Bootstrapping Trust in Networked Measurement Systems with Secure Sensors

Kristján Valur Jónsson
School of Computer Science, Reykjavik University
Menntavegur 1, Reykjavik, Iceland
kristjanvj@ru.is

Ýmir Vigfússon
School of Computer Science, Reykjavik University
Menntavegur 1, Reykjavik, Iceland
ymir@ru.is

***Abstract*—Aggregation of local sensor observations is a widely used and powerful approach to estimating the global state of large networked systems. However, guaranteeing the integrity of the aggregate remains an open problem, in particular for dynamic networked systems. We consider this problem in a single aggregator model – a system of several networked sensors and one inherently trusted collector. We argue that algorithmic approaches are insufficient in the general case and propose a simple, yet effective, solution based on the principles of trusted systems.**

I. Introduction

Informed decision making based on quantifiable information is contingent on accurate, high-quality measurements. We focus on networked measurement systems, in which sensors propagate local observations to a single collector in order to produce an aggregate – an approximate global view of the system [2]. In particular, we consider the integrity of the aggregation process in networked measurement systems in the presence of corrupt sensors, controlled by an insider adversary, whose goal is to stealthily [3] influence the global aggregate computed. We define such systems in a broad sense, encompassing diverse paradigms, such as wireless sensor networks [4], large-scale network monitoring systems [5] and industrial control systems.

The integrity of the entire aggregation process must be considered in the context of the consuming application and its operators. Our opinion is that the only applications which can tolerate arbitrary inputs, and hence, arbitrary aggregates, are trivial ones suitable only for the most basic of tasks. Critical aggregation examples include military applications of sensor networks [6], [7], public safety command and control [8] and nuclear plant monitoring [9]. We can also consider applications where money is at stake, such as accurate metering for utilities.

We ask the following question: *can the aggregation process be secured in the general case of arbitrary data types and aggregation functions and in dynamic networks, while at the same time giving sufficiently strong integrity guarantees?* In our view, this is infeasible unless one assumes some means of establishing a basis of trust at the sensor itself: we need some sort of integrity guarantees up front at the time the data is produced in order to guarantee integrity of the overall aggregation process. We propose to establish such guarantees by applying the principles of trusted systems to construct a *trusted sensor* – a verifiably correct, tamper-proof smart sensor [10], which provides cryptographically verifiable results to authorized recipients.

We restrict our current work to Wagner's single aggregator model in which a number of networked sensors contribute data to a single inherently trusted collector [11]. This model is conceptually simple, yet widely applicable, for instance in centralized network monitoring (e.g. SNMP polling), cloud-based sensing services (e.g. Pachube[1], and the relatively new paradigm of shared sensing via mobile devices [12].

In this paper, we discuss the problem of integrity preserving aggregation in the single aggregator framework and describe the design and prototyping of a system called *TSense*, which incorporates trusted smart sensors in a centralized client/server model with secure aggregation and authentication servers. The security guarantees are based on the concept of embedding the security mechanisms at the earliest possible point in the sensor chain: at the sensor "head" itself. This allows us to construct a secure measurement network of general-purpose networked observation platforms, e.g. wireless sensor nodes, routers or commodity PCs in a collaborative sensing project.

We can state that a solution which perfectly preserves integrity must be both *complete* and *correct* [13]. We will show (informally) that our solution meets the correctness criterion: if an observation platform contributes an update, then it will be a verifiably correct reflection of the sensor view at the time of sampling. Ensuring completeness is more problematic, as node churn and failures are a fact of life in a dynamic system and generate considerable uncertainty [14]. An asynchronous system with non-zero delays also introduces errors stemming from to differing sampling times. Both of these factors are outside of the scope of this work, but we remark that a reliability layer can mitigate the effects of naturally occurring packet loss.

We present a comprehensive client/server-based secure measurement system, based on the concept of trusted sensors in conjunction with an implicitly trusted support architecture. Our system design is supported by an open-source proof-of-

This work was supported in part by a Student Innovations Fund project grant from Rannís, the Icelandic Centre for Research, summer 2010 [1]. The project is also supported by Rannís Graduate Studies Grant #080520008 and funding by Reykjavik University.

[1] http://www.pachube.com

concept prototype which consists of a sensor implemented on an Arduino experimentation board.

II. System Model

Network model. Consider an asynchronous network of observation nodes S, each hosting a trusted sensor s, and a single collector and aggregator C. The sensors first observe some local phenomenon, such as temperature, pollution particle count or CPU load. Periodically, they then transmit a digest of the observations to the collector over an end-to-end secure channel $\mathcal{C}_{s_i,C} : s_i \Leftrightarrow C = s_i \leftrightarrow S_i \leftrightarrow C$, which has been established pairwise between trusted sensors and the collector over an arbitrary underlying communications graph. The actual communications path between any observation platform S_i and C may involve multiple hops, possibly even involving another observation node S, but the actions of intermediary nodes are strictly limited to routing.

Aggregation model. The collector C receives a set of observations $\mathbf{m}^t$ from some subset of observers $S^t \subseteq S$ over some period of time $\Delta = [t, t+\tau)$. An aggregate $y^t = f(\mathbf{m})$ is computed over the contributions in $\mathbf{m}$. A simple example is the SUM aggregation function: for contributions $[m_1, \dots, m_k]$ received in $[t, t+\tau)$, the aggregate is $y^t = \sum_{i=1}^{k} m_i$. We emphasize that our work is not restricted to scalar data types or aggregation functions. Rather, we consider any computable function, including complex sensor fusion [15], [16].

Adversarial model. Observation nodes S are vulnerable systems, such as unhardened sensor nodes or general-purpose computing platforms, and hence, inherently untrusted. In contrast, the trusted sensors s are considered incorruptible by virtue of physical and cryptographic protection. The collector C is considered inherently trusted with respect to the integrity of its actions by virtue of being the end consumer of the information produced; any malfeasance is contrary to its objectives.

This model is based on the work of Wagner [11]. Note that we neither consider the corrupt aggregator model [3] nor the hierarchical aggregation models [17]–[19] in this paper. A *stealthy internal adversary* [3] can corrupt any number of observation nodes, but not the collector. The objective of the adversary is to bias the computed aggregate by manipulating the output of the sensors under her control, but to do so over an extended period of time without significantly risking detection.

Outsider attackers are excluded by the assumption of end-to-end secure channels. They can be readily constructed, even for resource constrained platforms, by hop-by-hop security [20], [21] combined with end-to-end mutual authentication procedures. Note that the channel $\mathcal{C}_{s_i,C}$ effectively treats the hosting platform S_i as an outsider w.r.t. the aggregation process, thereby removing its capability to interfere with the value of the updates. The host S_i, however, can drop legitimate updates in their entirety.

We will not consider availability attacks, such as routing and DoS attacks, because they run counter to the goals of the stealthy adversary as defined above. We also disregard other important security objectives, such as confidentiality and privacy, although our solution can provide confidentiality at a modest additional cost. We further ignore process-of-measurement (PoM) attacks [22]–[24] in which the sensing process itself is attacked; we consider network or host-based attacks on the measurement process in the communications path from transducer to collector. We view measures to counter PoM attacks as complementary to the goals of this work.

III. TSense System Design

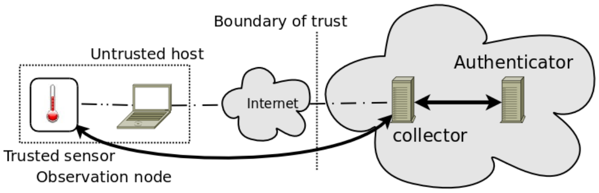


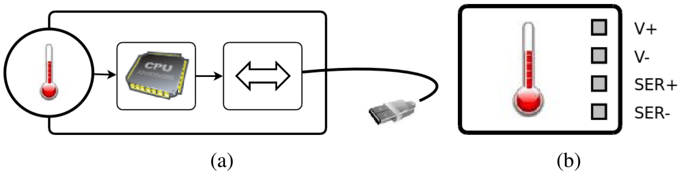
Fig. 1. System overview. Interaction of the entities of the TSensor system. Secure channels are indicated with bold lines.

Consider a networked measurement system as shown in Figure 1. A trusted central collector receives updates from a number of *observation nodes*, which are unhardened platforms, such as general-purpose laptops, hand-held devices, wireless sensor nodes or PLCs. Observation nodes can thus literally be in the hands of the adversary or corrupted in various ways. Historically, the nodes in wireless sensor networks have been considered inherently vulnerable due to cost constraints [25], while a look at the ISC's[2] top vulnerabilities list suffices as generalization to general-purpose platforms. In contrast, industrial control systems have been considered relatively immune to corruption owing to their relatively obscure systems and specialized, proprietary networks. However, these assumptions are challenged by the recent Stuxnet worm [26], as well as the increasing trend towards integration of industrial and business networks [27].

Each observation node contains one or more trusted sensors, which establish secure channels to the collector, thereby reducing the potential for the untrusted host to influence the aggregation process. This system can be extended to a distributed collector architecture in a straightforward manner by creating a trusted subnet within the boundary of trust.

The trusted sensor, shown schematically in Figure 3, is a tamper-resistant device whose instruction set and interface is small enough to be formally specified and verified. The integrity of the sensor data is ensured by an in-line cryptographic processor, as shown in Figure 2(a), comparable to a reference monitor in the trusted systems literature [28]. Hence, we can consider the use of a trusted sensor as an extension of the trusted computing base (TCB) of the hosting node to encompass the security critical functions of the sensing process. The trusted sensor operates in a close symbiotic relationship with its otherwise untrusted hosting node. Specifically, it has

[2] http://isc.sans.org



(a) (b)

Fig. 2. Trusted sensor: (a) Schematic of a trusted sensor with a USB-connector, (b) Sample pinout diagram. Bold outlines indicate tamper-proof enclosures.

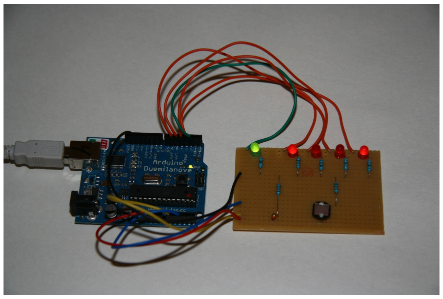


Fig. 3. The trusted sensor prototype. A sensor interface board is connected to the Arduino experimentation board. USB cable connects the Arduino board to a laptop computer (not shown) acting as a measurement platform.

only a rudimentary serial interface, as shown in Figure 2(b), which exposes a well-defined set of functionality. The trusted interface is utilized by an untrusted sensing application agent on the measurement node which may be arbitrarily influenced or even modified by the adversary. *The main security goal of the TSense system is to exclude such misbehavior and guarantee that any update produced by a measurement node is a truthful representation of the local state as observed by a trusted sensor.*

A natural question is whether the assumption of a local untrusted agent and untrusted measurement node is necessary. The short answer is no: a measurement node can be verified as a complete system and hardened to be sufficiently invulnerable to adversarial tampering. However, this is a non-trivial undertaking for any but the smallest system. We thus argue for securing only basic, rudimentary functionality in keeping with the spirit of trusted systems theory.

Identity and authentication. Each trusted sensor is uniquely identified by a public identity and a secret cryptographic key, both assigned at time of construction by the manufacturer. The manufacturer is assumed to be trusted for the purposes of this work; we do not consider the untrusted manufacturer problem [29]. Our system uses a shared secret key K_{As}, which is implemented in our prototype and described in Section IV; an asymmetric approach based on digital certificates and public key cryptography is also feasible. A shared private secret requires the use of a trusted 3^{rd} party – the authentication server in Figure 1. An unique identity per sensor ensures graceful degradation of security in the case of sensor breach.

Protocols. The cryptographic protocols are the glue that holds the system together. Trusted sensors provide the basis for trust, but secure authentication and transport protocols are required to establish the virtual secure channels between the measurement node and the trusted collector. We will describe the protocols further in Section IV, but let us begin by outlining the issues involved. First of all, we require a strong authentication protocol, allowing a sensor to be identified as a member of the trusted group. This is accomplished by the trusted sensor identity and the assistance of the trusted 3^{rd} party: any party which can match the secret K_{As} held by the authentication service for the identity s must be considered a member of the group of trusted sensors.

The authentication process establishes a shared secret between the sensor and collector, that is, a session key, which we call K_{Cs}. Prudent cryptographic principles dictate that one should use the session key sparingly. Hence, we specify a second protocol for periodic rotation of transport keys $K_{Cs,t}$.

The third protocol is a transport protocol, which applies fast symmetric authenticating encryption [30] to secure the integrity as well as privacy of the transmitted data against active and passive attackers, using $K_{Cs,t}$. Note that in all cases, we assume K to be a key pair, consisting of a key for encryption and an independent one for authentication, in keeping with the key separation principle [31].

IV. Prototype Implementation

We have implemented a complete proof-of-concept TSense system[3]. The trusted sensor, *tsensor*, shown in Figure 3, is a USB dongle with temperature and luminosity sensors, powered off the USB bus. We base our implementation on a popular embedded systems processor, the Atmel[4] ATmega328 [32]. For ease of prototyping, we used the Arduino Duemilanovae[5] experimentation board as our development platform. The collector and authenticator are implemented as daemons running on Ubuntu Linux servers.

Implementation. All code is written in C++, including the cryptographic protocols, AES-128 [33] in CBC mode [34] for encryption and AES-CMAC [35] for tagging. The tsensor prototype is written in about 1400 lines (SLOC) in a C++ variant developed for the Arduino platform; the cryptographic library, which is shared between the tsensor and server platforms, accounts for estimated 880 SLOC thereof. The compiled tsensor binary is 14.5KB and the executing code uses approximately 900 bytes of RAM, including measurement and processing buffers. A newer version of the cryptographic code for the Arduino platform has recently been launched as an open-source project[6].

We measured the throughput of the tsensor in our test environment to be 18.9KB/sec for encryption and 9.3KB/sec for decryption. In comparison, the same cryptographic code executing on our Linux servers achieved a throughput of 4.3MB/sec for encryption. We conclude that the performance

[3] The code is open-source and maintained at http://code.google.com/p/tsense.

[4] http://www.atmel.com

[5] http://arduino.cc/en/Main/ArduinoBoardDuemilanove

[6] https://github.com/kristjanvj/ACrypto

$$\begin{aligned} s \leftarrow S: &\quad \langle \textsc{identify} \rangle \\ s \rightarrow S \rightarrow C \Rightarrow A: &\quad \langle \textsc{auth}, s, \mathcal{ET}_{As}(s, N_s) \rangle \\ C \Leftarrow A: &\quad \langle \textsc{auth}, s, K_{Cs}, \mathcal{ET}_{As}(N_s, K_{Cs}) \rangle \\ s \leftarrow S \leftarrow C: &\quad \langle \textsc{auth}, \mathcal{ET}_{As}(N_s, K_{Cs}) \rangle \end{aligned}$$

Protocol 1: Authentication protocol (sketch). Sensor s and its untrusted host S communicate with collector C and authentication service A. $\mathcal{ET}_K$ denotes authenticating encryption using a shared key pair K. N_x is a freshly generated nonce; a counter suffices. Communication between C and A is protected by a TLS tunnel and encryption is implicit. K_{As} is the pairwise pre-shared key which serves as an unique authentication token for s.

$$\begin{aligned} s \rightarrow S \rightarrow C: &\quad \langle \textsc{rekey}, s, \mathcal{ET}_{Cs}(s, N_s) \rangle \\ s \leftarrow S \leftarrow C: &\quad \langle \textsc{newkey}, s, \mathcal{ET}_{Cs}(s, N_s, R) \rangle \end{aligned}$$

Protocol 2: Transport key establishment and refreshing protocol (sketch). Sensor s requests a transport key, using the session key K_{Cs} established during authentication. C returns key material R – a random number. Both nodes execute a key derivation function $K_{Cs,t} = \text{KDF}(R)$ to establish shared transport keys.

$$s \rightarrow S \rightarrow C : \langle \textsc{update}, s, \mathcal{ET}_{Cs,t}(s, t, D) \rangle$$

Protocol 3: Transport protocol (sketch). Sensor s sends encrypted and tagged data to C using the shared transport key set $K_{Cs,t}$. t is a timestamp of the data D.

of the sensors is quite acceptable for the task at hand. We defer measurements of power consumption and CPU cycles per operation to future work.

Protocols. The authentication protocol is a variant of the Needham-Schroeder symmetric trusted third party protocol [36]. A sketch is shown in Protocol 1. The untrusted agent on the measurement platform bootstraps the process by querying the sensor for its ID. The sensor responds by transmitting an encrypted packet, only readable by the authentication service (the trusted third party). At this point, the client has only two choices: to drop the packet in which case it has no further influence on the aggregation process, or to follow the protocol and transmit it to the collector. The collector forwards the unmodified packet to the authentication service, which returns a random session key (separately encrypted) to collector and sensor. The session key is used to set up a shared and periodically refreshed transport key between the sensor and collector, as shown in Protocol 2.

The data transfer protocol is shown in Protocol 3. This is a straightforward symmetric authenticating encryption protocol, which uses the transport key pair described above – AES in CBC mode for encryption and AES-CMAC for authentication in an encrypt-then-MAC composition [30]. Note that the protocol is shown as strictly best-effort, but can readily be made reliable by assuming an underlying reliable transport layer; our prototype uses TCP/IP.

V. Informal Security Analysis

We assume the protocols to be secure (within computational bounds) for the purposes of this brief analysis. We focus on the two important properties for *data integrity*, namely *completeness* and *correctness*, as specified by Narashima and Tsudik [13] in the context of distributed databases.

Integrity guarantees. The correctness of results is guaranteed by (i) the physical protection of the sensor and (ii) the authenticating encryption applied to the data produced. The authentication guarantees (within computational bounds) that any manipulation of the data by corrupt nodes, including the observation platform hosting the sensor, will be detected and the corresponding update discarded. The encryption adds an extra layer of difficulty for the adversary. This ensures that *if an update m_i^t by a sensor s_i is received and verified by the collector C, then m_i^t is correct.*

Completeness – the property that all sensors deliver results – is more challenging to guarantee, if only for the fact that malfunctions and churn are commonplace in all distributed systems. A corrupt observation node may mount drop attacks, transparently dropping inconvenient measurements, thereby influencing the aggregate result [14]. However, this is a much less powerful attack than the data modification attack, considered by Wagner [11], in which a single corrupt node may arbitrarily influence the aggregate. The effectiveness of the drop attack is reduced by the confidentiality guaranteed by the symmetric encryption, which guarantees that no unauthorized party, including the measurement node hosting the sensor, can observe its readings. Consequently, a corrupt measurement node cannot mount drop attacks on the basis of the observed sensor readings: if a malicious node is to drop packets, it must do so at random, thereby reducing the potential effectiveness of the attack.

Physical security. We do not address physical security in the prototype, as can readily be seen from Figure 3. The non-trivial issues regarding tamper-resistance [37], [38] as well as active and passive probing of the device interface [39] is reserved for future work.

Properly designed tamper-proofing should be of sufficient strength to slow the attackers progress in breaching the device. Further, the device should be rendered inoperable by the tampering. If these assumptions hold, we can claim graceful degradation of security: each breach requires considerable work, limiting the practical number of compromised sensors. Further, the internal secrets learned do not help the attacker, as they are unique to the now inoperable device.

VI. Background and Related Work

The term resilient aggregation was coined by Wagner [11], who considers the inherent vulnerabilities of many common scalar aggregation functions in a model in which a single inherently trusted aggregator receives data form several vulnerable sensors, any of which may be compromised. Wagner's results show that many common aggregation functions, such as SUM, MAX and MIN, are inherently vulnerable to even a single corrupt sensor.

Several approaches are suggested by Wagner to increase the resilience of aggregation, including truncation and employing inherently resilient aggregation functions. Truncation – limiting the sensor range to reduce the effects of adversaries – is employed by Chan *et al.* [19], but cannot be considered a good overall solution as it reduces the data fidelity while giving rather weak security guarantees. More resilient aggregation functions, such as the median, have been considered [40]. Probabilistic counting [41], [42] is another technique that provably increases the resilience of aggregation. Generalizing further, the previous work in this direction bases its security guarantees on rather simple specialized aggregation function on scalars, a severely restrictive assumption considering future sensing applications, such as sensor fusion [16], [43], [44]. In contrast, we support the general case of arbitrary data types, aggregation functions, as well as dynamic networks with ever changing node population and link conditions.

Secure aggregation is a widely studied problem for which a number of other models can be constructed. Chan *et al.* [3] consider the untrusted aggregator problem, also in a single aggregator model, with the stipulation that a small fraction of sensors can be compromised. A successful attack on an aggregator is clearly devastating, since it allows the adversary to manufacture arbitrary values at will. To counter such an attack, Chan *et al.* employ interactive proof techniques to limit the number of opportunities for the corrupt aggregator to falsify contributions of individual sensors, yet preserving the economy of aggregation in terms of message complexity. We consider this problem to be orthogonal to the subject of our paper, but remark that the concept of secure sensors can be extended to securing the aggregation process in a relatively easy manner by similarly secured aggregation processors [45].

The single aggregator model can be extended to a hierarchy of aggregators [17], [18], in which the aggregate is computed cooperatively in-network, resulting in considerable savings in terms of power, as well as message and time complexity. In this setting, however, the problem of securing the aggregation process becomes even harder and is currently considered to be an open problem, in particular w.r.t. dynamic systems. An algorithmic approach to securing such hierarchical aggregation networks is presented by Chan *et al.* [19], but relies on a static tree topology with fully known node membership. We again refer to our previous work in this regard [45].

Trusted devices have been applied in a range of situations: for instance to solve the fair exchange problem [46], [47], to provide secure storage primitives [48] and in secure multi-party computation [49]. Trusted sensors have been considered recently by several authors [44], [50], [51]. The projects cited all implement trusted sensing in a similar manner to our proposed solution, but with the crucial exception that trust is based on a Trusted Platform Module (TPM). In contrast, we propose to integrate dedicated minimal line-speed security logic in the sensors, rather than use a TPM, since the current generation of TPMs is bloated in terms of functionality and provides slow cryptographic operations [52]. We believe our approach of a dedicated and minimal sensor device is more prudent and in the spirit of the original concepts of trusted computing.

VII. Conclusions and Future Work

We discussed the problem of trusted sensing in distributed environments with untrusted parties. We proposed a solution based on the principles of trusted systems, consisting of tamper-proof trusted sensor modules, a set of protocols and trusted infrastructure components, which together form a trusted client/server-based measurement network that in turn guarantee (within computational bounds) safe data delivery from the tamper-proof sensor to a trusted collector.

We described an open-source proof-of-concept prototype implementation of our system, based on symmetric cryptographic primitives and a trusted third party authentication service. Our cryptographic code has been tested on both Arduino and Linux platforms and has been released into the public domain as an Arduino library.

Although limited our prototype is limited in several ways, we find it to be a reasonable stepping-stone towards the implementation of a trusted sensor. Future work includes verifying a custom security chip akin to SmartCard or RFID processors, which are currently up to the task in terms of size, cost and processing power. Further developments include developing of a version based on asymmetric cryptographic primitives that will decreased the reliance on trusted infrastructure services.

Acknowledgments

The authors would like to thank Kristján Rúnarsson and Benedikt Kristinsson for their work on the programming and testing of the TSense prototype.

References

[1] K. Rúnarsson, B. Kristinsson, and K. V. Jónsson, “TSense: Trusted sensors and support infrastructure,” Rannís NSN project report, September 2010.

[2] S. Keshav, “Efficient and decentralized computation of approximate global state,” *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 69–74, 2006.

[3] H. Chan, A. Perrig, B. Przydatek, and D. Song, “SIA: Secure information aggregation in sensor networks,” *J. Comp. Sec.*, vol. 15, no. 1, pp. 69–102, 2007.

[4] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless sensor network survey,” *Comput. Netw.*, vol. 52, no. 12, pp. 2292 – 2330, 2008.

[5] R. Stadler, M. Dam, A. Gonzalez, and F. Wuhib, “Decentralized real-time monitoring of network-wide aggregates,” in *LADIS*. New York, NY, USA: ACM, 2008, pp. 1–6.

[6] A. Arora, P. Dutta, S. Bapàt, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita, “A line in the sand: a wireless sensor network for target detection, classification, and tracking,” *Computer Networks*, vol. 46, p. 605634, 2004.

[7] T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J. A. Stankovic, and T. Abdelzaher, “Achieving real-time target tracking using wireless sensor network,” in *IEEE Real Time Tech. and App. Symp.*, 2006, pp. 37–48.

[8] L. Gomez, A. Laube, and C. Ulmer, “Secure sensor networks for public safety command and control system,” in *IEEE Conf. on Technologies for Homeland Security*, 2009.

[9] J. Barbarán, M. Díaz, I. naki Esteve, and B. Rubio, "RadMote: A mobile framework for radiation monitoring in nuclear power plants," *International Journ. of Electrical and Computer Eng.*, vol. 2, no. 10, 2007.

[10] S. J. Breckenridge, R. A.; Katzberg, "Smart sensors for the 80's - the status of smart sensors," in *Sensor Systems for the 80's Conf.*, 1980.

[11] D. Wagner, "Resilient aggregation in sensor networks," in *SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*. New York, NY, USA: ACM, 2004, pp. 78–87.

[12] A. Kansal, S. Nath, J. Liu, and F. Zhao, "SenseWeb: An infrastructure for shared sensing," *IEEE Multimedia*, vol. 14, no. 4, pp. 8–13, 2007.

[13] M. Narasimha and G. Tsudik, "Authentication of outsourced databases using signature aggregation and chaining," in *DASFAA*, 2006, pp. 420–436.

[14] M. Bawa, A. Gionis, H. Garcia-Molina, and R. Motwani, "The price of validity in dynamic networks," *J. Comput. Syst. Sci.*, vol. 73, no. 3, pp. 245–264, 2007.

[15] D. L. Hall and J. Llinas, "An introduction to multisensor data fusion," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 6–23, 1997.

[16] Y. Liu and S. Das, "Information-intensive wireless sensor networks: potential and challenges," *IEEE Comm. Mag.*, vol. 44, no. 11, pp. 142–147, November 2006.

[17] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "TAG: A Tiny AGgregation service for ad-hoc sensor networks," in *OSDI*, 2002, pp. 131–146.

[18] M. Dam and R. Stadler, "A generic protocol for network state aggregation," in *RVK*, Linköping, Sweden, Jun. 2005.

[19] H. Chan, A. Perrig, and D. Song, "Secure hierarchical in-network aggregation in sensor networks," in *ACM CCS*, New York, NY, USA, 2006, pp. 278–287.

[20] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: security protocols for sensor networks," *Wirel. Netw.*, vol. 8, no. 5, pp. 521–534, 2002.

[21] C. Karlof, N. Sastry, and D. Wagner, "TinySec: a link layer security architecture for wireless sensor networks," in *ACM SenSys*, New York, NY, USA, 2004, pp. 162–175.

[22] W. Trappe, Y. Zhang, and B. Nath, "MIAMI: methods and infrastructure for the assurance of measurement information," in *ACM DMSN*, New York, NY, USA, 2005, pp. 11–17.

[23] S. Zug, A. Dietrich, and J. Kaiser, "Detecting external measurement disturbances based on statistical analysis for smart sensors," in *IEEE ISIE*, 2009.

[24] S. Amin, X. Litrico, S. S. Sastry, and A. M. Bayen, "Stealthy deception attacks on water SCADA systems," in *ACM conf. on Hybrid systems: computation and control*, New York, NY, USA, 2010, pp. 161–170.

[25] E. Shi and A. Perrig, "Designing secure sensor networks," *IEEE Wirel. Netw.*, December 2004.

[26] "Worm targets industrial-plant operations," *Computer*, vol. 43, no. 11, pp. 15 –18, nov. 2010.

[27] H. Panetto and A. Molina, "Enterprise integration and interoperability in manufacturing systems: Trends and issues," *Computers in Industry*, vol. 59, no. 7, pp. 641 – 646, 2008.

[28] U.S. Department of Defense, "Trusted computer system evaluation criteria (orange book)," December 1985.

[29] R. Simha, B. Narahari, J. Zambreno, and A. Choudhary, "Secure execution with components from untrusted foundries," in *Proceedings of the Advanced Networking and Communications Hardware Workshop (ANCHOR)*, 2006.

[30] M. Bellare and C. Namprempre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm," London, UK, pp. 531–545, 2007.

[31] D. Gligoroski, S. Andova, and S. Knapskog, "On the importance of the key separation principle for different modes of operation," in *Information Security Practice and Experience*, ser. Lecture Notes in Computer Science, L. Chen, Y. Mu, and W. Susilo, Eds. Springer Berlin / Heidelberg, 2008, vol. 4991, pp. 404–418.

[32] Atmel, "Data sheet: Atmega48a / 48pa / 88a / 88pa / 168a / 168pa / 328 / 328p," 2010, retrieved aug. 2010.

[33] J. Daemen and V. Rijmen, "AES proposal: Rijndael," March 1999.

[34] M. Dworkin, "NIST special publication 800-38a: Recommendation for block cipher modes of operation: Methods and techniques," December 2001.

[35] J. Song, R. Poovendran, J. Lee, and T. Iwata, "RFC 4493: The AES-CMAC algorithm," June 2006. [Online]. Available: http://tools.ietf.org/html/rfc4493

[36] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers," *Commun. ACM*, vol. 21, no. 12, pp. 993–999, 1978.

[37] R. Anderson and M. Kuhn, "Tamper resistance – a cautionary note," in *USENIX Workshop on Electronic Commerce Proceedings*, 1996, pp. 1–11.

[38] R. J. Anderson and M. G. Kuhn, "Low cost attacks on tamper resistant devices," in *Intl. Workshop on Security Protocols*. London, UK: Springer-Verlag, 1997, pp. 125–136.

[39] K. Kursawe, D. Schellekens, and B. Preneel, "Analyzing trusted platform communication," in *ECRYPT Workshop, CRASH*, 2005, p. 8.

[40] S. Roy, M. Conti, S. Setia, and S. Jajodia, "Securely computing an approximate median in wireless sensor networks," in *ACM SecureComm*, New York, NY, USA, 2008, pp. 1–10.

[41] M. Garofalakis, J. Hellerstein, and P. Maniatis, "Proof sketches: Verifiable in-network aggregation," *IEEE ICDE*, pp. 996–1005, April 2007.

[42] S. Nath, P. B. Gibbons, S. Seshan, and Z. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," *ACM Trans. Sen. Netw.*, vol. 4, no. 2, pp. 1–40, 2008.

[43] R. Brooks, P. Ramanathan, and A. Sayeed, "Distributed target classification and tracking in sensor networks," *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1163 – 1171, aug. 2003.

[44] T. Winkler and B. Rinner, "Securing embedded smart cameras with trusted computing," *EURASIP J. on Wireless Comm. and Netw.*, 2011.

[45] K. V. Jónsson and Ý. Vigfússon, "Securing distributed aggregation with trusted devices," in *NordSec*, 2011.

[46] G. Avoine and S. Vaudenay, "Fair Exchange with Guardian Angels," in *WISA*, ser. Lecture Notes in Computer Sciences, 2003, pp. 188–202. [Online]. Available: http://www.springer.com/

[47] M. T. Dashti, "Optimistic fair exchange using trusted devices," in *Proceedings of the 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems*, ser. SSS '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 711–725.

[48] D. Levin, J. R. Douceur, J. R. Lorch, and T. Moscibroda, "TrInc: small trusted hardware for large distributed systems," in *USENIX NSDI*, Berkeley, CA, USA, 2009, pp. 1–14.

[49] M. Fort, F. Freiling, L. D. Penso, Z. Benenson, and D. Kesdogan, "TrustedPals: Secure multiparty computation implemented with smart cards," in *ESORICS*, 2006.

[50] A. Dua, N. Bulusu, W.-C. Feng, and W. Hu, "Towards trustworthy participatory sensing," in *USENIX HotSec*, Berkeley, CA, USA, 2009, pp. 8–8.

[51] S. Saroiu and A. Wolman, "I am a sensor, and I approve this message," in *HotMobile '10: Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications*. New York, NY, USA: ACM, 2010, pp. 37–42.

[52] K. Kursawe, "From trusted systems to the smart grid," Presentation, KTH Royal Institute of Technology, January 2011.