preparing random oracle model.

In some situations It is desirable to use very short signatures, with more accuracy, one wishes to minimize the total length of the original message and the appended signature also the motivation for short signatures has assign from the needs of various banking , postal , income-tax, LIC service which are currently investigating the possibilities of integrating digital signatures into stamping machines. If we use limited space, it will helps in designing low-cost barcode printing machines and optical readers.

II. RELATED WORK

1-D Barcodes are alternating patterns of light and dark that encode specific information chunks. When scanned barcodes can be converted back into the original string of text. Barcodes can be scanned on the fly with little or no error under less than ideal conditions. (i.g failed or damaged items). The scanners that read barcodes emit a laser beam of a specific frequency that works by distinguishing the edges with in a symbol allowing them to be scanned in one-direction. Each symbology has unique start and stop has that allows scanner to discriminate between symobolgies without human intervention.

A 2-D code stores information along the weight as well as the length of the symbol since both dimension contain information at least some of the vertical redundancy is lost and error-correction techniques must be used to prevent misreads and produce acceptable read rates. The 2-D symbol can be read with hand held moving beam scanners by sweeping the horizontal beam down the symbol. Now a day's 2-D symbologies are using in health care industry, electronic industries etc. there are well over thirty different 2-D sysmoblgies available today. The reader can get a better idea of this diversity of consulting [2]. More recently the ability to encode a portable database has made 2-D symobologies attractive in postal applications. Mainly in storing name, address, business replay card0073.If the replay card is only coded with a serial number, the few replies must be checked again a very large database, perhaps millions of names. This can be quite expensive in computer time. If all the important information is printed in 2-D code at the time, the mailing label is printed with very little additional cost.

III. PROPOSED ALGORITHM

NYBERG-RAPPEL SIGNATURE

We can say that a signature scheme allows a message recovery if the message "hello" is a deterministic function of the signature. Such signature makes it possible to avoid sending the message together with the signature. However one should be very careful since such schemes are inherently subset to for series.A DSA-like signature with message recovery has been considered by Nyberg and Rappel and an ECDSA variant of this scheme described.

Signature

1. Generate a random key pair $\{u,V\}$
2. Form f from m by adding the proper redundancy
3. Encode V as an integer i
4. $C \leftarrow i+f \bmod r$
5. If $c=0$ go to step 1
6. $d \leftarrow u-sc \bmod r$
7. output the pair $\{c,d\}$ as the signature

Verification

1. input a signature $\{c,d\}$
2. if $c \notin [1,r-1]$ or $d \notin [1,r-1]$,output invalid and stop
3. $P \leftarrow d.G+c.W$
4. If $P=0$, output invalid and stop
5. Encode P as an integer i
6. $f \leftarrow c-i \bmod r$
7. if the redundancy of f is incorrect output invalid and stop
8. output valid and the underlying message m

In the above, *f* is a message with appendix. It simply means that it has an adequate redundancy. The encoding mentioned in step 3 is defined in the standard. Its particular format is not important to us. Applying a has function to this encoding consists of replacing step 3 by: "3. Encode-and-hash V as an inter i".

Our proposal allows to sign a message m=m1||m2, where || denotes concatenation and to only transmit m2 together with the signature, the partial message recovery concept is, of course, not new; the RSA-oriented ISO 9796-2 standard [7] specifies explicitly two recovery modes (total and partial) but to the best of our knowledge, this notation was never extended to the DLP context, we propose to sign m, using the algorithm described in the following algorithm where H denotes any standard has function such as SHA-1

Signature

```
1.  Generate a random key pair {u,V}
2.  Form f1 from m1 by adding the proper redundancy
3.  Encode and hash V as an integer i
4.  C←i+$f_1$mod r
5.  If c=0 go to step 1
6.  $f_2$←H($m_2$),d←$u^{-1}$($f_2$+sc) mod r
7.  if d=0 go to step 1
8.  output the pair {c,d} as the signature
```

Verification

```
1.  input a signature {c,d} and a partial message m2
2.  if c ∉ [1,r-1] or d ∉ [1,r-1],output invalid and stop
3.  $f_2$←H($m_2$),h←$d^{-1}$ mod r ,$h_1$←$f_2$h mod r
4.  $h_2$←ch mod r, P←$h_1$.G+$h_2$.W
5.  If P=0, output invalid and stop
6.  Encode-and-hash P as an integer i
7.  $f_1$ ← c-i mod r
8.  if the redundancy of $f_1$ is incorrect output invalid and stop
9.  output valid and the underlying message $m_1$
```

Truncating d

We now turn to the second optimization suggested above. It consists in truncating k signature bytes. For example, one could omit the k trailing (or leading) bytes of c. this basically means issuing 28k candidate signatures. The correct signature is spotted at signature verification: only the correct choice is accepted by the verification algorithm.

It is easily seen that the security of the truncated signature is closely related to the original scheme. At attacker able to forge a truncated Signature will complete his forgery to an actual signature by using the verification algorithm. Thus, the only difference is the verifier's workload.

Signature

```
1.  Generate a random key pair {u,V}
2.  Discard the l trailing bits $m_1$
3.  Form $f_1$ from $m_1$ by adding the proper redundancy
4.  Encode and hash V as an integer i
5.  C←i+$f_1$ mod r
6.  If c=0 or i ≠$m_1$ mod $2^{8l}$ go to step 1
7.  $f_2$←H($m_2$),d←$u^{-1}$($f_2$+sc) mod r
8.  if d=0 go to step 1
9.  output the pair {c,d} as the signature
```

Verification

```
1.  input a signature {c,d} and a partial message m2
2.  if c ∉ [1,r-1] or d ∉ [1,r-1],output invalid and stop
3.  $f_2$←H($m_2$),h←$d^{-1}$ mod r ,$h_1$←$f_2$h mod r
4.  $h_2$←ch mod r, P←$h_1$.G+$h_2$.W
5.  If P=0, output invalid and stop
6.  Encode-and-hash P as an integer i
7.  $f_1$ ← c-i mod r
8.  if the redundancy of $f_1$ is incorrect output invalid and stop
9.  append to $m_1$' the l trailing bytes of i
10. output valid and the underlying message $m_1$
```

At first glance, it seems that, in order to check truncated signature, the verifier will have to verify 28k signatures, which appears prohibitive even for k=1. However, optimizations are possible since the various elliptic curve points that the verifier should compute are

$$P=h1.G+h2.W$$

Where only h2=ce-1 mor r depends on c. Let c0 can be completion of the truncated value of c by zeros. Writing p as

$$Pj=h1.G+c_od-1.W+jd-1.W$$

We see that the verification algorithm can be organized as follows:

```
1.  z←d-1.w
2.  p←p0+c0.Z
3.  while a correct signature has not been found P←P+Z
```

considering that c, d are 160 bit integers and that a standard double-and-add algorithm is used, one can estimate the number of elliptic curve operations needed to compute P0 as close to 240. Z and P0 can be simultaneously computer in about 320 additions by sharing the "double" part. Finally, step 3 is expected to require 128 extra additions. For K=1, the overhead does not exceed the verification time of a regular signature.

There is trick which slightly improves performances: instead of using the signature {c,d}, one can use{h2,d}, with h2=cd-1 mod h. truncating h2 yields slightly better computational estimates.

Securities Proof

We use the random oracle model to provide evidence in favor of the security of the new scheme. We will thus assume that the function R(V) which encodes the point v as an integer I and computes i mod r is random. Finally, we will assume that the probability £ that a random element *f* of [0, r -1] has the expected redundancy is very small. Basically, we want to show that an adversary who can forge a message / signature pair with probability £ + α significantly above £ can be used to solve the ECDL problem with non-negligible probabilistic however we will not be careful about the security estimates for we only wish to support the correctness of our design.

Adaptive Attacks

We show how to modify the security proof that was just given to cover the adaptive case. We have to explain how to turn the attacker in to a machine that discloses the logarithm of a given element W in case G.

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 1, *January 2015*

To simulate the signer when he has to output the signature of a message m=m1||m2, we pick the signature {c,d} at random query the H-Oracle at m2 and compute the point.
$V=(f_2d^{-1})G+(cd^{-1})w$
With $f_2=H(m_2)$

Band width optimizations
We now investigate possible optimizations of our scheme so that allow to save a few extra bytes. We use 2 different tricks Transmitting additional message bytes as a sub minimal part of the signature by suitable choosing the random part during signature generation. Truncating the signature, leaving completion to be performed during the verification phase.

Packing bytes into i
Assume that one wishes to embed l bytes of m in I, where l is a small integer for example, assume that we try to stuff these bytes into the trailing part of I. one would then repeat the first steps of the signature generation algorithm until a correct value of I appears, i.e an I whose trailing bytes match the given l bytes of the message clearly, this is possible only if l is small and yields the scheme presented in figure 6 that allows to sign a message m=m1||m2, where m1 has 10 + l bytes and to only transmit m2. The security proof of section through, word for word, for the modified scheme.

Note that preprocessing appears very helpful here. Basically, one should store pairs {u,i} and access these pairs by the value of I mod 28l. Signature generation might fail if the table's list of elements is empty at some l byte locations. Thus, it is important to keep a sufficiently large number t of elements for each l byte values and to refresh the table regularly.

The size of the table is ≈ 40t28l bytes; l=3 corresponds to 640t Mbytes which is quite acceptable; l=4 goes up to 160t Gbytes, which appears too much. Note that l is not necessarily an integer: bytes can be cut into nibbles and l=3.5 could also be considered (10t Gbytes).

IV. CONCLUSION

We have shown how to minimize the overall length of an elliptic curve signature i.e the sum of the lengths of the signature itself and of the message (or part of the message) that has to be sent together with the signature. Up to thirteen message bytes can be recovered in a secure way from a signature and an additional one-byte saving on the signature itself can be obtained.
The proposed schemes have been validated by a proof in the random oracle model and can therefore be considered sound. All our schemes have ordinary discrete logarithm analogs.

REFERENCES

1. M. Abe and T. Okamoto, A signature scheme with message recovery as secure as discrete logarrithms, Proceedings of asiacrypt'99, LNCS, Springer-Verlag, to appear, 1999.
2. http://www.adams1.com/pub/russadam/stack.html
3. http://www.azalea.com
4. M. Bellare and P. Rogaway, Random oracles are practical: a paradigm for designing e_cient protocols, Proceedings of the 1-st ACM conference on communications and computer security, pp. 62{73, 1993.
5. M. Bellare and P. Rogaway, The exact security of digital signatures – How to sign with RSA and Rabin, Proceedings of eurocrypt'96, LNCS 950, Springer-Verlag, pp. 399{416, 1996.
6. D. Coppersmith, S. Halevi and C. Jutla, iso 9796-1 and the new forgery strategy., manuscript, July 28, 1999.
7. J.-S. Coron, D. Naccache and J.P. Stern, On the security of RSA padding, Proceedings of crypto'99, LNCS 1666, Springer-Verlag, pp. 1{18, 1999.