

## Retrieval of secure Data for military networks

<sup>1</sup>Pekka Siva Parvathi & <sup>2</sup> K.Srujana M.Tech

<sup>2</sup>Associate Professor, Department Of computer Science .

<sup>1,2</sup>Prakasam Engineering College ,Kandukuru – Prakasam .

**Abstract-** *Wireless sensor networks will be widely used to focused on making these networks feasible and useful, security has received little attention. We present a suite of security protocols optimized for sensor networks: SPINS.SPINS has two secure building blocks: SNEP and TESLA. SNEP includes: data confidentiality, two-party data authentication, and evidence of data freshness. TESLA provides authenticated broadcast for severely resource-constrained environments. We implemented the above protocols, and show that they are practical even on minimal hard ware: the performance of the protocol suite easily matches the data rate of our network. Additionally, we demonstrate that the suite can be used for building higher level protocols.*

**Keywords :** secure communication protocols; sense or networks; mobile ad hoc networks; MANET; authentication of wireless communication; secrecy and confidentiality, cryptography.

### I. Introduction

We envision a future where thousands to millions of small sensors form self-organizing wireless networks. How can we provide security for these sensor networks? Security is Not easy, compared with conventional desktop computers, Severe challenges exist .these sensors will have Limited Processing power, storage, bandwidth, and energy .We need to surmount these challenges, because security is so important. Sensor networks will expand to all aspects of our lives. Here are some typical applications:

**Emergency response information:** sensor networks will collect information about the status of buildings, people ,and transportation pathways. Sensor information must be collected and passed on in meaningful, secure ways to emergency response personnel.

**Medical monitoring:** we envision a future where individuals with some types of medical conditions receive constant monitoring through sensors that monitor health conditions. For some types of medical conditions, remote sensors may apply remedies (such as instant release of emergency Medication to the bloodstream).

**Battle field management:** remote sensors can help eliminate some of the confusion associated with combat. They can allow accurate collection of information about current battle field conditions as

well as giving appropriate information to soldiers, weapons, and vehicles in the battle field. At UC Berkeley, we think these systems are important, and we are starting a major initiative to explore the use of wireless sensor networks. security and privacy questions arise if third parties can read or tamper with sensor data. We envision wireless sensor networks being widely used .including for emergency and life-critical systems and here the questions of security are foremost.

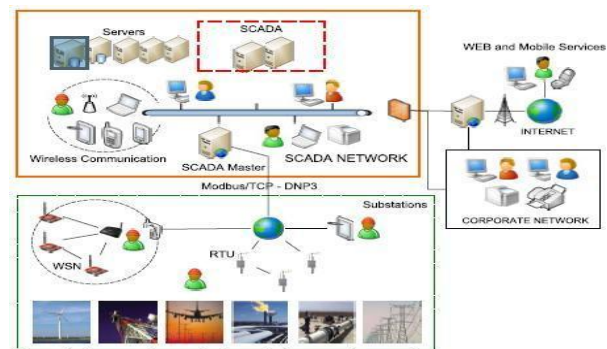


Fig1: Wireless Sensor Network

This article presents a set of Security Protocols for Sensor Networks, SPINS. The chief contributions of this article are:

- Exploring the challenges for security in sensor networks.
- Designing and developing TESLA providing authenticated streaming broadcast.
- Designing and developing SNEP (Secure Network Encryption Protocol) providing data confidentiality, two party data authentication, and data freshness, with low over head.
- Designing and developing an authenticated routing protocol using our building blocks.

### Data confidentiality

A sensor network should not leak sensor readings to neighboring networks. In many applications (e.g., key distribution) nodes communicate highly sensitive data. The standard approach for keeping sensitive data secret is to encrypt the data with a secret key that only

hence intended receivers possess, achieving confidentiality. Given the observed

Communication patterns, we set up secure channels

between nodes and base stations and later bootstrap other secure channels as necessary.

## Data authentication

Message authentication is important for many applications in sensor networks (including administrative tasks such as network reprogramming or controlling sensor node duty cycle). Since an adversary can easily inject messages, the receiver needs to ensure that data used in any decision-making process originates from a trusted source. Informally, data authentication allows a receiver to verify that the data really was sent by the claimed sender. Informally, data authentication allows a receiver to verify that the data really was sent by the claimed sender. In the two-party communication case, data authentication can be achieved through a purely symmetric mechanism: The sender and the receiver share a secret key to compute a message authentication code (MAC) of all communicated data. When a message with a correct MAC arrives, the receiver knows that it must have been sent by the sender. This style of authentication cannot be applied to a broadcast setting, without placing much stronger trust assumptions on the network nodes. If one sender wants to send authentic data to mutually untrusted receivers, using a symmetric MAC is insecure: any one of the receivers knows the MAC key, and hence, could impersonate the sender and forge messages to other receivers. Hence, we need an asymmetric mechanism to achieve authenticated broadcast. One of our contributions is to construct authenticated broadcast from symmetric primitives only, and introduce asymmetry with delayed key disclosure and one-way function key chains.

## NETWORK ARCHITECTURE

In this section, we describe the DTN architecture and define the security model.

Fig. 1. Architecture of secure data retrieval in a disruption-tolerant military network.

### A. System Description and Assumptions

Fig. 1 shows the architecture of the DTN. As shown in Fig. 1, the architecture consists of the following system entities.

1) Key Authorities: They are key generation centers that generate public/secret parameters for CP-ABE. The key authorities consist of a central authority and multiple local authorities. We assume that there are secure and reliable communication channels between a central authority and each local authority during the initial key setup and generation phase. Each local authority manages different attributes and issues corresponding attribute keys to users. They grant differential access rights to individual users based on the users' attributes. The key authorities are assumed to be honest-but-curious. That is, they will honestly execute the assigned tasks in the system, however they would like to learn information of encrypted contents as

much as possible. 2) Storage node: This is an entity that stores data from senders and provide corresponding access to users. It may be mobile or static [4], [5]. Similar to the previous schemes, we also assume the storage node to be semi-trusted, that is honest-but-curious.

3) Sender: This is an entity who owns confidential messages or data (e.g., a commander) and wishes to store them into the external data storage node for ease of sharing or for reliable delivery to users in the extreme networking environments. A sender is responsible for defining (attribute based) access policy and enforcing it on its own data by encrypting the data under the policy before storing it to the storage node.

4) User: This is a mobile node who wants to access the data stored at the storage node (e.g., a soldier). If a user possesses a set of attributes satisfying the access policy of the encrypted data defined by the sender, and is not revoked in any of the attributes, then he will be able to decrypt the cipher text and obtain the data. Since the key authorities are semi-trusted, they should be deterred from accessing plaintext of the data in the storage node; meanwhile, they should be still able to issue secret keys to users.

In order to realize this somewhat contradictory requirement, the central authority and the local authorities engage in the arithmetic 2PC protocol with master secret keys of their own and HUR AND KANG: SECURE DATA RETRIEVAL FOR DECENTRALIZED DISRUPTION-TOLERANT MILITARY NETWORKS 19

Issue independent key components to users during the key issuing phase. The 2PC protocol prevents them from knowing each other's master secrets so that none of them can generate the whole set of secret keys of users individually. Thus, we take an assumption that the central authority does not collude with the local authorities (otherwise, they can guess the secret keys of every user by sharing their master secrets). B. Threat Model and Security Requirements

1) Data confidentiality: Unauthorized users who do not have enough credentials satisfying the access policy should be deterred from accessing the plain data in the storage node. In addition, unauthorized access from the storage node or key authorities should be also prevented. 2) Collusion-resistance: If multiple users collude, they may be able to decrypt a cipher text by combining their attributes even if each of the users cannot decrypt the ciphertext alone [11]–[13]. For example, suppose there exist a user with attributes {"Battalion 1", "Region 1"} and another user with attributes {"Battalion 2", "Region 2"}. They may succeed in decrypting a cipher text encrypted under the access policy of ("Battalion 1" AND "Region 2"), even if each of them cannot decrypt it individually. We do not want these

colluders to be able to decrypt the secret information by combining their attributes. We also consider collusion attack among curious local authorities to derive users' keys.

3) Backward and forward Secrecy: In the context of ABE, backward secrecy means that any user who comes to hold an attribute (that satisfies the access policy) should be prevented from accessing the plaintext of the previous data exchanged before he holds the attribute. On the other hand, forward secrecy means that any user who drops an attribute should be prevented from accessing the plaintext of the subsequent data exchanged after he drops the attribute, unless the other valid attributes that he is holding satisfy the access policy.

## II. Related Work

Because of stringent resource constraints on the sensor nodes, implementation of the cryptographic primitives is a major challenge. We can sacrifice some security to achieve feasibility and efficiency, but we still need a core level of strong cryptography. Below we discuss how we provide strong cryptography despite restricted resources. Memory size is a constraint: our sensor nodes have 8 Kbytes of read-only program memory, and 512 bytes of RAM. The program memory is used for Tiny OS, our security infrastructure, and the actual sensor net application. To save program memory we implement all cryptographic primitives from one single block cipher [2]. Block cipher. We evaluated several algorithms for use as a block cipher. An initial choice was the AES algorithm from Irondale [12]; however, after further inspection, we sought alternatives with smaller code size and higher speed. The baseline version of Rijndael uses over 800 bytes of lookup tables which is too large for our memory-deprived nodes. An optimized version of that algorithm (about a 100 times faster) uses over 10 Kbytes of lookup tables. Similarly, we rejected the DES block cipher which requires a 512-entry S Box table and a 256-entry table for various permutations [32]. A small encryption algorithm such as TEA [54] is a possibility, but it has not yet been subject to cryptanalytic scrutiny. We use RC5 [47] because of its small code size and high efficiency. RC5 does not rely on multiplication and does not require large tables. However, RC5 does use 32-bit data-dependent rotates, which are expensive on our Atmel processor (it only supports an 8-bit single bit rotate operation). Even though the RC5 algorithm can be expressed succinctly, the common RC5 libraries are too large for our platform. With a judicious selection of functionality, we use a subset of RC5 from Open SSL, and after further tuning of the code we achieve an additional 40% reduction in code size. Encryption function. To save code space, we use the same function for both encryption and decryption. The counter (CTR) mode of block ciphers has this

property. CTR mode is a stream cipher. Therefore, the size of the cipher text is exactly the size of the plaintext and not a multiple of the block size. This property is particularly desirable in our environment. Message sending and receiving consume a lot of energy. Also, longer messages have a higher probability of data corruption. Therefore, block cipher message expansion is undesirable. CTR mode requires a counter for proper operation. Reusing a counter value severely degrades security. In addition, CTR-mode offers semantic security. The same plaintext sent at different times is encrypted into different cipher text since the encryption pads are generated from different counters. To an adversary who does not know the key, these messages will appear as two unrelated random strings. Since the sender and the receiver share the counter, we do not need to include it in the message. If the two nodes lose the synchronization of the counter, they can simply transmit the counter explicitly to resynchronize using SNEP with strong freshness. Freshness. Weak freshness is automatically provided by the CTR encryption. Since the sender increments the counter after each message, the receiver verifies weak freshness by verifying that received messages have a monotonically increasing counter. For applications requiring strong freshness, the sender creates a random (an unpredictable 64-bit value) and includes it in the request message to the receiver. The receiver generates the response message and includes it once in the MAC computation (see section 5). If the MAC of the response verifies successfully, the node knows that the response was generated after it sent the request message and hence achieves strong freshness. Random-number generation. The node has its own sensors, wireless receiver, and scheduling process, from which we could derive random digits. But to minimize power requirements, we use a MAC function as our pseudo-random number generator (PRG), with the secret pseudo-random number generator key. We also keep a counter that we increment after each pseudo-random block.

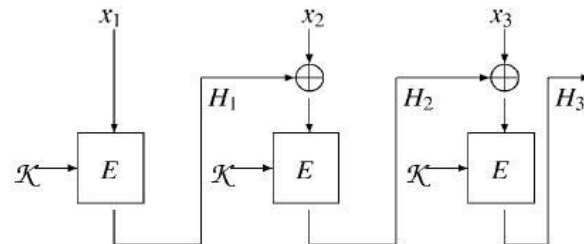


Fig2: MAC Authentication Code

## III. Performance Analysis

We evaluate the implementation of our protocols by





code size, RAM size, and processor and communication overhead. The code size of three implementations of crypto routines in Tiny OS. The smallest version of the crypto routines occupies about 20% of the available code space. The difference between the fastest and the smallest implementation stems from two different implementations of the variable rotate function. The TESLA protocol uses another 574 bytes. Together, the crypto library and the protocol implementation consume about 2 Kbytes of program memory, which is acceptable in most applications. It is important to identify reusable routines to minimize call setup costs. For example, Open SSL implements RC5 encryption as a function. On our sensor hardware, the code size of call setup and return outweigh the code size of the body of the RC5 function. We implement RC5 as a macro and only expose interfaces to the MAC and CTR-ENCRYPT functions. The performance of the cryptographic primitives is adequate for the bandwidth supported by the current generation of network sensors. Key setup is relatively expensive (4 ms). In contrast, the fast version of the code uses less than 2.5 ms to encrypt a 16 byte message and to compute the MAC (the smaller but slower version takes less than 3.5 ms). Let us compare these times against the speed of our network. Our radio operates at 10 kbps at the physical layer. If we assume that we communicate at this rate, we can perform key setup, an encryption, and a MAC for every message we send out. In our implementation, TESLA discloses the key after two intervals. The stringent buffering requirements also dictate that we cannot drop more than one key disclosure beacon. We require a maximum of two key setup operations and two CTR encryptions to check the validity of a disclosed TESLA key. Additionally, we perform up to two key setup operations, two CTR encryptions, and up to four MAC operations to check the integrity of a TESLA message. That gives an upper bound of 17.8 ms for checking the buffered messages. This amount of work is easily performed on our processor. In fact, the limiting factor on the bandwidth of authenticated broadcast traffic is the amount of buffering we can dedicate on individual sensor nodes. Table 4 shows the memory size required by the security modules. We configure the TESLA protocol with four messages: the disclosure interval dictates a buffer space of three messages just for key disclosure, and we need an additional buffer to use this primitive in a more flexible way. Despite allocating minimal amounts of memory to \_TESLA, the protocols we implement consume half of the available memory, and we cannot afford any more memory. Energy costs. We examine the energy costs of security mechanisms. Most energy costs will come from extra transmissions required by the protocols. Remaining security issues. Although this protocol suite addresses many security related problems, there remain many additional issues. First, we do not address the

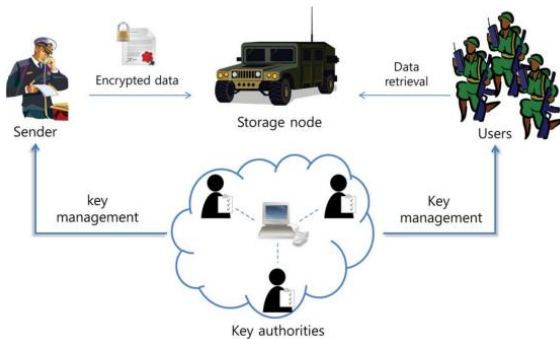
problem of information leakage through covert channels. Second, we do not deal completely with compromised sensors, we merely ensure that compromising a single sensor does not reveal the keys of all the sensors in the network. Third, we do not deal with denial-of-service (DoS) attacks in this work. Since we operate on a wireless network, an adversary can always perform DoS attack by jamming the wireless channel with a strong signal. Finally, due to our hardware limitations, we cannot provide Diffie-Hellman style key agreement or use digital signatures to achieve non-repudiation. For the majority of sensor network applications, authentication is sufficient.

### Authenticated Routing

Using the TESLA protocol, we developed a lightweight, authenticated ad hoc routing protocol that builds an authenticated routing topology. Ad hoc routing has been an active area of research [11]. Marti et al. discuss a mechanism to protect an ad hoc network against misbehaving nodes that fail to forward packets correctly [28]. They describe two mechanisms: a watchdog to detect misbehaving neighboring nodes, and a path rater to keep state about the goodness of other nodes. They propose running these mechanisms on each node. However, we are not aware of a routing protocol that uses authenticated routing messages. It is possible for a malicious user to take over the network by injecting erroneous, replaying old, or advertise incorrect routing information. The authenticated routing scheme we developed mitigates these problems. The routing scheme within our prototype network assumes bidirectional communication channels. The route discovery depends on periodic broadcast of beacons. Every node, upon reception of a beacon packet, checks whether it has already received a beacon (which is a normal packet with a globally unique sender ID and current time at base station, protected by a MAC to ensure integrity and that the data is authentic) in the current epoch. If a node hears the beacon within the epoch, it does not take any further action. Otherwise, the node accepts the sender of the beacon as its parent to route towards the base station. Additionally, the node would repeat the beacon with the sender ID changed to itself. This route discovery resembles a distributed, breadth first search algorithm, and produces a routing topology. However, in the above algorithm, route discovery depends only on the receipt of route packet, not on its contents. It is easy for any node to claim to be a valid base station. In contrast, we note that the TESLA key disclosure packets can easily function as routing beacons. We accept only the sources of authenticated beacons as valid parents. Reception of a TESLA packet guarantees that that packet originated at the base station, and that it is fresh. For each time interval, we accept as the parent the first node sending a successfully authenticated packet. Combining TESLA key disclosure with distribution of routing beacons allows us to combine transmission of the

keys with network maintenance. We have outlined a scheme leading to a lightweight authenticated routing protocol for sensor networks. Since each node accepts only the first authenticated packet as the one to use in routing, it is impossible for an attacker to reroute arbitrary links within the sensor network. Each node verifies the behavior of the parent by implementing functionality similar to watchdogs described in [8]. The authenticated routing scheme above is just one way to build authenticated ad hoc routing protocol using TESLA. In protocols where base stations are not involved in route construction, TESLA can still be used for security. In these cases, the initiating node will temporarily act as base station and beacons authenticated route updates.

### Node-to-node key agreement



A convenient technology for bootstrapping secure connections is to use public key cryptography protocols for symmetric key setup [2]. Unfortunately, our resource constrained sensor nodes prevent us from using computationally expensive public key cryptography. We need to construct our protocols solely from symmetric key algorithms. We design a symmetric protocol that uses the base station as a trusted agent for key setup. Assume that the node wants to establish a shared secret session key.

$$\begin{aligned}
 A \rightarrow B: & \quad N_A, A, \\
 B \rightarrow S: & \quad N_A, N_B, A, B, \text{MAC}(K'_{BS}, N_A | N_B | A | B), \\
 S \rightarrow A: & \quad \{SK_{AB}\}_{K_{SA}}, \text{MAC}(K'_{SA}, N_A | B | \{SK_{AB}\}_{K_{SA}}), \\
 S \rightarrow B: & \quad \{SK_{AB}\}_{K_{SB}}, \text{MAC}(K'_{SB}, N_A | B | \{SK_{AB}\}_{K_{SB}}).
 \end{aligned}$$

The protocol uses our SNEP protocol with strong freshness that the key was really generated by the base station. Note that the MAC in the second protocol message helps defend the base station from denial-of-service attacks, and the base station only sends two messages to and if it received legitimate request from one of the nodes. A nice feature of the above protocol is that the base station performs most of the transmission work. Many other protocols involve a ticket that the server sends to one of the parties which forwards it to the other node, which requires more energy for the nodes to forward the message. The Kerberos key

agreement protocol achieves similar properties, but it does not provide strong key freshness [17,13]. If Kerberos used SNEP with strong freshness, then Kerberos would have greater security. The key distribution for resources starved devices in a mobile environment [5]. Park et al. [7] point out weaknesses and improvements. Beller and Yacobi further develop key agreement and authentication protocols [4]. Boyd and Mathuria survey the previous work on key distribution and authentication for resource-starved devices in mobile environments [8]. The majority of these approach easily on asymmetric cryptography. Bergstrom et al. consider the problem of secure remote control of resource-starved devices in a home [6]. Fox and Gribble present a security protocol providing secure access to application level proxy services [16]. Their protocol is designed to interact with a proxy to Kerberos and to facilitate porting services relying on Kerberos to wireless devices. The work of Patel and Crocroft focuses on security solutions for mobile user devices [39]. Unfortunately, their work uses asymmetric cryptography and is, Page 4

hence, too expensive for the environments we envision. The work of Czerwinski et al. also relies on asymmetric cryptography for authentication [10]. Stajano and Anderson discuss the issues of bootstrapping security devices [51]. Their solution requires physical contact of the new device with a master device to imprint the trusted and secret information. Zhou and Haas propose to secure ad hoc networks using asymmetric cryptography [57]. Recently, Basagni et al. proposed to use a network-wide symmetric key to secure an ad hoc routing protocol [2]. While this approach is efficient, it does not resist compromise of a single node. Carman et al. analyze a wide variety of approaches for key agreement and key distribution in sensor networks [9]. They analyze the overhead of these protocols on a variety of hardware platforms. Marti et al. discuss a mechanism to

protect an ad hoc network against misbehaving nodes that fail to forward packets correctly [28]. They propose that each node runs a watchdog (to detect misbehaving neighboring nodes) and a path rater (to keep state about the goodness of other nodes); their solution, however, is better suited for traditional networks, with emphasis on reliable point-to-point communication, than to sensor networks. Hubaux et al. present a system for ad hoc peer-to-peer authentication based on public key certificates [24]. They consider an ad hoc network with nodes powerful enough for performing asymmetric cryptographic operations.

### IV. Conclusion

We designed and built a security subsystem for an extremely limited sensor network platform. We have identified and implemented useful security protocols for sensor networks: authenticated and confidential communication, and Authenticated broadcast. We have implemented applications including an authenticated routing scheme and a secure node-to-node key agreement protocol. Most of our design is universal and applicable to other networks of low-end devices. Our primitives only depend on fast symmetric cryptography, and apply to a wide variety of device configurations. On our limited platform energy spent for security is negligible compared with to energy spent on sending or receiving messages. It is possible to encrypt and authenticate all sensor readings. The communication costs are also small. Data authentication, freshness, and confidentiality properties use up a net 6 bytes out of 30 byte packets. So, it is feasible to guarantee these properties on a per packet basis. It is difficult to improve on this scheme, as transmitting a MAC is fundamental to guaranteeing data authentication. Certain elements of the design were the available experimental platform. If we had a more powerful platform, we could have used block ciphers other than RC5. The emphasis on code reuse is another property forced by our platform. A more powerful device would allow more modes of authentication. In particular, memory restrictions on buffering limit the effective bandwidth of authenticated broadcast. Despite the shortcomings of our target platform, we built a system that is secure and works. With our techniques, we believe security systems can become an integral part of practical sensor networks.

## REFERENCES

- [1] J. Peerenboom and R. Fisher, "Analyzing Cross-Sector Interdependencies", IEEE Computer Society, HICSS, IEEE Computer Society, pp. 112–119, 2007.
- [2] ZigBee Alliance, <http://www.zigbee.org/>, accessed on February, 2010.
- [3] WirelessHART, <http://WirelessHART.hartcomm.org/>, HART Communication Foundation, accessed on February, 2010.
- [4] ISA100, "Wireless Systems for Automation", <http://www.isa.org/Content/NavigationMenu/Technical/Information/ASCI/ISA100/Wireless/CompLiance/Institute/ISA100/Wireless/CompLiance/Institute.htm>, accessed on February, 2010.
- [5] Modbus-IDA, "The Architecture for Distributed Automation", <http://www.modbus.org/>, accessed on February, 2010.
- [6] DNP3, "DNP Users Group", <http://www.dnp.org>, accessed on February, 2010.
- [7] HART Communication, <http://www.hartcommunication.org/>, accessed on February, 2010.
- [8] J. Lopez, R. Roman and C. Alcaraz, "Analysis of Security Threats, Requirements, Technologies and Standards in Wireless Sensor Network", Foundations of Security Analysis and Design V, LNCS 5705, pp. 289–338, Springer, 2009.
- [9] R. Roman, C. Alcaraz and N. Sklavos, "On the Hardware Implementation Efficiency of Cryptographic Primitives", Wireless Sensor Network Security, Cryptology and Information Security Series, vol. 1, pp. 285–305, 2008.
- [10] IEEE 802.15.4-2006, "IEEE Standard for Information Technology-Telecommunications and Information Exchange between Systems-Local and Metropolitan Area Networks", <http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf>, 2006.
- [11] D. Johnson, D.A. Maltz and J. Broch, The dynamic source routing protocol for mobile ad hoc networks, Internet draft, Mobile Ad-Hoc Network (MANET) Working Group, IETF (1999).
- [12] J. Daemen and V. Rijmen, A proposal: Rijndael (1999).
- [13] Dallas, iButton: A Java-powered cryptographic iButton, <http://www.ibutton.com/ibuttons/java.html>
- [14] W. Diffie and M.E. Hellman, Privacy and authentication: An introduction to cryptography, Proceedings of the IEEE 67(3) (1979) 397.427.
- [15] Fortezza, Fortezza: Application implementers guide (1995).
- [16] A. Fox and S.D. Gribble, Security on the move: Indirect authentication using Kerberos, in: International Conference on Mobile Computing and Networking (MobiCom'96) (1996) pp. 155.164.
- [17] R. Gennaro and P. Rohatgi, How to sign digital streams, in: Advances in Cryptology - Crypto'97, Lecture Notes in Computer Science, Vol. 1.1294 (1997) pp. 180.197.
- [18] O. Goldreich, S. Goldwasser and S. Micali, How to construct random functions, Journal of the ACM 33(4) (1986) 792.807.
- [19] S. Goldwasser and S. Micali, Probabilistic encryption, Journal of Computer Security 28 (1984) 270.299.
- [20] Z. Haas and M. Perlman, The Zone Routing Protocol (ZRP) for ad hoc networks, Internet draft, Mobile Ad-Hoc Network (MANET) Working Group, IETF (1998).
- [21] N.M. Haller, The S/KEY one-time password system, in: Symposium on Network and Distributed Systems Security (1994).



- [22] D. Harkins and D. Carrel, The Internet key exchange (IKE), RFC2409, Information Sciences Institute, University of Southern California(1998).
- [23] J. Hill, R. Szcwcyk, A.Woo, S. Hollar, D. Culler and K. Pister, System architecture directions for networked sensors, in: International Conference on Architectural Support for Programming Languages and Operating Systems.
- [24] J.-P. Hubaux, L. Buttyán and S. Capkun, The quest for security in mobile ad hoc networks, in: ACM Symposium on Mobile Ad Hoc Networking and Computing (2001).
- [25] D.B. Johnson and D.A. Maltz, Dynamic source routing in ad hoc wireless networks, in: Mobile Computing (Kluwer Academic, 1996) chapter 5, pp. 153.181.
- [26] Y.-B. Ko and N. Vaidya, Location -Aided Routing (LAR) in mobile ad hoc networks, in: International Conference on Mobile Computing and Networking (MobiCom'98) (1998).
- [27] J. Kohl and C. Neuman, The Kerberos network authentication service (V5), RFC 1510 (1993).
- [28] S. Marti, T. Giuli, K. Lai and M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in: International Conference on Mobile Computing and Networking (MobiCom 2000) (2000) pp. 255.265.
- [29] A.J. Menezes, P.C. van Oorschot and S.A. Vanstone, Handbook of Applied Cryptography (CRC Press, 1997).
- [30] S.P. Miller, C. Neuman, J.I. Schiller and J.H. Saltzer, Kerberos authentication and authorization system, Project Athena Technical Plan (1987).