Efficient Protocols for Signing Routing Messages

Kan Zhang
Cambridge University Computer Laboratory
Pembroke Street, Cambridge CB2 3QG, UK
Email kz200@cl.cam.ac.uk

Abstract

In this work, we aim to reduce the computational costs of using public-key digital signatures in securing routing protocols. Two protocols (COSP and IOSP) using one-time digital signatures are introduced to provide the functionality of public-key digital signatures. Our protocols are intended to be used in place of public-key digital signatures for signing all kinds of message exchanges among routers. We obtained more than ten-fold increase in speed compared with public-key signatures. Our protocols overcome the shortcomings identified in previous works, such as timing constraints, limited applications and high storage and computational costs for volatile environments [12].

Since our protocols are non-interactive, they provide full functionality of a true signature. However, our protocols are not intended for replacing public-key infrastructures completely. Instead, public-key infrastructures are used to set up COSP and IOSP. As a general approach, our protocols can be used with public-key cryptosystems for efficient message signing in much the same way as secret-key cryptosystems are used in conjunction with public-key systems for efficient data encryption.

1. Introduction

Routing protocols distribute information regarding the topology of network among the routers in the network. Disseminating routing information reliably is essential to Internet routing protocols. The routing information each router receives from others serves as the basis for forwarding packets from their source to their destination. Without accurate routing information, packet transmission through the network is at best inefficient and at worst may fail completely.

The routing protocols that operate in the Internet are all subject to certain sorts of attacks. Because routers function cooperatively based on the routing information they receive from their peer routers, they are all threatened by the possibility that routing information might be replayed, or that new bogus routing information might be generated and inserted into the communication.

Presently, many Internet routing protocols (RIP [23], OSPF [28], ISIS [15], IDRP [16]) reserve fields in the packet format for authentication use. However, the strongest authentication mechanism defined for these fields in some protocols (RIP, OSPF, ISIS) is clear-text passwords. The sniffer attacks demonstrate that clear-text passwords are not strong enough protection. Cryptographic protection of source authenticity and message integrity provides stronger protection.

Considerable work has been done to secure various routing protocols [30, 37, 17, 10, 24, 36, 18, 35]. In many of these approaches, public-key digital signatures are used to provide authenticity and integrity of routing messages. Using digital signatures by itself does not protect against the internal threat of a faulty router. However, it does protect routing information against faulty intermediate routers as well as external intruders. In short, digital signature helps to achieve [36]:

- Byzantine robustness to faults in non-routing nodes in an internet,
- Byzantine robustness to faults in routing nodes concerning links not incident on those nodes, and
- simple robustness of all other faults in routing nodes.

However, public-key digital signatures can be costly. Generating and verifying public-key digital signatures are time-consuming. In routing protocol context, message exchange among routers happens very frequently and digital signatures have to be generated and verified in real time. In link-state routing protocols, verification of digital signatures can be more of a problem, since each message signature is verified by a large number of routers. When links and nodes fluctuate, the pain is particularly felt since routers have to update their routing tables in a timely fashion.

In this paper, we propose some protocols that can substantially reduce the computational costs of using digital signatures. Our approaches use one-time digital signatures

based on one-way hash functions. The paper is organized as follows. In Section 2, we give some introduction to one-time signature schemes. Previous work is briefly discussed in Section 3. Two new protocols, COSP and IOSP, are presented in Section 4 and 5, respectively. Performance issues, storage requirements and comparisons of these two protocols are addressed in the following 3 Sections. The applicability of our protocols as efficient alternatives to public-key digital signatures is discussed in Section 9. Finally, we conclude in Section 10.

2. One-time Signature Schemes

One-time signature schemes are based on a public function f that is easy to compute but computationally infeasible to invert, for suitable definitions of "easy" and "infeasible". Such functions are called one-way functions (OWF) and were first employed for use in login procedures by Needham [39]. If the output of a one-way function is of fixed length, it is called a one-way hash function (OWHF). More precisely, the definition of OWHF is given as [3]:
Definition A function h that maps bit strings, either of an arbitrary length or a predetermined length, to strings of a fixed length is a OWHF if it satisfies three additional properties:

- Given x, it is easy to compute $h(x)$
- Given $h(x)$, it is hard to compute x
- It is hard to find two values x and y such that $h(x) = h(y)$, but $x \neq y$.

Some authors describe the third property as *collision-resistance*.

One-time signature scheme was first introduced by Lamport [19, 8]. For signing a single bit, choose as the secret key two values x_1 and x_2 (representing '0' and '1') at random and publish their images under a one-way function $y_1 = f(x_1)$ and $y_2 = f(x_2)$ as the public key. These x's and y's are called *secret key components* and *public key components*, respectively. To sign a single bit message, reveal the pre-image corresponding to the actual '0' or '1'. For signing longer messages, several instances of this scheme can be used.

Motivated by Lamport's approach, many researchers have subsequently proposed more efficient one-time signature schemes. Merkle [26, 25] proposed an improvement which reduces the number of public key components in the Lamport method by almost two-fold. Instead of generating two x's and two y's for each bit of the message, the signer can generate only one x and one y for each bit of the message to be signed. When one of the bits in the message to be signed is a '1', the signer releases the corresponding value of x; but when the bit to be signed is a '0', the signer releases nothing. Because this allows the receiver to pretend that he did not receive some of the x's, and therefore to pretend that some of the '1' bits in the signed message were '0', the signer must also sign count of the '0' bits in the message. Now, when the receiver pretends that a '1' bit was actually a '0' bit, he must also increase the value of the count field, which can't be done. Because the count field has only $\log_2 n$ bits in it, the signature size is decreased by almost a factor of two, i.e., from $2n$ to $n + \lfloor \log_2 n \rfloor + 1$.

As an example, if we wished to sign the 8-bit message '0100 1110' we would first count the number of '0' bits (there are 4) and then append a 3-bit count field (with the value 4) to the original 8-bit message producing the 11-bit message '0100 1110 100' which we would sign by releasing $x[2]$, $x[5]$, $x[6]$, $x[7]$ and $x[9]$. The receiver cannot pretend that he did not receive $x[2]$, because the resulting erroneous message '0000 1110 100' would have 5 '0's in it, not 4. Similarly, pretending he did not receive $x[9]$ would produce the erroneous message '0100 1110 000' in which the count field indicates that there should be no '0's at all. There is no combination of x's that the receiver could pretend not to have received that would let him concoct a legitimate message.

Winternitz [25] proposed an improvement which reduces the signature size by several folds at the expense of increased computational effort. In Winternitz's method, the OWF is applied to two secret key components iteratively for a fixed number of times, resulting in a two-component public key. Meyer and Matyas [27] proposed as a further improvement to use more than two chains of function evaluations. Their scheme was generalized further in [9] and later in [38] to a scheme with l chains of length k where the signatures consist of one node in each chain such that the total sum of the levels of these nodes (within their chains) is constant.

The schemes described so far can only be used to sign a single message. Merkle [25, 26] proposed so-called tree-authentication schemes for signing several messages consecutively with a single public key. Multiple keys can be put on the leaves of an authentication tree and authenticated through the paths of the tree. Vaudenay [38] showed an improved scheme requiring less memory. Bleichenbacher and Maurer [5, 6, 7] formalized the concept of one-time signatures using directed acyclic graphs and proved some theoretical results. These schemes for signing multiple messages use a single public key for verification, a feature resembles public-key cryptography. However, it is achieved at higher computational costs.

3. Related Work

Hauser et al. [12] have identified the high costs of using public-key digital signatures in securing link-state routing protocols and proposed two techniques for efficient and secure processing of link state updates. The first technique (SLS) is geared towards a relatively stable internetwork environment using a single chain of hashes as authentication tokens. Each hash can be seen as a one-time signature for one bit of information, i.e., no link adjacent to the signing router changes state. In order to sign more than one bit of information, a second technique (FLS) has been presented using a set of hash chains with every pair of them represent the status, i.e. UP and DOWN respectively, of a single link. These hash chains can be seen as one-time signatures of the status of each link.

As have been noted by the authors (first 3 points), their techniques are limited in the following aspects:

- Very frequent state changes
  If the environment is such that link and node outages occur very often, FLS becomes unworkable since hash chains are used in agreement with pre-set time intervals. The pre-set time intervals are needed to guard against delay-and-forge attack by intermediate malicious routers. (We will say more about it later.)

- Clock drifts
  If the clocks of routers are not synchronized within the window of the pre-set time interval, the above delay-and-forge becomes possible even if the hashes are sent out at pre-set time intervals, since there is no guarantee of timeliness/freshness of signatures.

- Multiple-valued link costs
  In FLS, link state is assumed to be binary: UP or DOWN, i.e., the cost of a link is a fixed number. If multi-valued link costs are supported, the number of hash chains has to grow by a factor of m - the number of all possible values of link costs. Considering the increased storage and computation costs, it may be more efficient to resort to public-key digital signatures.

- Large or changing number of links
  In FLS, the signature size is proportional to the number of links. If the number of links is large, the storage requirements might render the scheme infeasible. If the number of links changes, the number of hash chains has to change, too. So does the signature size.

- Applicability to other routing messages
  Their approaches can only be used as signatures for Link State Updates (LSUs), since the meaning of the hash chains is fixed. They cannot be used for arbitrary messages.

In this paper, we present protocols based on one-time signatures that overcome these deficiencies. Our protocols are non-interactive and provide the full functionality of a *true signature*.

4. Chained One-time Signature Protocol

Chained One-time Signature Protocol (COSP) overcomes the last 3 shortcomings of FLS listed above. Instead of signing the status of each link, in COSP we sign the whole routing message. The signature size is fixed irrespective of the message content.

We assume there is a public-key infrastructure available as used in previous works, e.g., [30, 24, 12]. However, in order to reduce the high computational costs, the public-key infrastructure is not used to sign routing messages directly as in previous approaches. Instead we use the the public-key infrastructure to set up COSP.

Let M_i be the ith routing message to be sent and two hash functions f and h are agreed upon by all parties. Hash function f is applied to the message M_i to obtain its hash $f(M_i)$. This hash value $f(M_i)$ is to be signed to provide authenticity and integrity of message M_i. Suppose the output of hash function f is l-bit long. Using Merkle's scheme [25], we need $n(= l + \lfloor \log_2 l \rfloor + 1)$ one-time public key components to sign $f(M_i)$.

In order to sign more than one message, we need multiple sets of these one-time public key components. However, if we keep a large number of these sets, the storage requirements may become prohibitive, since each router has to keep a copy of these sets from every other router in link state routing. Instead, we derive multiple sets of public key components from hash chains by repeated hashing of the public key components in the first set. Hence, the sets of one-time public key components and secret key components are inter-linked, i.e., the ith set of secret key components are used as the $(i+1)$th set of public key components.

The forming of a hash chain $h^1(x), \cdots, h^i(x), \cdots, h^n(x)$ of length n from a bit string x is as follows: $h^0(x) = x$ and $h^i(x) = h(h^{i-1}(x))$ for $i = 1, \cdots, n$, where h can be, for example, MD5 or SHA. Hash chains have been widely used in a number of applications, such as one-time authentication [20, 11], micro-payment [1, 13, 29, 32], conditional anonymity [14], non-repudiation [2] and certificate revocation [22].

Suppose we want to be able to sign k messages for each Setup. Our basic protocol goes as follows:

- Setup

  1. Each router chooses at random as secret key components x_j, $j = 1, ..., n$.

2. Each router prepares a table of n hash chains of length k:

$$\begin{array}{cccccc} 0 & h^0(x_1), & h^0(x_2), & \cdots, & h^0(x_n) \\ 1 & h^1(x_1), & h^1(x_2), & \cdots, & h^1(x_n) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ k & h^k(x_1), & h^k(x_2), & \cdots, & h^k(x_n) \end{array}$$

3. Each router broadcasts the kth row of his table signed using public-key cryptography.
4. Each router verifies the received h^k's from other routers are good using public-key cryptography and stores them as v_j, $j = 1, \cdots, n$. These v_j's are the one-time public key components of the corresponding router.

- Signing the ith message M_i
  1. Obtain a n-bit binary string g by concatenating $f(M_i)$ with a count field using Merkle's method as explained above.
  2. Form the one-time signature by concatenating the hash values $h^{k-i}(x_j)$ in the $(k-i)$th row of the table for all j such that $g_j = 1$, where g_j is the jth bit of string g.
- Verification
  1. Obtain the n-bit binary string g by concatenating $f(M_i)$ with a count field using Merkle's method as explained above.
  2. For all j such that $g_j = 1$, check if
  $$h^{i-i'}(r_j) = v_j, \qquad (1)$$
  where r_j and v_j are the received and stored value for the jth bit, respectively, and v_j is last updated for message i'.
  3. If true, accept the message and update v_j with value r_j so that when he evaluates Eq. (1) for message $i'' > i$ in the future he only needs to perform $i'' - i$ hash computations.

Using COSP, we are able to sign arbitrary messages for a pre-determined number of times. However, the routers have to sign messages at fixed time interval T and their clocks have to be synchronized within time window T, as in the schemes of Hauser et al. [12]. Otherwise, we suffer from the same delay-and-forge attack as shown in [12]. For example, a malicious intermediate router receives the ith message from router A, in which the jth bit in the message digest is '0' and he wants to impersonate router A with a forged message F whose message digest is the same as the true ith message except that the jth bit is '1'. He keeps the message and waits for the $(i+1)$th message from A. If the jth bit in the $(i+1)$th message digest is '1', then he can compute the hash value for the jth bit in ith message by hashing the jth value he gets in $(i+1)$th message. Now he can forge a good signature for message F and forward F to next router. If the next router has a clock drift of more than T, the forged message will be accepted as both timely and genuine.

5. Independent One-time Signature Protocol

In COSP as well as in FLS, delay-and-forge attack is possible because different sets of one-time public key components and secret key components are inter-linked, i.e., the ith set of secret key components are used as the $(i+1)$th set of public key components. One way to overcome the first two shortcomings of FLS is to remove this link as used in Independent One-time Signature Protocol (IOSP).

In Independent One-time Signature Protocol (IOSP), each set of secret key components, and therefore the corresponding public key components, are chosen independently from one another. Every set of secret and public key components are used only once. In order to sign messages consecutively, in every signed message, a public key for verifying the next message is enclosed. However, we don't have to put the public key components themselves in the message, which will results in considerable message expansion. Instead, for verifying next message, we only need to give as public key P the hash value of the list of public key components for next message. The protocol goes as follows:

- Setup
  1. Each router chooses at random as secret key components x_j, $j = 1, ..., n$.
  2. Each router computes one-time public key $P = h(h(x_1)\| \cdots \|h(x_n))$, where $\|$ means concatenation.
  3. Each router signs P using public-key cryptography and broadcasts P with its public-key signature.
  4. Each router verifies the received P's from other routers are good using public-key cryptography and stores them.
- Signing the ith message M_i
  1. Choose at random as secret key components for next message x'_j, $j = 1, ..., n$.
  2. Compute one-time public key P' for next message as $P' = h(h(x'_1)\| \cdots \|h(x'_n))$.
  3. Obtain a n-bit binary string g by concatenating $f(M_i\|P')$ with a count field using Merkle's method as explained above.

4. Compute one-time signature S by concatenating signature components s_j, $j = 1, \cdots, n$, given by

$$s_j = \begin{cases} h(x_j) & \text{if } g_j = 0 \\ x_j & \text{if } g_j = 1 \end{cases}$$

where g_j is the jth bit of string g.

5. Send out message $(M_i \| P')$ with one-time signature S.
6. Update x_j with value x'_j.

- Verification

1. Obtain the n-bit binary string g by concatenating $f(M_i \| P')$ with a count field using Merkle's method as explained above.
2. Compute $V = h(v_1 \| v_2 \| \cdots \| v_n)$, where v_j, $j = 1, \cdots, n$ is given by

$$v_j = \begin{cases} r_j & \text{if } g_j = 0 \\ h(r_j) & \text{if } g_j = 1 \end{cases}$$

where r_j is the received jth signature component and g_j is the jth bit of string g.

3. If $V = P$, accept the message and update P with value P'.

6. Performance

IOSP needs at most $n + 1$ hash computations to verify a one-time signature. On average, COSP needs similar computation while IOSP needs about half of that. We tested the signature verification speeds of IOSP using MD5 [31] for both f and h and compared with RSA [33] signature verification using 1024-bit modulus and 8-bit public-exponent on a 200MHz Pentium PC running Windows NT 4.0 with 64MB RAM. The crypto library we used was CryptoLib 1.1 by Jack Lacy et al. [21]. We observed around 133,000 hash computations per second for MD5 and around 110 signature verifications per second for RSA. Roughly speaking, signature verification using IOSP runs more than 10 times faster than RSA.

For signature generation, our protocols use little time. Whereas, for small public exponents RSA signature generation is much slower than RSA signature verification.

On average, one-time public key components generation for COSP and IOSP needs n hash computations. However, this can be done off-line.

7. Storage Requirements

The storage requirement for one-time public key P in IOSP is quite small, i.e., one hash-length per router. For MD5, it is only 16 bytes per router. The storage requirement for one-time public key components in COSP is $(l + \lfloor \log_2 l \rfloor + 1) \times m$ bits for each router, where l and m are the output length (in bits) of hash function f and h, respectively. If we have 1000 routers in a routing domain, the storage requirement is about 2MB. It is not likely to be a burden for link state routing in small routing domains or for distance vector routing. However, if memory space is a problem, possibly in the case of link state routing in large routing domains, we can resort to IOSP.

The storage requirement for one-time secret key components in IOSP is a few thousand bytes, e.g., 2KB for MD5. In COSP it is k times of that, where k is the number of messages we plan to sign for each Setup. Alternatively, storage requirement can be reduced to that of IOSP by just storing x_i. However, we need to do $n \times (i - 1)$ hash computations for generating the secret key components for the ith message. As a tradeoff between computation and storage, we can store the values x_i, $h^c(x_i)$, $h^{2c}(x_i)$, $\cdots$, for some c [20].

The storage requirement for one-time secret key components in both IOSP and COSP can be further reduced to a single value by generating all the secret key components in a pseudo-random fashion from a single secret key as used in [34, 4].

8. Comparison between COSP and IOSP

Generally speaking, IOSP signature verification runs twice as fast as COSP and IOSP uses less memory for storing the one-time secret key components and public key components. However, on average the signature size of COSP is roughly half of that of IOSP, e.g., 2KB for IOSP and 1KB for COSP using MD5.

Another attraction of COSP is that if router A misses some messages from router B, A can easily catch up since future secret key components can be authenticated using past public key components. Whereas, in IOSP if router A misses a message from router B, they have to re-setup. Therefore, COSP is more tolerant of unreliable packet delivery. However, COSP has to be used with timing constraints.

9. Applicability as Efficient Alternatives to Public-key Signatures

In addition to securing routing messages, our protocols can be used as efficient alternatives to public-key digital signatures for signing arbitrary messages. Our protocols are non-interactive, and hence, especially useful for signing broadcasted messages with a relatively stable audience.

As a general approach, the way our protocols being used with public-key systems for message signing is similar to

that of secret-key cryptography being used with public-key systems for data encryption. Since public-key encryption is expensive, in real systems we often use public-key systems to exchange session keys which are then used by secret-key systems to encrypt data. In our case, we use public-key cryptosystems to establish public key (components) for one-time signature schemes and use one-time signature protocols for signing messages.

10. Conclusion

In this work, we aim to reduce the computational costs of using public-key digital signatures in securing routing protocols. Two protocols (COSP and IOSP) using one-time digital signatures are introduced to provide the functionality of public-key digital signatures. Our protocols are intended to be used in place of public-key digital signatures for signing all kinds of message exchanges among routers. We obtained more than ten-fold increase in speed compared with public-key signatures.

Our protocols overcome the deficiencies identified in previous works, such as timing constraints, limited applications and high storage and computational costs for volatile environments [12]. The signature size and computational costs of our protocols are fixed irrespective of the size or content of message being signed. IOSP can be used at any frequency without timing constraints.

Since our protocols are non-interactive, they provide full functionality of a true signature. However, our protocols are not intended for replacing public-key infrastructures completely. Instead, as in previous work [12], public-key infrastructures are used to setup COSP and IOSP. Public-key cryptography is good at building the initial trust among users on a large and segmented network, like the Internet.

As a general approach, our protocols can be used with public-key cryptosystems for efficient message signing in much the same way as secret-key cryptosystems are used in conjunction with public-key systems for efficient data encryption.

11. Acknowledgements

I would like to thank Roger Needham, Ross Anderson and Yacov Yacobi for helpful comments and discussions. Many thanks to Fabien Petitcolas for helping me with the tests. I am also grateful to anonymous reviewers for helpful comments and suggestions.

References

[1] R. Anderson, C. Manifavas and C. Sutherland, NetCard - A Practical Electronic-Cash System, *Proc. 1996 Security Protocols Workshop*, Cambridge, UK, April 1996.

[2] N. Asokan, G. Tsudik and M. Waidner, Server-Supported Signatures, *Proc. 1996 European Symposium on Research in Computer Security*, September 1996.

[3] T.A. Berson, L. Gong and T.M.A. Lomas, Secure, Keyed, and Collisionful Hash Functions, *Technical Report SRI-CSL-94-08*, SRI International, May 1994.

[4] U. Blumenthal, N.C. Hien and B. Wijnen, Key Derivation for Network Management Applications, *IEEE Network*, May/June 1997.

[5] D. Bleichenbacher and U.M. Maurer, Directed Acyclic Graphs, One-way Functions and Digital Signatures, *Proc. CRYPTO'94*, LNCS 839, Springer Verlag, 1994, pp 75-82.

[6] D. Bleichenbacher, U.M. Maurer, Optimal Tree-Based One-time Digital Signature Schemes, *Proc. STACS'96*, LNCS 1046, Springer-Verlag, pages: 363-374, 1996.

[7] D. Bleichenbacher, U.M. Maurer, On the efficiency of one-time digital signatures, *Proc. ASIACRYPT'96*, LNCS 1163. Springer-Verlag, pages: 145-158, 1996.

[8] W. Diffie and M. Hellman, New Directions in Cryptography, *IEEE Trans. on Information Theory*, IT-22, November 1976, pp 644-654.

[9] S. Even, O. Goldreich and S. Micali, On-line/off-line digital signatures, *Proc. CRYPTO'89*, LNCS 435, Springer Verlag, 1990, pp 263-275.

[10] G.G. Finn, Reducing the vulnerability of dynamic computer networks, *Technical Report, ISI/RR-88-201*, USC ISI, Marina del Ray, CA, June 1988.

[11] N.M. Haller, The S/key(tm) one-time password system, *Proc. 1994 Internet Society Symposium on Network and Distributed System Security*, pp 151-157, San Diego, CA, February 1994.

[12] R. Hauser, T. Przygienda and G. Tsudik, Reducing the Cost of Security in Link-State Routing, *Proc. 1997 Internet Society Symposium on Network and Distributed System Security*, San Diego, CA, February 1997.

[13] R. Hauser, M. Steiner and M. Waidner, Micropayments based on iKP, *Proc. SECURICOM'96*, May 1996. (Also available as IBM Research Report.)

[14] R. Hauser and G. Tsudik, On Shopping Incognito, *Proc. 2nd USENIX Workshop on Electronic Commerce*, November 1996.

[15] Joint Technical Committee ISO/IEC JTC 1 *Information Technology*. Information Technology - Telecommunications and Information Exchange between Systems - Intermediate System to Intermediate System Intra-domain Routeing Information Exchange Protocol for Use in Conjunction with the Connectionless-mode Network Service (ISO 8473). ISO/IEC 10589, International Organization for Standardization, April 1992.

[16] Joint Technical Committee ISO/IEC JTC 1 *Information Technology*. Information Processing Systems - Telecommunications and Information Exchange between Systems - Protocol for Exchange of Inter-domain Routeing Information among Intermediate Systems to Support Forwarding of ISO 8473 PDUs. ISO/IEC 10747, International Organization for Standardization, October 1993.

[17] B. Kumar and J. Crowcroft, Integrating security in inter-domain routing protocols, *Computer Communications Review*, 23(5), October 1993.

[18] B. Kumar, Integration of Security in Network Routing Protocols, *SIGSAC Reviews*, 11(2):18-25, 1993.

[19] L. Lamport, Constructing digital signatures from one-way function, *Technical Report SRI-CSL-98*, SRI International, October 1979.

[20] L. Lamport, Password Authentication with Insecure Communication, *Communications of the ACM*, 24:11, Nov. 1981, pp 770-772.

[21] J.B. Lacy, D.P. Mitchell and W.M. Schell, CryptoLib: Cryptography in Software, *Proceedings of UNIX Security Symposium IV*, USENIX Association, 1993, pp 1-17.

[22] S. Micali, Enhanced Certificate Revocation System, *Technical Memo MIT/LCS/TM-542*, November 1995.

[23] Gary Malkin, RIP Version 2 Carrying Additional Information, Internet RFC 1723, Xylogics, Inc., November 1994.

[24] S.L. Murphy, M.R. Badger, Digital Signature Protection of OSPF Routing Protocol, *Proc. 1996 Internet Society Symposium on Network and Distributed System Security*, pp 93-102, San Diego, CA, February 1996.

[25] R.C. Merkle, A Digital Signature Based on a Conventional Encryption Function, *Proc. CRYPTO'87*, LNCS 293, Springer Verlag, 1987, pp 369-378.

[26] R.C. Merkle, A Certified Digital Signature, *Proc. CRYPTO'89*, LNCS 435, Springer Verlag, 1990, pp 218-238.

[27] C. Meyer and S. Matyas, *Cryptography - a new dimension in computer data security*, John Wiley & Sons, Inc., 1982.

[28] John Moy, OSPF Version 2, Internet RFC 1583, Proteon, Inc., March 1994.

[29] T.P. Pedersen, Electronic Payments of Small Amounts, *Proc. 1996 Security Protocols Workshop*, Cambridge, UK, April 1996.

[30] R. Perlman, Network Layer Protocols with Byzantine Robustness, *Technical Report MIT/LCS/TR-429*, Massachusetts Institute of Technology, Oct. 1988.

[31] R.L. Rivest, The MD5 Message Digest Algorithm, RFC1321, Apr 1992.

[32] R.L. Rivest and A. Shamir, PayWord and MicroMint: Two Simple Micropayment Schemes, *Proc. 1996 Security Protocols Workshop*, Cambridge, UK, April 1996.

[33] R.L. Rivest, A. Shamir and L.M. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, *Communications of the ACM*, 21:2, Feb 1978, pp 120-126.

[34] A.D. Rubin, Independent One-Time Passwords, *Proc. 5th UNIX Security Symposium*, USENIX Association, June 1995.

[35] K.E. Sirois and S.T. Kent, Securing the Nimrod Routing Architecture, *Proc. 1997 Internet Society Symposium on Network and Distributed System Security*, San Diego, CA, February 1997.

[36] B.R. Smith, S. Murthy and J.J. Garcia-Luna-Aceves, Securing Distance-Vector Routing Protocols, *Proc. 1997 Internet Society Symposium on Network and Distributed System Security*, San Diego, CA, February 1997.

[37] M. Steenstrup, Inter-Domain Policy Routing Protocol Specification: Version 1, Internet RFC 1479, BBN Systems and Technologies, July 1993.

[38] S. Vaudenay, One-time identification with low memory, *Proc. EUROCODE'92*, CISM Courses and lectures No. 339, pp. 217-228, Springer-Verlag, 1993.

[39] M. V. Wilkes, *Time-Sharing Computer Systems*, New York: Elsevier, 1972.