A Flexible Approach to Embedded Network Multicast Authentication

Chris Szilagyi
ECE Department
Carnegie Mellon University
szilagyi@cmu.edu

Philip Koopman
ECE Department
Carnegie Mellon University
koopman@cmu.edu

ABSTRACT

Distributed embedded systems are becoming increasingly vulnerable to attack as they are connected to external networks. Unfortunately, they often have no built-in authentication capability. Multicast authentication mechanisms required to secure embedded networks must function within the unique constraints of these systems, making it difficult to apply previously proposed schemes. We propose an authentication approach using message authentication codes which exploits the time-triggered nature of many embedded systems by putting only a few authentication code bits in each message, and by requiring authentication to be confirmed by the correct reception of multiple messages. This approach can work for both state transition commands and reactive control messages, and enables a tradeoff among per-message authentication cost, application-level latency, and the probability of induced system failure. Authentication parameters can be tuned on a per-message basis while satisfying typical wired embedded network constraints.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: Security and Protection

General Terms

Design, Reliability, Security.

Keywords

Distributed Embedded Systems, Networks, Security, Multicast, Authentication, Controller Area Network, CAN, FlexRay, Time-Triggered Protocol, TTP, In-vehicle, Real-time.

1. INTRODUCTION

Distributed embedded network protocols such as Controller Area Network (CAN) [3], FlexRay [1], and Time-Triggered Protocol (TTP) [15] are used in a wide variety of safety-critical applications. While these wired network protocols have been developed primarily for in-vehicle automotive networks, they are also seen in aviation, robotics, and industrial automation systems. Safety, reliability, performance, and cost have traditionally been the primary concerns in these systems.

Wired embedded network protocols are, for the most part, not designed with security in mind. This is largely because in the past embedded networks were isolated from the Internet, and could only be attacked by someone having direct physical access to the network. But now these embedded networks are susceptible to attacks, just as enterprise networks are, because manufacturers are incorporating connectivity to the Internet or wireless networks [14]. If an attacker corrupts even a single node via an external network or other method of attack, they will gain access to the internal safety-critical traffic on the wired embedded network. Wired embedded network protocols typically do not include built-in support for authenticating transmitters, restricting the messages transmitters can send, encrypting message payloads, or preventing denial of service (DoS) attacks. Wolf et al. [28] illustrate a variety of attacks on these distributed embedded network protocols, focusing on attacks which will disable the network.

In this paper we focus upon providing message authentication for wired embedded control networks such as CAN, TTP, and FlexRay. Because these protocols do not incorporate authentication, they are vulnerable to masquerade and replay attacks [27]. A masquerade attack occurs when an entity sends a fraudulent message identifying itself as another legitimate node. Replay attacks occur when an old message is retransmitted and accepted as a fresh message. Embedded network protocols provide means to identify the sender node, and incorporate error detection techniques. However, these techniques do not prevent a malicious entity (or non-malicious defective software) from masquerading as a legitimate node or replaying messages within the network. Masquerade attacks can be performed by changing an identifier field and recalculating error checking codes, or by broadcasting during another node's designated time slot in a Time Division Multiple Access (TDMA) protocol [18]. Replay attacks can be performed simply by recording a message and resending it in a similar fashion. TTP may be less vulnerable because the sender identity is implicit in the time slot being used. An adversary may need to exert additional effort to overcome this characteristic.

An attacker with physical access to an in-vehicle network and knowledge of message formatting and identifiers can easily send a spoofed message to unlock an automobile's car doors, start the engine, or operate other vehicle equipment. With wireless connectivity to a corrupted node, an attacker might activate a car's electronic parking brake while traveling on the highway, or shut off the headlights while traveling at night. Nilsson and Larson [19] demonstrate how such an attack can be performed on the CAN protocol through simulation.

While gateways between internal and external networks might help improve security, it seems plausible that attackers will be able to circumvent or penetrate gateways and obtain the ability to send messages on an internal embedded network. Preventing such attacks requires strong authentication of nodes as an additional layer of protection. This presents a particular challenge in distributed embedded networks, because any authentication scheme must support multicast authentication subject to the constraints of: resource limited nodes, small packet sizes, potentially high packet loss rates, and tight real-time deadlines.

We present an authentication method for distributed embedded networks which conforms to these common embedded system constraints. Our method allows the system designer to perform a tradeoff among per-message authentication cost, application level latency, and the probability of induced system failure. This is accomplished by appending truncated Message Authentication Codes (MACs) of only a few bits to each message. The time-triggered embedded applications we consider broadcast periodic updates of the values of system inputs and variables. This allows us to aggregate authentication from several messages before permitting an irrevocable alteration to the state of the system. Additionally, this approach allows us to reduce the probability of successful attacks on reactive control functions by making it difficult for attackers to forge enough messages in a short period of time to produce a system control failure.

This paper first identifies the impact of embedded network constraints on authentication, and describes why existing authentication schemes do not fully satisfy these constraints. We then describe an authentication scheme which conforms to the constraints, and takes advantage of existing properties of embedded network protocols. Additionally, we identify two types of embedded network messages, each requiring differing methods of authentication, and analyze the security of our scheme for each. Lastly, we introduce the notion of an engineering tradeoff among per-message authentication costs, application latency, and the probability of induced system failure.

2. Embedded Network Constraints

Distributed embedded networks are composed of a number of Electronic Control Units (ECUs). Each ECU performs a set of functions in the system. These ECUs are interconnected to form a network, and communicate using a protocol such as CAN, FlexRay, or TTP. In this paper, we will consider these protocols as they are commonly used in time-triggered applications. These protocols are among the most capable of those currently in use in wired embedded system networks. Many other protocols are even less capable, but have generally similar requirements and constraints:

- **Multicast Communications** - All messages sent on a distributed embedded network are inherently multicast, because all nodes within the embedded system need to coordinate their actions. Once a sender has transmitted a packet, all other nodes connected to the network receive the message. (In CAN, hardware performs message filtering at the receiver based on content.) Each packet includes the sender's identity, but does not include explicit destination information. [5] provides a description of multicast authentication issues along with some solutions. The configuration of the network is usually fixed at design time, with little or no run-time reconfiguration.
- **Resource Limited Nodes** - Processing and storage capabilities of nodes are often limited due to cost considerations at design time. For example, the S12XD series, produced by Freescale [2], is a family of 16-bit microcontrollers designed for use in general automotive body applications. These microcontrollers provide up to 32 kilobytes of RAM, 512 kilobytes of Flash memory, and four kilobytes of EEPROM, with a core operating frequency of 80 MHz. Flash memory is generally not written to except for software updates, so the EEPROM holds non-volatile application data. Any buffering and storage for authentication consume space in RAM, which is far more expensive and scarce than flash memory in such systems. Authentication mechanisms which require large amounts of processing power or storage in RAM may not be feasible. More powerful ECUs are infeasible for most nodes in the system, and many nodes are 8-bit ECUs with significantly smaller memories due to cost and power consumption considerations.
- **Small Packet Sizes** - Packet sizes are very small in embedded network protocols when compared to those in enterprise networks. These packets have maximum data payload sizes as small as eight bytes in the case of CAN, with the largest payloads for FlexRay and TTP being 254 bytes and 236 bytes respectively. Due to cost, signal integrity, and network node synchronization concerns, data rates are limited to 1 Mbit/sec for CAN and 10 Mbit/sec for TTP and FlexRay. Low-cost embedded networks can be orders of magnitude slower than that. Authentication should incur minimal bandwidth overhead.
- **Tolerance to Packet Loss** - Distributed embedded systems are subject to message blackouts due to environmental disturbances such as interference from large electric motors. High quality cable shielding is often impractical due to cost, size, and weight considerations. As such, authentication schemes must be tolerant to packet loss.
- **Real-Time Deadlines** - In real-time safety-critical systems, delays are not tolerated. Processes which cannot be completed within specified deadlines for the system cannot be used. Authentication of nodes must occur within a known time bound, with that bound being fast enough to match the physical time constants of the system being controlled (often on the order of tens or hundreds of milliseconds).

3. Related Work

This section describes the related work in multicast authentication and previous work in authentication for embedded networks.

3.1 Existing Multicast Authentication with Respect to Embedded Constraints

The multicast nature of distributed embedded communications makes authentication particularly challenging. Point-to-point cryptographic mechanisms, such as appending a MAC [27] to a message using a single key shared across all nodes, do not provide adequate authentication. If more than two nodes hold the same shared key, it becomes impossible to discern which one transmitted the message. Any receiver of a message could masquerade as the sender. For this reason, multicast authentication requires some form of asymmetry.

As a simple extension of the single shared key scheme to provide asymmetry, a sender could establish shared pair-wise keys with every other node. For each transmitted message, the sender would append a distinct MAC for each receiver to the message, providing strong authentication. A receiver would know that a message with a valid MAC could only have come from the sender, because those two nodes share a secret key and the receiver did not send the message. However, the bandwidth overhead of using full-size MACs makes this approach infeasible for embedded networks.

Public key cryptography using digital signatures is another asymmetric approach. While this could provide strong source authentication, digital signatures have very high processing and bandwidth overhead. The processing overhead alone makes it impractical for a resource constrained node to compute digital signatures for each message it sends. Several schemes suggest amortizing the cost of the digital signature over several packets [17][24][29][20]. But, known approaches may not be suitable when sending time-triggered embedded messages due to bandwidth overhead or intolerance to lost packets. Additionally, attackers can perform a denial of service attack, forcing a node to consume extra resources by processing arbitrary forged signatures, as noted in [22].

One-time digital signature schemes [21][10][8] allow senders to sign messages much faster than with traditional digital signatures by utilizing one-way hash functions. Unfortunately, one-time digital signatures can incur several kilobytes of authentication data per message. This makes them impractical for embedded networks with small packet sizes and time-triggered communication.

Canetti et al. [5] suggest a scheme which appends *k* MACs to each message, computed using *k* different keys. The keys are distributed amongst receivers such that at least *w* receivers must conspire in order to forge a message as the sender. This scheme requires computation of *k* MACs, and incurs considerable bandwidth overhead due to the attachment of these MACs. Additionally, this scheme is vulnerable to collusion.

Bergadano et al. [4] and the TESLA protocol [22] utilize time-delayed release of keys for authentication. By releasing keys at a pre-specified interval after a MAC is released, receivers can confirm the authenticity of the data from a sender. The released keys are computed using one-way hash chains. Resource constrained nodes may not have sufficient storage required for key chains to authenticate periodic messages in a time-triggered system. µTESLA [23], a version of TESLA for resource constrained sensor networks, limits the number of authenticated senders and utilizes a base station for communications. These options are not available for most distributed embedded real-time control systems, which use peer-to-peer wired networks. Although a node, such as an embedded gateway, might act as a base station, it also introduces an undesirable single point of failure. Additionally, one would expect that the gateway node would be the one node on the network most vulnerable to compromise from an external attacker, because it is the one node connected to external networks. Compromise of the base station node would compromise the security of the entire system. It would be desirable to have a practical approach that does not depend upon a base station.

3.2 Embedded Network Authentication

While there have been many publications on multicast authentication, little prior work has focused upon the requirements for authentication methods specifically for wired distributed embedded networks. There have been approaches which apply security to resource constrained wireless sensor networks such as SPINS [23] and TinySec [13]. However, those approaches are specifically designed for use in wireless networks, which have significantly different constraints than wired networks. Secure aggregation has also been used to reduce security overhead in both sensor networks [12][25] and Vehicular Ad-Hoc Networks (VANET) [26]. Those approaches focus on secure aggregation of data from multiple sensors in close geographic proximity rather than time-triggered messages in close temporal proximity.

Morris and Koopman [18] identify the potential for masquerade failures to be used to cause accidental or malicious failures, via allowing non-critical nodes to masquerade as higher criticality nodes. Additionally, they propose the use of several counter-measures of varying strengths to prevent masquerading failures between nodes of varying criticality. Their approach assumed an attack was due to a non-malicious software fault or was being made by an unsophisticated attacker, unfamiliar with cryptology.

Wolf et al. [28] provide an overview of the security vulnerabilities of various in-vehicle network protocols including Local Interconnect Network (LIN), Media Oriented System Transport (MOST), CAN, and FlexRay. These vulnerabilities primarily focus upon attacks which will disable the networks. Additionally, they state the need for confidentiality and authentication. Wolf et al. suggest the use of digital signatures or the asymmetric MAC scheme proposed by Canetti et al. [5] for authenticating sent packets along with gateways between individual in-vehicle networks. These authentication schemes may not be suitable for some distributed embedded networks, as discussed in Section 3.1.

There have been several publications demonstrating attacks on the authenticity of messages and nodes in embedded networks. Nilsson and Larson [19] detail the actions which an attacker may take, and demonstrate masquerade attacks on CAN using simulation. Additionally, they discuss the possibility of viruses transmitted over CAN and preventative measures. Hoppe et al. [11] and Lang et al. [16] demonstrate a combination of eavesdropping and replay attacks on CAN.

Lastly, Chávez et al. [6] propose using RC4 encryption to provide confidentiality on CAN buses. Chávez et al. dismiss authentication and non-repudiation as unnecessary in these networks, under the assumption that message identifiers and error detection provide sufficient confirmation of the sender's identity. Our work relaxes this assumption by assuming that sender identity can be forged.

4. Criticality Based Authentication

In order to provide multicast authentication on a per message basis for time-triggered communications, our approach uses truncated MACs. In time-triggered communications, each node periodically broadcasts the current state of each of its state variables and sensor inputs to the rest of the network. This information is often broadcast faster than the rate at which receivers must act upon this data in their control loops. This faster rate gives the system a degree of resilience to unexpected operating situations and message losses.

In our approach, when a node sends a message, it computes a MAC for each distinct receiving node in the network over the message and the current time (or TDMA round number) using a pair-wise shared secret key. Each MAC is truncated down to just a few bits, and appended to the message. (If there is concern that the low bits of the MAC are not sufficiently random, all bits of the MAC can be hashed. For example, XORing all MAC bytes together would create a condensed 8-bit version of the MAC.) By only using a few bits, one MAC per receiver can be placed in a packet, as illustrated in Figure 1. The receivers verify their respective MACs and signal an error if the MAC does not match the message in the current time interval. Nodes act upon authenticated messages depending upon the type of message received. The number of bits in each truncated MAC depend on a variety of factors, but could be as little as one bit per MAC.

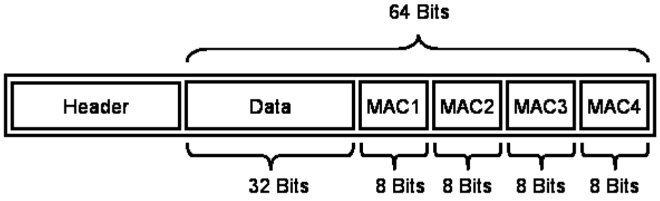

Figure 1. Example packet containing 32 bits of message data and four 8-bit MACs, for four receivers.

This allows the designers of the system to perform a tradeoff among the required amount of bandwidth they are willing to sacrifice for per-message authentication, application level latency, and the probability of induced system failure. This tradeoff is based upon the criticality level of the message and the message type. Criticality is related to the amount of physical change which can be exerted on the environment around the system, or potential for monetary loss or damages.

4.1 Message Types

We identify two types of messages in embedded networks with different requirements for authentication: state-changing messages, and reactive control messages.

4.1.1 State-Changing Messages

State-changing messages cause transitions within finite state machines in the system design, or cause discrete, discontinuous output changes in actuators. If an attacker successfully executes an undesired state change, the system must attempt to roll back to an earlier correct state to undo any damages. Depending on the action, such a roll-back may or may not be possible. For example, triggering a pyrotechnic that deploys an airbag is a discontinuous actuator state change that cannot be rolled back automatically.

For state-changing messages, nodes must receive a certain number of correctly authenticated consistent messages directing the state change before executing the action. The number of consecutively authenticated messages which must be received is proportional to the criticality of the message. For example, an attacker who managed to forge a message to turn on the four-way flashers of an automobile would only cause some confusion and irritation that is easily rolled back by turning the flashers back off. That state change might require only a few consecutive authenticated messages before the system accepted the state change commanded by those consecutive messages as valid. On the other hand, if an attacker successfully forged messages to unlock the doors and turn on the engine to facilitate car theft, the resulting damage could be greater. A receiver of those messages would wait to receive a larger quantity of correctly authenticated messages before accepting the state change as authentic.

4.1.2 Reactive Control System Messages

Reactive control system messages cause updates to continuous or ordered values in network nodes running feedback control loops. These loops often contain a low pass filter to actuator changes (implicit or explicit), such as physical inertia, which limits the possible impact of a single forged message. In the event of a single successfully forged message, this low pass filtering characteristic damps out the possible impact on the system. So long as a sufficiently small fraction of messages can be successfully forged, the system can either ride out disturbances or have time to notice an attack is taking place before significant damage has been done or the system has become unsafe.

So long as there are enough MAC bits used to keep successful forgeries sufficiently infrequent, nodes can authenticate each reactive control system message individually. An attacker would need to correctly forge many messages within some period of time to produce a potentially damaging physical output from an actuator. As the number of messages within a short time period required to produce a damaging output increases, the probability that an attacker can forge such a series of messages in a short enough period of time so as to induce a system failure decreases. If sufficient MAC bits are not available in each message to keep the probability of a successful forgery sufficiently low, then several messages in a row might be required to have correct MACs before a new actuator output value is accepted as valid.

4.2 Additional Properties

Our scheme provides three additional beneficial properties. First, each transmitted packet contains all authentication information for that packet. This allows some amount of authentication to be performed for every packet received. No buffering is required by the sender or receiver. Second, authentication information is fully contained within each individual packet, so our scheme is tolerant to packet loss. Lastly, this scheme has ideal resistance to node compromise, because an attacker can only masquerade as those nodes from which they have extracted key material.

4.3 Assumptions

Our approach makes three assumptions:

- Each sender has sufficient computational resources to compute one MAC per receiver per message that is sent. MACs can be computed relatively quickly, and the number of receivers is quite limited in embedded networks.
- The number of bits in a message is greater than the number of receivers of a message. Embedded networks typically incorporate a small number of nodes, usually fewer than 32. This allows authenticators for each receiver in the packet, leaving room for the message.

- Nodes use existing cryptographic one-way hash functions, such as SHA-256, and MAC functions, to implement authentication. We assume the underlying cryptographic primitives are secure. We do not rely on specific MAC or one-way hash functions to implement our scheme.

4.4 Attacker Model

We consider an active attacker model [27] in which an attacker may modify, inject, drop, or eavesdrop upon network traffic. Attackers may physically access the network lines, or access the network through a corrupted node. Attacks through corrupted nodes include connections from an external network through a gateway, malicious insider code, physically compromised devices, and malicious devices physically attached to the network.

Attackers accessing the network through corrupted nodes will have access to the key material in those nodes. Regardless of the key material possessed by the attacker, they must not be able to masquerade as any node they do not control to perform a successful attack, except with some negligible probability.

We constrain the attacker to one forgery attempt per message, since receivers only accept a single message per time slot in a time-triggered application.

It is likely that any single successful forgery attempt will only succeed in fooling a subset of receiving nodes, because each receiving node bases its acceptance of a message on a different MAC value. Whether it is possible to successively fool different nodes one at a time to accomplish a global malicious state change depends on the details of system design. In particular, doing this would require finding an enduring state change that can be accomplished with only a few successfully forged messages per node, and that does not revert to a non-malicious state during the time it takes to successfully messages to other nodes. Commonly used fault containment mechanisms such as group membership would form strong countermeasures to such divide-and-conquer attacks on nodes.

Lastly, it should be noted that this scheme does not seek to protect against DoS attacks. Wolfe et al. [28] surveys numerous existing vulnerabilities in these networks to simple DoS attacks. This scheme presents additional opportunities for DoS attacks, such as intentionally sending incorrect MAC values, but in general does not make the DoS issue worse than it already is.

5. Criticality Based Authentication Process

This section describes the process which the wired embedded network nodes will use to provide authentication.

5.1 Key Initialization

A node establishes shared secret authentication keys with all other nodes at time of installation. This can be accomplished by having maintenance or factory personnel program each node with the respective shared secret keys when the node is installed. This method is not ideal, since it requires additional work by personnel to establish the keys, and places a large amount of trust in these personnel. Alternately, another approach is to provide each node with a public and private Diffie-Hellman [7] key pair, which has been digitally signed by the manufacturer's secret key. Each node also has the manufacturer's public key. At time of installation, the nodes could exchange their Diffie-Hellman public keys and certificates. Each pair of nodes then authenticates the certificates and uses the Diffie-Hellman key exchange protocol to compute a shared secret key for authentication.

For a system with n nodes, this scheme will require establishing $O(n^2)$ keys. While this overhead is high, it is incurred only once at time of installation, while the system is inactive. Embedded networks have very stable hardware configurations, which often last for months or years. Thus, a one-time key distribution cost is a minor concern in most situations.

Additionally, in a typical embedded system, all nodes wired to the network are known at design time. It is reasonable to assume a node will know the standard configuration and what nodes comprise the group it is communicating with. This is in contrast to enterprise networks, where network nodes are expected to change constantly.

5.2 Replay Protection

We use time synchronization to prevent replay attacks. At system startup, nodes perform pair-wise synchronization of clocks to some predefined granularity, which might be on the order of the time it takes to transmit a full round of all messages. A network wide synchronization is not necessary, because pair-wise MACs are used for authentication. Pair-wise synchronization can be accomplished through the use of a secure time synchronization protocol such as Secure Pair-wise Synchronization [9]. Experimentation in [9] demonstrates time synchronization to a time tick granularity on the order of microseconds. In a distributed embedded network, synchronization to the nearest message round is often adequate, which is often on the order of tens or hundreds of milliseconds, and might be a service built in to the communication protocol.

5.3 Run-Time Authentication and Trade Offs

In order to provide multicast authentication, each pair of nodes must establish a shared secret key, and securely synchronize their clocks. For each message which is sent, the node first computes a MAC for each receiver using the shared secret keys. For a receiver i, the sender computes MAC M_i, which is computed over the message m and synchronized time T_i, using shared key K_i. "||" denotes concatenation.

$$M_i \quad \leftarrow \quad \mathrm{MAC}_{Ki}(m \parallel T_i)$$

Each MAC is truncated, and b lower order bits of each MAC are appended to the message. For n receivers, the data payload of the packet consists of: $m \parallel [M_1]_b \parallel [M_2]_b \parallel \ldots \parallel [M_{n-1}]_b \parallel [M_n]_b$, where [] denotes truncation. Using a few bits per message reduces the amount of message overhead so that all MACs fit within a single packet. As these time-triggered messages are received, the authentication information accumulates, granting greater confidence in the authenticity of the messages. Each time-triggered message is verified independently of all other messages. Fooling a receiver once has minimal impact, because an injected failure is cleared unless the attacker continues to successfully forge messages.

5.3.1 State-changing Message Verification

For state-changing messages, a receiver waits until a predefined number of consecutive messages are received before executing the received command. For a receiver which waits for x correctly authenticated messages to arrive, the probability of a successful forgery is equal to 2^{-xb}. The probability of a successful forgery drops exponentially as the number of bits *b* of the MAC increases, or the number of messages required *x* increases (Figure 2). The system designer trades increased bandwidth and latency for lower probability of induced system failure. Additionally, there will be a limit on the value of *x*, based upon the maximum tolerated latency for the message.

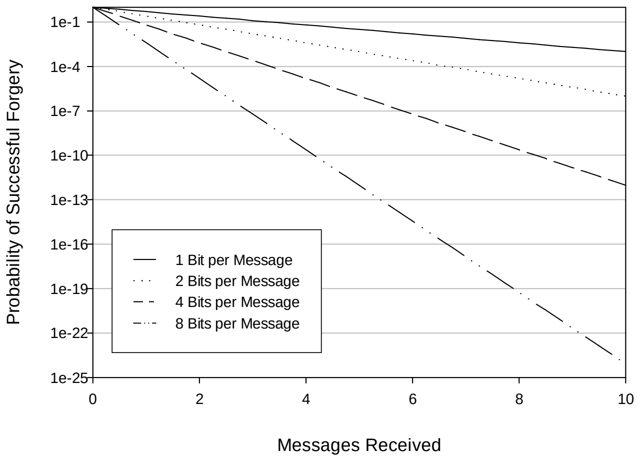

Figure 2. Probability of successful forgery for one, two, and eight bit MACs over ten messages.

5.3.2 *Reactive Control Message Verification*

For reactive control system messages, the probability of a successful forgery for any individual message to a particular receiver is only equal to 2^{-b}. While this does not provide highly secure authentication for an individual message, each successfully forged message will only cause some increment of physical change produced by the receiving node. In order to produce a successful attack, the attacker must forge multiple packets within some time period. If the attacker must forge *y* consecutive packets within this time period, the probability that the attacker succeeds is equal to 2^{-yb}. The probability that an attacker will be able to produce dangerous outputs decreases exponentially as the required number of correctly forged messages increases. The designer selects the value of *b* for this type of message based upon the amount of physical change produced per message, so that the product *yb* is sufficiently large for the probability of system failure to be considered acceptable. This approach supports trading increased bandwidth for reduced probability of system failure.

6. Conclusions

Distributed embedded networks are becoming increasingly vulnerable to masquerade and replay attacks due to increased connectivity, creating a need for authentication. A significant challenge is that solutions for multicast authentication must take into account the unique constraints of these systems. We present a method based on truncated MACs to authenticate state-changing and reactive control system messages, along with associated analysis and tradeoffs. While this method provides authentication which is loss tolerant, requires no message buffering, and has ideal resistance to node compromise, this scheme still requires bandwidth overhead that scales linearly with the number of receivers, computation of one MAC per receiver for each message, and a limit on the number of receivers in practical implementations. In the future, we intend to present results from attacks on real systems, provide methodologies for engineers to perform tradeoffs in embedded network authentication, improve scalability, and further reduce bandwidth and computation overhead to provide even more flexible authentication solutions for distributed embedded networks.

7. Acknowledgements

Support for this work was provided by the General Motors Collaborative Research Laboratory at Carnegie Mellon University.

8. References

[1] FlexRay Consortium. FlexRay Communications System Protocol Specification, Version 2.1, Revision A, December 2005.

[2] Freescale Semiconductor. S12XD Product Summary Page. Retrieved June 2008 from http://www.freescale.com/.

[3] R. Bosch GmbH, CAN Specification, Version 2, September 1991.

[4] F. Bergadano, D. Cavagnino, and B. Crispo. Individual Single-Source Authentication on the MBONE. In *Proc. of the 2000 IEEE Int'l Conf. on Multimedia and Expo*, volume 1, pages 541–544. IEEE, 2000.

[5] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast security: a taxonomy and some efficient constructions. In *INFOCOM '99: Proc. 18th Annual Joint Conf. of the IEEE Computer and Communications Societies*, volume 2, pages 708–716. IEEE, 1999.

[6] M. L. Chavez, C. H. Rosete, and F. R. Henriquez. Achieving Confidentiality Security Service for CAN. In *CONIELE-COMP '05: Proc. of the 15th Int'l Conf. on Electronics, Communications and Computers*, pages 166–170. IEEE, 2005.

[7] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, vol. 22, pages 644-654, 1976.

[8] S. Even, O. Goldreich, and S. Micali. On-line/off-line digital signatures. In *CRYPTO '89: Proc. on Advances in Cryptology*, pages 263–275. Springer-Verlag, 1989.

[9] S. Ganeriwal, S. Čapkun, C.-C. Han, and M. B. Srivastava. Secure time synchronization service for sensor networks. In *WiSe '05: Proc. of the 4th ACM Workshop on Wireless Security*, pages 97–106. ACM, 2005.

[10] R. Gennaro and P. Rohatgi. How to Sign Digital Streams. In *CRYPTO '97: Proc. of the 17th Annual Int'l Cryptology Conf. on Advances in Cryptology*, pages 180–197. Springer-Verlag, 1997.

[11] T. Hoppe and J. Dittman. Sniffing/Replay Attacks on CAN Buses: A simulated attack on the electric window lift classified using an adapted CERT taxonomy. In *Proc. of the 2nd Workshop on Embedded Systems Security (WESS)*, 2007.

[12] L. Hu and D. Evans. Secure Aggregation for Wireless Networks. In *Proc. of the 2003 Symposium on Applications and the Internet Workshops*, pages 384–394. IEEE, 2003.

[13] C. Karlof, N. Sastry, and D. Wagner. TinySec: a link layer security architecture for wireless sensor networks. In *SenSys*

’04: Proc. of the 2nd Int’l Conf. on Embedded Networked Sensor Systems, pages 162–175. ACM, 2004.

[14] P. Koopman, J. Morris, and P. Narasimhan. Challenges in Deeply Networked System Survivability. *NATO Advanced Research Workshop on Security and Embedded Systems*, pages 57–64, 2005.

[15] H. Kopetz and G. Grunsteidl. TTP - A time-triggered protocol for fault-tolerant real-time systems. In *Proc. of the 23rd Int’l Symposium on Fault-Tolerant Computing*, pages 524–533, 1993.

[16] A. Lang, J. Dittman, S. Kiltz, and T. Hoppe. Future Perspectives: The car and its IP address - A potential safety and security risk assessment. In *Proc. of the 26th Int’l Conf. on Computer Safety, Reliability and Security (SAFECOMP)*, pages 40-53. Springer-Verlag, 2007.

[17] S. Miner and J. Staddon. Graph-Based Authentication of Digital Streams. In *SP ’01: Proc. of the 2001 IEEE Symposium on Security and Privacy*, pages 232–246. IEEE, 2001.

[18] J. Morris and P. Koopman. Critical Message Integrity Over A Shared Network. *5th IFAC Int’l Conf. on Fieldbus Systems and their Applications*, pages 145-151, 2003.

[19] D. Nilsson and U. Larson. Simulated Attacks on CAN Buses: Vehicle Virus. *5th IASTED Asian Conf. on Communication Systems and Networks*, 2008.

[20] J. M. Park, E. K. P. Chong, and H. J. Siegel. Efficient Multicast Packet Authentication Using Signature Amortization. In *SP ’02: Proc. of the 2002 IEEE Symposium on Security and Privacy*, pages 227–240. IEEE, 2002.

[21] A. Perrig. The BiBa one-time signature and broadcast authentication protocol. In *CCS ’01: Proc. of the 8th ACM Conf. on Computer and Communications Security*, pages 28–37. ACM, 2001.

[22] A. Perrig, R. Canetti, J. Tygar, and D. Song. The TESLA Broadcast Authentication Protocol. *RSA CryptoBytes*, vol. 5, pages 2-13, 2002.

[23] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler. SPINS: security protocols for sensor networks. *Wireless Networks*, vol. 8(no. 5):pages 521–534, 2002.

[24] A. Perrig, J. D. Tygar, D. Song, and R. Canetti. Efficient Authentication and Signing of Multicast Streams over Lossy Channels. In *SP ’00: Proc. of the 2000 IEEE Symposium on Security and Privacy*, pages 56–73. IEEE, 2000.

[25] B. Przydatek, D. Song, and A. Perrig. SIA: Secure Information Aggregation in Sensor Networks. In *SenSys ’03: Proc. of the 1st Int’l Conf. on Embedded Networked Sensor Systems*, pages 255–265. ACM, 2003.

[26] M. Raya, A. Aziz, and J.-P. Hubaux. Efficient secure aggregation in VANETs. In *VANET ’06: Proc. of the 3rd Int’l Workshop on Vehicular Ad Hoc Networks*, pages 67–75. ACM, 2006.

[27] Schneier. *Applied Cryptography (2nd ed.): Protocols, Algorithms, and Source Code in C.* John Wiley & Sons, Inc., New York, NY, USA, 1995.

[28] M. Wolf, A. Weimerskirch, and C. Paar. Security in Automotive Bus Systems. *Workshop on Embedded Security in Cars*, 2004.

[29] C. K. Wong and S. S. Lam. Digital Signatures for Flows and Multicasts. In *ICNP ’98: Proc. of the 6th Int’l Conf. on Network Protocols*, pages 198–209. IEEE, 1998.