Techniques for Securing Substation Automation Systems

David Guidry[1], Mike Burmester[1], Xiuwen Liu[1], Jonathan Jenkins[1], Sean Easton[1], Xin Yuan[1], and Josef Allen[2]

[1] Florida State University, Tallahassee, FL 32306, USA
guidry,burmester,liux,jenkins,easton,xyan@{cs.fsu.edu}
[2] Oak Ridge National Laboratory, P.O. Box 2008, Oak Ridge, TN 37831, USA
allenjd@ornl.gov

Abstract. Most critical infrastructure systems can be modeled as cyber-physical systems whose cyber components control the underlying physical components so as to optimize specified system objectives based on physical properties, physical constraints, and the current and estimated state of the system. Such systems usually require supports for security and performance guarantees: wrongly received or missed commands can render the entire system unstable. Yet, securing cyber-physical systems with heterogeneous components is still an open and challenging problem. In this paper, we propose techniques for resilient substation automation of power utility systems with security based on the trusted computing paradigm. By using trusted platform module (TPM)-enabled components and a novel access control structure that enforces *need-to-get* (availability) policies, we show how to develop IEC/TR 61850-90-5 compliant substation automation systems that are resilient. We demonstrate the feasibility of our approach by analyzing and experimenting with an open source IEC/TR 61850-90-5 implementation.

1 Introduction

Due to the potential significant impact of critical infrastructure failures on our society, it is essential that such systems are properly secured. Most critical infrastructures such as the electricity grid can be modeled as cyber-physical systems, where cyber components and physical components are closely coupled. One of the key characteristics of such systems is that timing is essential, where "correct" commands issued at the "wrong" time relative to the state of the underlying physical system may have disastrous consequences.

Developing efficient and secure cyber-physical systems for controlling and monitoring the electricity grid is extremely challenging due to the propagation speed of electricity, which imposes stringent time constraints for processing and communicating data and control commands. Traditional security techniques are developed for cyber systems, where the criteria are confidentiality, integrity and availability, in *this* order. Most approaches do not address the availability issue, which renders them not directly applicable for securing cyber-physical systems such as the electricity grid. On the other hand, critical infrastructures also offer unique features that can be used to enhance security. For example, by their very nature, critical infrastructures have components that require physical verification. This fact can be utilized to enhance security beyond what could be achieved in systems with cyber-only components.

In this paper, we propose techniques for securing substation automation systems, a fundamental building block of the electricity grid, that utilize the unique characteristics of electricity grids. Our scheme incorporates the trusted computing paradigm supported by trusted-platform module (TPM)-enabled components, and utilizes an access control structure for need-to-get policies to meet the stringent time constraints of the system. Built-on an open source implementation of IEC/TR 61850-90-5, we have established a networked testbed to test and evaluate the proposed techniques. The feasibility of our approach is demonstrated by analyzing and experimenting with the open source IEC/TR 61850-90-5 implementation. The contributions of this work include the following:

- We show how the trusted platform module (TPM) technology can be used to secure cyber-physical applications.
- We propose an access control structure that enforces *need-to-get* policies for cyber-physical systems where availability is a primary concern.
- We show how IEC/TR 61850-90-5 compliant resilient substation automation systems can be developed based on the proposed techniques.
- We demonstrate the feasibility of our approach by analyzing and experimenting with an open source IEC/TR 61850-90-5 implementation recently released by SISCO [10].

The rest of the paper is organized as follows. In Section 2, we briefly introduce the Technical Report IEC/TR 61850-90-5, a newly introduced transmission protocol specification standard for substation automation. We describe the trusted computing paradigm in Section 3. A novel access control structure that enforces need-to-get policies is presented in Section 4. Section 5 discusses how resilient cyber-physical systems can be developed based on the proposed approach. Section 6 describes our testbed and experimental results. We conclude the paper in Section 7.

2 IEC/TR 61850-90-5

IEC/TR 61850-90-5 [6] is currently a technical report for communication networks and systems that support power utility automation. It provides substation automation for heterogeneous intelligent electronic device (IED) platforms. The objective is to achieve low cost wide area monitoring, control and protection (WAMCAP) for power utility systems. IEC/TR 61850-90-5 specifies the use of the IP transport protocol (either IP multicast or unicast) for exchanging data between phasor measurement units (PMU) and phasor data concentrators (PDC). Exchanged data is encapsulated in IP packets whose payloads are either sampled values from the PMUs (called *Sample Value (SV) packets*), or control packets (called *Generic Object Oriented Substation Events (GOOSE) packets*). Such packets allow the reporting/control information to be distributed in wide area network environments. To multicast SV or GOOSE packets over IP, the packet profiles are adapted, resulting in *multicast* SV *control blocks* (MSVCB) and GOOSE *control blocks* (GoCB). IEC/TR 61850-90-5 does not address security issues.

The Technical Specifications IEC 62351-6 document [7] addresses some security issues of networked substations: it specifies appropriate cryptographic mechanisms for

network applications, and addresses the security for some IEC/TR 61850-90-5 protocols. However security, which is critical for the safe operations of networked substations, is not fully addressed. True automation for power utility systems requires comprehensive security. The IEC/TR 61850-90-5 will need to offer real-time security, with assured and trusted communication. In particular, it will need to address comprehensively *availability*, *integrity* (and *authentication*), and when needed *confidentiality*. It must provide confidence in message delivery, in the authenticity of the sender and receiver, and in the integrity of data sent before, during, and after unexpected or extreme events, e.g. faults, rolling blackout, voltage collapse. Key to this, is assured communication.

There are several security issues that must be addressed before an effective and open standard can be established so that vendors can provide WAMCAP. In this paper our goal is to address this issue. We present techniques that support assured and trusted communication via trusted computing technologies with the focus on the availability aspect of security that is often ignored in traditional security mechanisms.

Figure 1 illustrates a substation with IEC/TR 61850-90-5 products. The IEDs communicate with each other and outside the substation through the router while IED communication to high voltage (HV) equipment is through specialized components. The router is in charge of the trust of the devices on its LAN. If a device is untrustworthy then the router will prevent any individual and/or substation to communicate with the device, causing it to be quarantined. The router is the substation representative for all trusted communication.

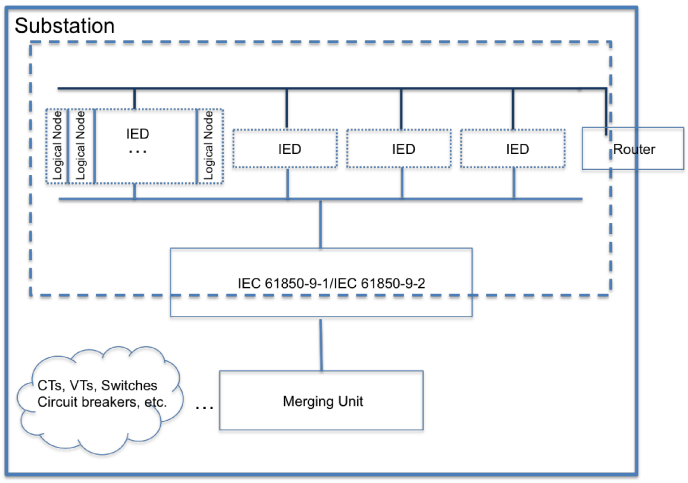


Fig. 1. An illustration of a substation in which the communication between IEDs is IEC/TR 61850-90-5 compliant.

To achieve real-time security it is important that both the reporting SV packets and the control GOOSE packets are received in real-time, when they are needed. We define the *active time* of a packet to be the time it takes: (a) to prepare the packet (*e.g.*, from synchrophasor data), plus (b) the time it takes to transmit it, plus (c) the time it takes

to receive and verify it. If the packet is dropped then its active time is ∞. IEC/TR 61850-90-5 specifies a threshold for the active time of packets that should be less than 4 milliseconds.

3 Trusted Computing

Trusted Computing (TC) as defined by the Trusted Computing Group (TCG) [11] is a technology for securing distributed systems by using *real-time trust engines* (called *roots of trust*), that: (a) attest to the integrity of the system (*remote attestation*), (b) protect keys and stored data (*sealed storage*) and (c) support cryptographic mechanisms.

The following technical details define the structure of a TC system. Key to the protection of the system is a trusted platform module (TPM), a real-time trust engine that acts as a *root of trust* for all security operations of the system. The functionality requirement for the TPM is to guarantee that the system will behave in a well defined manner (as expected) for the intended purpose.

3.1 The Trusted Platform Module

A Trusted Platform Module (TPM) [13] has two basic capabilities: remote attestation and sealed storage, and provides a range of cryptographic primitives including: a random number generator; hashing functions; asymmetric encryption/decryption; two unique asymmetric non-migratable key pairs (set at the time of manufacture): an attestation identity key pair for signing data originating at the TPM, and an endorsement key pair for decrypting owner authorization data and messages associated with the attestation key creation; symmetric keys to bind small amounts of data (typically keys) and to authenticate transport sessions. The TPM also has a small amount of storage (mainly for keys).

There are three roots of trust in a TPM: a root of trust for measurement (RTM) for making reliable integrity measurements, a root of trust for storage (RTS) to protect keys and data entrusted to the TPM, and a root of trust for reporting (RTR) to (a) expose shielded locations for storage of integrity measurements and (b) attest to the authenticity of stored values (based on trusted platform identities).

Security is based on an integrity protected boot-up process in which executable code and associated configuration data is measured before it is executed—this requires that a hash of the BIOS code is stored in a platform configuration register (PCR). For remote attestation the TPM uses an attestation identity key (AIK) to assert the state of its current software environment and its state to a third party—by signing its current PCR values. For sealed storage, encryption/decryption/authentication keys are released from protected storage, conditional on the current software state (using the current PCR values).

Figure 2 illustrates the components of a TPM that support RTR (reporting) and RTS (storage) engines. The I/O manages data flows over the communication buses (internal and external) by encrypting/decrypting data. Non-volatile storage is used for persistent keys. The PCR can be implemented in volatile or non-volatile memory. The Program Code contains firmware for measuring platform devices. Opt-In is used to customize

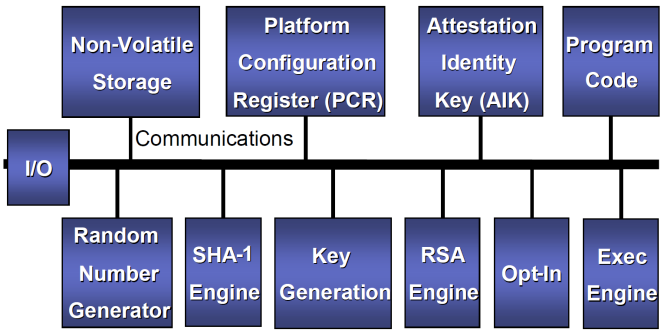


Fig. 2. The components of a Trusted Platform Module.

TPM modules. Exec Engine runs program codes: it performs TPM initialization and takes measurements. The other components were discussed earlier.

A TPM can be implemented as a hardware, software or an embedded software device. The TCG requires that TPM implementations are physically protected from tampering. This includes binding the TPM to other physical parts of the platform (*e.g.*, the motherboard) so it cannot be transferred to other platforms. TPM implementations have a unique (asymmetric) endorsement key (EK) that is created at the time of manufacture of the TPM. This key, is used together with an endorsement key credential (EKC), to secure all transactions and assert that the holder of the private part of the EK is the corresponding TPM, thus conforming to the TCG specifications. Before a TPM can be used a *take ownership* procedure must be performed to bind the usage of the TPM to a specific application device/user. For more details on the architecture of TPM the reader is referred to [13].

3.2 TC compliant Cyber-Physical Systems

Cyber-Physical Systems (CPS) combine computation and communication with physical processes. They monitor and control physical processors in real time, usually with feedback loops. Typically, a Control Center manages a network of CPS (substations).

We consider CPS architectures in which a TPM microprocessor is used to protect trusted applications. We distinguish three components.

1. **The secure kernel** that contains trusted hardware and the TPM that provides the necessary cryptographic mechanisms to platforms for program attestation and sealed storage via the RTR, RTM and RTS engines. The RTM engine is also used for trusted boot-up and the RTS engine for remote attestation. The kernel logically separates execution from upper layer applications.
2. **The sealed storage** used for storing sensitive data and keys: access to it requires the release of keys from the TPM (via the RTS engine).
3. **The Sensing and Control unit and the Applications**. All internal communication between the Sensing and Control devices and the Applications, and external communication with other CPS is protected by using the RTR and RTS engines (point-to-point encrypted and authenticated). Access control policies are enforced using credentials and certificates.

3.3 The Trusted Network Connect

The Trusted Network Connect (TNC) [12] is a TC platform interoperability architecture for trusted access control that is based on the TPM module. What distinguishes the TNC from other interoperability architectures is the requirement that the configuration of the OS of the client and server (and associated configuration data) is checked prior to a communication channel being established. More specifically, a (trusted) link between a client and server is established only if:

1. The identity of the client and the server is trusted. For this purpose we need a Public Key Infrastructure in which Certifying Authorities issue certificates that establish trust-links between a Root Authority (a Trusted Third Party) and the TPMs of the client/server.
2. The client is allowed access to the server. For this purpose an Access Control System (typically RBAC [9]) with credentials is used.
3. The identities of the client and server are authenticated. This requires a root of trust engine to be invoked on the TPM of both sides to release the required keys for a handshake protocol [12] to be executed. The TPM will only release keys if the current configuration state of the OS of the parties allow for this.
4. The handshake protocol is properly executed. The TPM will enforce the integrity of communicated data (by releasing appropriate message authentication keys), and depending on the application the confidentiality of communicated data.

3.4 Trusted Substation Automation Systems

The network of a power utility system (a *region*) has several *zones*, each one of which has several substations. For trusted interoperability, the TNC platform is used.

Substation-to-substation communication within a zone is over an intranet (controlled by the power utilities). Secure substation automation system (SAS) communication is crucial since it allows one substation to contact another substation to increase the output of backup generator after a primary generator is already running at maximum capacity. For secure station-to-station communication the communication must be trusted: (a) the sender must trust its own components before sending any information, and (b) the receiver must ensure that the information sent comes from a trusted sender. This is achieved by having each substation of the SAS to be TC-compliant. The same approach is used for trusted zone-to-zone communication. However here the communication is over an open network (the Internet).

4 An Access Control Structure for Need-to-Get Policies

Access control systems are trust infrastructures for managing the resources of distributed systems in a secure way. Security typically involves *confidentiality, integrity* and/or *availability*. For cyber systems (such as computer or network systems) the security focus is typically privacy and/or integrity. The Bell-LaPadula [1] model describes an access structure that captures *need to know* security requirements (confidentiality).

The Biba [2] access control model captures integrity requirements. However for cyber-physical systems, and in particular critical infrastructure systems, the primary security concern is typically availability. [1]

In this section we present an access control structure for managing the resources of a cyber-physical system, for which the primary security goal is availability, with secondary goal integrity, and tertiary goal confidentiality. This captures *need to get* policies as opposed to *need to know* policies. Our access control structure is based on information flow models (see e.g., [5]), adapted for our particular application.

We define an *access control communication structure* by a tuple $\mathcal{A} = (N, L, P, \succeq)$, where N is a set of nodes, L a set of links, P a partially ordered set of *security classes* (called *priorities* or *congestion levels*), and "$\succeq$" a *flow relation* defined on pairs of security classes. For this relation, a packet m with priority $p \in P$ is allowed to be transmitted over a channel of L with congestion $p' \in P$ if, and only if, $p \succeq p'$. That is, the network system will transmit m via a network channel if, and only if, the priority of m dominates the congestion level of the channel.[2] We take P to be a linearly ordered set $\{\mathsf{lo} = \ell_1, \ldots, \mathsf{hi} = \ell_k\}$,[3] and the *need to get* (availability) policy is: *packets are transmitted via a network channel if, an only if, their priority dominates the congestion level of the channel*. Observe that the network will drop any packets that are not transmitted. For substation automation systems (SAS), the priorities of packets are static, defined by their profile, while the congestion level of the links is dynamic, defined by the prevailing traffic flow (see Section 4.2).

Remark 1. The following argument justifies our approach. Assume that the network system is designed to guarantee communication availability when only (authorized) packets with hi priority are transmitted. This requires that there is a bound on the flow of such packets and that the network system is designed to have sufficient redundancy to cope with this traffic. Since SAS network systems are: (a) TPM-enabled (Section 3.1) and (b) private (controlled by the power utility organization (Section 3.3)), availability is only impacted when authorized flow traffic is congested, resulting in packet bits getting lost/corrupted. When this happens the congestion level of the network channels is raised, in the extreme to hi: this restricts the traffic to hi priority packets. As a consequence the transmission time is reduced and we get real-time availability.

For our application the communication network of a SAS contains two types of traffic: *reporting* traffic in which phasor measurements collected by phasor measurement units (PMU) are sent using sample value (SV) packets to a control center for *situational awareness*, and *control* traffic for *event driven communication* (*e.g.*, for critical control applications) in which generic object oriented substation event (GOOSE) packets are used to regulate the SAS.

[1] There are exceptions: *e.g.*, for medical cyber-physical systems the primary security concern is, typically, confidentiality.

[2] The general communication model also involves a *class combining operator* "$\oplus$" that specifies, for any pair of operand classes, the class in which the result of any binary function on values from the operand classes belongs. For our SAS application classes cannot be combined.

[3] In general, there should be at least $k = 3$ priorities to allow for a more flexible control of traffic flows–see Section 4.3.

4.1 The security priority of a packet

The header frame of a packet has several components. These include the security priority of the packet, a packet identifier, and a message authentication code (MAC). The packet identifier is typically a monotonic increasing counter value (that will identify dropped packets) and the MAC authenticates the source, the target and confirms the integrity of the payload. Additionally, the header may include an error detection code (*e.g.,* a *cyclic redundancy code*) or the payload may be encoded using an error correcting code (e.g., a *Reed-Solomon code* [8]). Hi priority packets are restricted to this structure. For lower priority packets the payload is encrypted. When these packets are received, the payload must first be decrypted, and then decoded (if an error correction code was used), and finally its integrity verified using the message authentication code.

4.2 The congestion level of a link

The congestion level of a link is determined by its traffic flow. This is established dynamically by a *real-time intelligent agent* that monitors traffic flows. When the traffic flow exceeds a certain threshold, the congestion level of the link is raised. The effect of this is to reduce the traffic flow rate to a level for which availability is guaranteed (*e.g.,* when only hi priority packets are allowed to be transmitted.

Remark 2. Given a payload, a network node can construct a packet with hi priority (*e.g.,* by not encrypting the payload) or lo priority, so that its priority will always dominate the congestion level of the channel, which guarantees that it will be transmitted. That is, nodes control both the generation and the transmission of packets. This is not a violation of the *separation of duties* security requirement because network nodes are TPM-compliant and different roots of trust are used for each duty (the root of trust for measurement (RTM) and the root of trust for reporting (RTR)–Section 3.1). As a consequence, when the congestion level is elevated no lo priority packet is transmitted. This guarantees real-time communication availability.

4.3 SV packets and GOOSE packets

Sample vector (SV) packets and generic object oriented substation event (GOOSE) packets have essentially the same structure. From a security point of view however, availability for GOOSE packets can be critical (their payload contains control information), while the redundancy in SV packets makes them less critical. Therefore, it may be desirable to have two high level classes: $\mathsf{hi1} = \ell_{k-1} \succeq \mathsf{hi2} = \ell_k$, with hi1 requiring error detecting codes for GOOSE packets and hi2 requiring error correction codes.

4.4 Relationship to existing mechanisms for supporting availability

Traditionally, network availability can be achieved with Quality-of-Service (QoS) support. For example, Internet QoS mechanisms including Integrated Services (Intserv) [4] and Differentiated Services (DiffServ) [3] can both guarantee network availability. IntServ guarantees end-to-end services by reserving bandwidth along all links in the

path from source to destination while DiffServ allows different bandwidth for different types of traffics on each link. Our access control structure for need-to-get policies is somewhat similar to DiffServ. The main difference is that DiffServ differentiates the services to different types of traffics at any time while our scheme only differentiates services when the network is under the congestion condition: when the link is not congested, our scheme treats all packets the same.

5 Resilient Cyber-Physical Systems

In this section we show how to achieve resiliency in real-time for substation automation systems (SAS) of a power utility that conforms with IEC/TR 61850-90-5 (Section 2), by using TPM-engines (Section 3.1) and an access control infrastructure that enforces *need to get policies* (Section 4).

First, note that, as pointed out earlier in Remark 1, we must assume that the network has sufficient redundancy so that, *we get real-time resilience when only hi priority (authorized) packets are transmitted*. This is necessary because otherwise it would be impossible to achieve real-time resilience even when no packets get corrupted or dropped. This can be achieved if we assume that there is a bound on the flow of such packets and that the SAS network system is designed with sufficient redundancy to cope with this traffic.

Next, observe that since the power utility conforms to IEC/TR 6150-90-5, the communication network of the SAS consists of traffics involving either reporting SV packets or control GOOSE packets. We also assume that the SAS is TPM-enabled (Section 3.1). This means that all actions by network nodes are managed by *trust engines* that attest to the integrity of the nodes, that protect stored data, and that prevent nodes from behaving in an unauthorized way (compromised nodes will be prevented by their trust engine). Furthermore, the adversary cannot use the SAS nodes as proxies for DoS or DDoS attacks, because their trust engines (the roots of trust for reporting) will not provide the required keys for authenticating unauthorized packets (remote attestation will prevent faulty PMU generating SV packets).

Finally by using an access control infrastructure that enforces *need to get policies* to manage the SAS network traffic (Section 4), we make certain that whenever the traffic flow is high and packets may get dropped, then the congestion level of the network links gets elevated, which implies that only higher priority traffic can use the network—this guarantees real-time resilience (by our first assumption).

6 Emulations and Empirical study

In order to study the feasibility of the proposed scheme, we have set up a testbed consisting of networked machines capable of operating both Windows 7 and Ubuntu Linux; these machines are used as our intelligent electronic devices connected via a Cisco Catalyst 3560G series PoE 48 switch. For all the experiments reported here, all the machines are configured to run Ubuntu Linux. To carry out the experiments, we started with an open source IEC/TR 61850-90-5 implementation released by SISCO [10]; we

have made substantial modifications to the implementation, including porting the implementation from a Windows-only one to a POSIX compatible implementation, and extending support for authentication and cryptographic protocols.

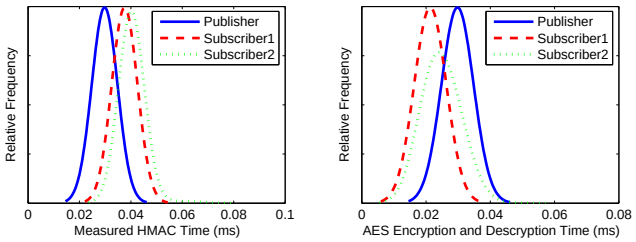
Our modified version of SISCOs package implements the IEC/TR 61850-90-5 APIs, as well as additional security components. There are five major phases of communication in the implementation: (1) Software Initialization which must be performed before any communication can be achieved across the network; by parsing a configuration file, it establishes the nodes role in the simulation as a publisher, subscriber, or both, and prepares the necessary components to receive and/or transmit information. (2) Key Distribution Center registration, wherein credentials necessary to communicate with the KDC and with nodes on the network are created and distributed. (3) Transmit, wherein packets are created, then encoded, a process which includes encryption (if desired) and authentication, finally the packets are sent. (4) Receive, wherein packets are received, decoded (which refers to verification of the authenticity and decryption), and processed. (5) Finally there is a Termination sequence which must be carried out prior to shutting down the system; this process ensures proper unsubscription from multicast sessions and deallocation of resources dedicated to the service during operation.

To verify the feasibility our suggested techniques, it is necessary to monitor the networks ability to transmit the SV and GOOSE packets effectively, and to understand the impact of various cryptographic functions on the ability of the network to meet the requirements of our need to get policy. To accomplish this we added a meta-framework to the implementation which saves timing information at key stages during the operation of the system. Specifically, on the publisher machine a time-mark is saved: (a) at the time when the packet is initially built, (b) directly before it is encrypted, (c) directly before it is HMACed for authentication, and (d) directly before it is sent. On the subscriber machine, the time is recorded: (e) when the packet is first received, (f) directly before it is authenticated, (g) directly before it is decrypted, and (h) finally again when the packet is fully processed by our system. Using these timestamps we are able to observe the entire transmission time for each packet, as well as the time required for sub-elements within the transmission and receiving sequences; these observations allow us to evaluate the ability of our framework to fulfill the timing constraints of IEC/TR 61850-90-5, while providing the additional security components necessary to achieve trusted communication.

In the experiments, the session keys are generated using the TPM module on the sender and then the keys are shared with all receivers in the multicast group. While this introduces weak security links, it is the choice as multi-cast is being used in the IEC/TR 61850-90-5 implementation. From a security point of view, multiple unicasts should be used instead if stronger security is desired.

6.1 Experiment Results

To simulate the heterogeneity of deployed systems, we report the results using three machines, one publisher and two subscribers. The processor of the publisher is an Intel Xeon E5405 2.00GHz x8 with 4 GB of memory; the first subscriber has an Intel Xeon 5120 1.86GHz x2 with 2 GB of memory and the second receiver has an Intel Xeon E5506 2.13GHz x4 with 6 GB of memory.



(a) (b)

Fig. 3. (a) HMAC computation time, and (b) AES encryption/decryption time on the 3 machines.

Figure 3(a) shows the HMAC computation time on the three machines. On the publisher, the average computation time for HMAC (SHA1) is 0.0298 millisecond with a standard deviation of 0.00028 millisecond. On the first subscriber, the average computation time for HMAC is 0.038 millisecond with a standard deviation of 0.00033 millisecond; on the second subscriber, the average computation time for HMAC is 0.041 millisecond with a standard deviation of 0.0026 millisecond. While the computation time for HMAC is not constant, as it is affected by other system activities, the time is fairly consistent. Note that for substation automation, where the required processing time is within 4 milliseconds, depending on the machine, the HMAC computation time does not have a substantial impact. Among the three machines, the worst is about 1% of the required time. However the HMAC computation time varies significantly among the three machines with the largest difference being about 37%. The results show that different types of machines can have different processing times. This is a practical issue that must be addressed in the real-time communication in a large interconnected system where machines are heterogeneous.

We have measured the time required for privacy (confidentiality) using AES. Figure 3(b) shows the encryption on the publisher and decryption on the two subscribers. The subscribers are more efficient in AES computation than the publisher. The mean time for AES on the subscribers is 0.0213 millisecond and 0.0248 millisecond respectively, while it is 0.661 millisecond on the publisher. Clearly newer processors are optimized for AES-like computation. This highlights the importance of developing systems that work on heterogeneous platforms as trusted infrastructures are likely to be deployed incrementally. For PMU measurements and GOOSE packets, since the packet size is fairly small, AES encryption and decryption is fairly efficient. While these estimations need to be further verified, they are fairly consistent for over 500 packets.

We have measured the transmission time and the total required time to process each packet. The average transmission time is 0.0494 millisecond for receiver 1 and 0.0476 millisecond for receiver 2; clearly the transmission time here is smaller compared to typical substations. The average total time for each packet is 0.6837 millisecond and 0.8057 millisecond. These results indicate that the active time of a packet in an IEC/TR 61850-90-5 enabled system is less than 1 millisecond. To meet the IEC/TR 61850-90-5 time constraint of 4 millisecond end-to-end time, more than 3 milliseconds can be allocated to packet transmission. Consequently, by using our access control structure that guarantees delivery for high priority packets (since the queuing delay at each router/switch is bounded) we achieve availability. Integrity of packets is provided by

the HMACs and confidentiality by AES. It follows that our approach/techniques will secure substation automation for power utility systems.

7 Conclusion

In this paper we propose techniques for resilient cyber-physical systems that are directly applicable to secure substation automation for power utility systems. A distinctive feature of our approach includes (1) incorporating TPM-enabled trusted computing technologies, and (2) enforcing *need-to-get* policies that support the time-sensitive nature of critical infrastructures, especially the electricity grid. The feasibility of our approach is demonstrated through analyzing and experimenting with an open source IEC/TR 61850-90-5 implementation.

References

1. BELL, DAVID ELLIOTT AND LAPADULA, LEONARD J. Secure Computer Systems: Mathematical Foundations. *MITRE Corporation, http://www.albany.edu/acc/courses/ia/classics/belllapadula1.pdf* (1973).
2. BIBA, K. J. Integrity Considerations for Secure Computer Systems. *MITRE Corporation, Technical Report, ESD-TR-76-372, MTR-3135* (April 1977).
3. BLAKE, S., CLARK, D., CARLSON, M., DAVIES, E., WANG, Z., AND WEISS, W. An Architecture for Differentiated Services. *RFC 2475* (December 1998).
4. BRADEN, R., CLARK, D., AND SHENKER, S. Integrated Services in the Internet Architecture: an Overview. *RFC 1633* (June 1994).
5. DENNING, D. E. A lattice model of secure information flow. *Commun. ACM 19*, 5 (May 1976), 236–243.
6. INTERNATIONAL ELECTROTECHNICAL COMMISSION. IEC/TR 61850-90-5, Edition 1.0 2012-05, Technical Report, Power systems management and associated information exchange Data and communications security. *http://webstore.iec.ch/preview/info_iec61850-90-5%7Bed1.0%7Den.pdf* (May 2012).
7. INTERNATIONAL ELECTROTECHNICAL COMMISSION. IEC/TS 62351-1, First edition 2007-05, Technical Specifications. *http://webstore.iec.ch/preview/info_iec61850-90-5%7Bed1.0%7Den.pdf* (May 2012).
8. REED, I. S. AND SOLOMON, G. Polynomial Codes Over Certain Finite Fields. *SIAM Journal of Applied Math. 8* (1960), 300–304.
9. SANDHU, R., COYNE, E., FEINSTEIN, H., AND YOUMAN, C. Role-based access control models. *IEEE Computer (IEEE Press) 29*, 2 (1996).
10. SISCO. Cisco and SISCO Collaborate on Open Source Synchrophasor Framework, Press Release. *http://www.sisconet.com/downloads/90-5_Cisco_SISCO.pdf* (2011).
11. TRUSTED COMPUTING GROUP (TCG). http://www.trustedcomputinggroup.org/.
12. TRUSTED NETWORK CONNECT ARCHITECTURE FOR INTEROPERABILITY (TNC), SPECIFICATION 1.3. Revision 6, April 2008.
13. TRUSTED PLATFORM MODULE (TPM) STRUCTURES, LEVEL 2, VERSION 1.2. Revision 116, Communication Networks and Systems for Power Utility Automation. *http://www.trustedcomputinggroup.org/resources/tpm_main_specification* (March 2011).