Towards a Secure Electricity Grid

Mike Burmester, Joshua Lawrence, David Guidry, Sean Easton, Sereyvathana Ty[*]
Xiuwen Liu, Xin Yuan, and Jonathan Jenkins

Department of Computer Science, Florida State University, Tallahassee, FL 32306, U.S.A.
{burmester, lawrence, dguidry, easton, liux, xyuan, jenkins}@cs.fsu.edu

[*]*Sandia National Laboratories, New Mexico, P.O. Box 5800, Albuquerque, NM 87185, U.S.A.*
sty@sandia.gov

***Abstract*—The transmission of bulk power within a zone of an interconnected region of an electric grid is controlled by substation automation systems. The substations are where electricity is routed throughout the grid, as well as the control and communication nodes of the network grid. It is crucial for the security of the electric grid that there should be no break in the network communication. Currently, IEC 61850 specifies the communication interface and gives utility companies interoperability for Intelligent Electronic Devices of substation automation systems and is intended to support Distributed Wide Area Monitoring, Control and Protection. This requires ultra real-time data feeds that must be trusted. Currently there is no agreed upon security standard that accompanies IEC 61850. In this paper we propose a framework architecture that extends IEC 61850 to capture trusted substation automation by combining (i) Trusted Computing engines, (ii) a Kerberos multicast authentication service, and (iii) a real-time attribute-based access control system. We then integrate this framework into an open source IEC 61850 profiler (a real-time emulator) for substation automation recently released by SISCO, and show that the integrated profiler is IEC 61850 compliant, while supporting integrity, confidentiality and real-time availability (with end-to-end time for critical data feeds less than $4\,ms$), against strong adversaries (including insiders).**

I. Introduction

IEC 61850 [1] is a technical report of the International Electrotechnical Commission for standardizing substation automation systems (SAS) of electric grids. The report provides for reliable and efficient communication between heterogeneous intelligent electronic devices (IED) for real-time substation automation. This is crucial, since for critical infrastructures there is no room for communication breakdown between components supplied by different vendors.

IEC 61850-90-5 [2] extends IEC 61850 by specifying the use of the IP transport protocol (both IP Unicast and Multicast) for exchanging data between Phasor Measurement Unit(s) (PMU) and Phasor Data Concentrator(s) (PDC). Generic Object Oriented Substation Event (GOOSE) messages and Sample Value (SV) messages are encapsulated in IP packets and formulated so that they can be distributed in Wide Area Network (WAN) environments. This allows for low cost Wide Area Monitoring, Protection and Control (WAMPAC).

Figure 1 illustrates a substation broken into IEC 61850 components. Communication between IEDs is through the switch, and communication outside the substation is through the router, while for high voltage (HV) equipment communication is through a merging unit for associated IEDs. SAS communication also occurs within a zone, controlled by an electricity generation company such as the Tennessee Valley Authority, or between multiple zones (zone-to-zone) within a region, such as the Eastern Interconnect in the United States. Trusted and reliable communication is vital to running and managing a healthy electric grid, particularly when power anomalies have to be reported in real-time.

Security, which is critical for the safety of networked substations, has not been fully addressed in IEC 61850. The standard IEC 62351 [3] addresses the security of some of the IEC 61850 protocols. However, at present, it does not address security issues related to WAMPAC, where ultra real-time data feeds are used for protection and control.

For trusted automation, security issues must be addressed in real-time. For this, parts of the IEC 61850 will need to support *real-time assured* and *trusted* communication. With respect to security, IEC 62351 will need to be significantly augmented to provide *real-time availability*, *authentication*, *integrity*, and when needed *scalable-confidentiality*. IEC 62351 must also provide trust in the integrity of the sender, receiver and the data sent before, during and after high availability events, *e.g.* faults, rolling blackout, and voltage collapse. Before an effective and open standard can be established so that vendors can provide WAMPAC, these issues must be resolved.

In this paper we present a framework for assured and trusted real-time communication by employing:

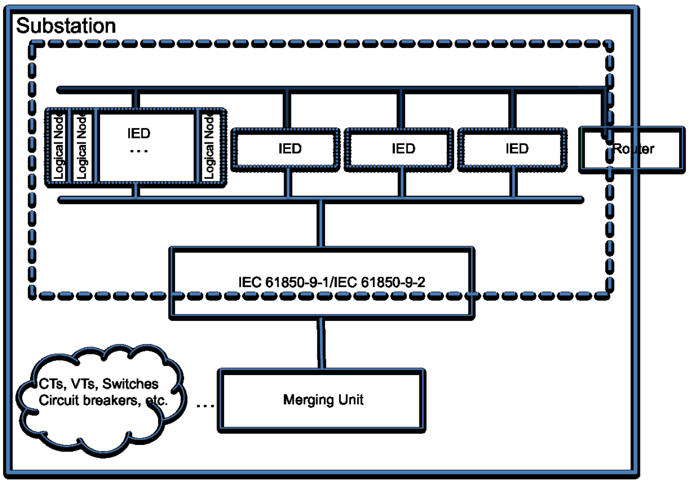

Fig. 1. IEC 61850 view of a substation

1) A Trusted Platform Module interface for built-in non-migratable trust;
2) A Kerberos multicast authentication service;
3) A real-time attribute-based access control system.

The rest of this a paper is organized as follows. In Section II we summarize the Trusted Platform Module (TPM) architecture, and show how it is used to assure trust in the behavior of cyber-physical devices, and in particular the components of an electric grid. We then consider the Trusted Network Connect platform and show how it is used for trusted SAS interoperability. SAS communication is typically multicast. In Section III we survey the Kerberos authentication service and extend it for multicast applications, and in Section IV we consider an attribute-based access control service that supports real-time availability. In our framework this service is used to guarantee real-time data feeds. In Section V we show that our framework will secure IEC 61850-90-5 communication, and in Section VI we integrate our framework with an open source IEC 61850-90-5 profiler for trusted IEC 61850-90-5 profiling. We use the extended profiler in Section VII to demonstrate the effectiveness and sufficiency of our framework. We conclude in Section VIII with a summary and discussion of security implications and future work.

II. The Trusted Platform Module Interface

The Trust Platform Module (TPM) is an interface for binding data to platform configurations of hardware systems to enhance software security. The Trusted Computing Group (TCG) [4] defines the architecture for TPM in terms of trusted engines, called *roots of trust*, used to establish trust in the expected behavior of the system.

A TPM [5] interface can be implemented as a hardware, software, or embedded software cryptographic device that has two basic capabilities: *remote attestation* and *sealed storage*. The TPM device 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.

There are three mandatory 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 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).

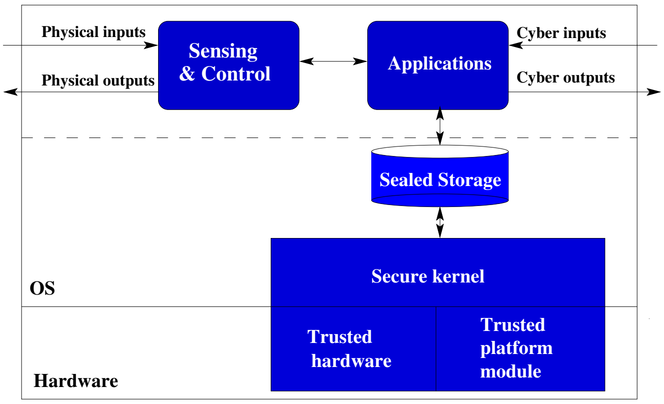

Fig. 2. A TC-compliant cyber-physical-device.

The components of a TPM support RTR (for reporting) and RTS (for storage) engines. They include: an I/O, a PCR, a Program Code, an Opt-In and an Exec Engine. The I/O manages data flows over communication buses (internal and external) by encrypting/decrypting data. Non-volatile storage is used for persistent keys (such as the endorsement key). The PCR can be implemented in volatile or non-volatile memory. The Program Code contains firmware for measuring platform devices. The Opt-In is used to customize TPM modules. The Exec Engine runs program codes: it performs TPM initialization and takes measurements.

The TCG requires that TPM modules be 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. Finally we note that there exist implementations in which the TPM can be run on virtual machines (vTPM) and implementations for which the TPM is bound to a portable device (mTPM). For more details on the architecture of TPM, the reader is referred to [5].

A. 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. Figure 2 illustrates the architecture of a cyber-physical device in which a TPM microprocessor is used to protect trusted applications. Three components are distinguished.

- The secure kernel which contains trusted hardware and a TPM. The TPM 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.
- A sealed storage unit, used for storing sensitive data and keys: access to it requires the release of keys from the TPM (via the RTS engine).
- The Sensing and Control unit and the Applications.

B. The Trusted Network Connect

The Trusted Network Connect (TNC) [6] is a TC platform interoperability architecture for trusted access control that is based on TPM architectures. What distinguishes TNC from other interoperability architectures is the requirement that the OS configuration 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. A distributed Public Key Infrastructure is used 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 granted access to the server. An access control service (typically RBAC [7]) and credentials are used.
3) The identities of the client and server are authenticated. A root of trust engine is invoked on the TPMs of both parties to release the required keys for a handshake protocol [6]. The TPM will only release keys if the current configuration state of the OS of the parties allows it.
4) The handshake protocol is properly executed.

The TPM interface will enforce the integrity of communicated data (by releasing appropriate message authentication keys), and depend on the application to provide confidentiality of communicated data.

C. Trusted Substation Automation

A substation automation system (SAS) consists of multiple substations connected via an intranet network. To establish a trusted SAS, each component in this zone should be TC-compliant. For trusted interoperability, the TNC platform is used. Trust in the SAS is established by invoking the roots of trust, and by enforcing trusted communication via the TNC platform.

III. A Kerberos multicast authentication service

Kerberos is a single-sign-on authentication service for client-server applications. It is not designed for multicast applications where a client wants authenticated multicast access to several principals, a common requirement with cyber-physical systems. Furthermore it does not address fully Denial of Service (DoS) threats or insider attacks, that can incapacitate power grids and more generally critical infrastructures.

In this section we show how to extend Kerberos to capture both requirements: efficient authenticated group multicast by using the approach in [8], as well as security against DoS threats and insider attacks by using a TPM interface (Section II). We first describe the Kerberos multicast extension and then how to integrate Kerberos with a TPM interface.

The Kerberos protocol consists of several sub-protocols. We refer the reader to RFC 4130 [9] for details. Suppose that a client principal with a valid (non expired) Ticket Granting Ticket TGT (obtained from an Authentication Server AS) requests from the Ticket Granting Service (TGS) authenticated multicast access to a group GRP of application principals. We distinguish two cases:

1) GRP along with its long term secret group key is listed in the Kerberos database.
2) GRP is a new group, not listed in the Kerberos database.

In the first case, the client can get a session group key to access GRP as in the original Kerberos protocol, by sending an application request GRP_AP_REQ to the TGS with the group name GRP, a valid TGT, and an authenticator. In the second case the TGS must establish a long term session group key and distribute this key to all the principals of GRP. This requires the TGS to establish an authentication channel with each one of the principals in GRP, and then use this channel to transport a long term group key. Here we use two Kerberos client/application message authentication types: KRB_AP_REQ and KRB_AP_REP. The TGS selects a random group key and sends a request KRB_AP_REQ to each principal of the group GRP that contains: the GRP name and a list of its principals, and the long term group key (encrypted with the secret key of the service principal). Each application principal in GRP then sends a reply KRB_AP_REP that contains an authenticator (encrypted with the secret key it shares with the TGS). For more details see [8].

The Kerberos multicast group authentication protocol is secured by using a TPM interface. Suppose that all principals of the Kerberos system, as well as the trusted KDC Servers are TPM-compliant. Then the private keys of each principal are in shielded locations and only become available when needed by Kerberos, provided the principal's OS has not been compromised (a sampling of the configuration of its state prior to secret keys being released for encryption must match the PCR value). Furthermore, prior to establishing an authentication link, remote attestation will ensure that there are trust paths that link the KDC to the client and the service principals (via shared secret keys). This will protect the system from external integrity and confidentiality threats, as well as insider threats that target the private long term keys of principals.

IV. Real-Time Attribute-Based Access

Access control systems are trust infrastructures that manage the resources of computer or network systems. Early systems include the Bell-LaPadula model [10] that enforces *need-to-know* (confidentiality) policies and the Biba model [11] that enforces integrity policies. For these systems the subjects (users) and objects (resources) are assigned security labels selected from the same partially ordered set, with read (respectively, write) access granted only if the label of the subject (respectively, object) dominates the label of the object (respectively, subject). These models are not dynamic and attempts to make them dynamic are not scalable.

With Role Based Access Control (RBAC) [12], [7], access permissions are assigned to roles and users are assigned to one or more roles. A subject can exercise an access permission only if the permission is authorized for the activated role in a given session. While RBAC scales better than previous models, it is not suitable for highly dynamic systems. Extensions such as the Temporal RBAC [13] and the Generalized Temporal RBAC [14] address some of these issues by allowing periodic enabling of roles, trigger-enabled temporal dependencies among roles, and role-permission assignments. However when the number of subject and object attributes becomes large, RBAC roles and permissions grow exponentially [15].

The Attribute-Based Access Control (ABAC) model [15], [16] assigns attributes to subjects and objects. Authorization is defined for subject descriptors consisting of several attribute conditions. Permissions consist of object descriptors that contain sets of attributes and attribute conditions and an operation on the objects. Environment attributes can also be used to control access. In the basic ABAC model [15] access to an object o by a subject s in a particular environment e is resolved by evaluating policy rules, where each rule can be seen as a boolean function f evaluated over the attribute values of s, o, e contained in their descriptors: $access(s, o, e) \leftarrow f(attr(s); attr(o); attr(e))$.

ABAC encompasses the functionality of RBAC models by treating security labels and identities or roles as attributes. Compared to RBAC, ABAC captures environment and dynamic attributes such as temporal issues. However in highly dynamic real-time applications the event space that determines the attribute values can get very large thus making any attempt to adapt ABAC models to capture real-time availability scenarios nonscalable.

The Internet Engineering Task Force (IETF) proposed two Quality-of-Service (QoS) architectures that support real-time availability for Internet communication: DiffServ (Differentiated Services) [17] and IntServ (Integrated Services) [18]. DiffServ addresses per-hop forwarding behavior (PHB) of IP packets while IntServ addresses per-flow behavior of end-to-end streamed data. These architectures distinguish between core routers that queue and schedule packets, and edge routers that monitor and police traffic flows.

For real-time availability, core and edge routers must be trusted to adhere to committed reservations, and core routers must be able to verify that packet flows are authentic. IETF [19] proposes the use of the authentication mechanisms defined in [20] and [21] for hop-by-hop integrity and node authentication. These will protect against corruption and spoofing of RSVP packets. However they will not protect against Distributed DoS attacks [19]: a determined adversary can send a large amount of traffic that can lead to an amplification attack.

A. Adding real-time availability to ABAC

We briefly overview the real-time attribute-based access (T-ABAC) control system proposed in [22]. This uses several features of DiffServ and IntServ, but focuses on guaranteed real-time availability in the presence of a malicious adversary, rather than QoS. In particular when an edge router is congested, availability is restricted to those packets with the highest priority, with all other packets dropped.

For any subject s, availability of a resource o at time t_i is determined based on the real-time attributes $attr(x; t_i)$, $x \in \{s, o\}$, with values in a linearly ordered set of availability labels. We call the label of $attr(x, t_i)$, the *priority* label when x is a subject, and the congestion level when x is an object. The availability labels are dynamically determined based on user events, temporal events, the context of the requested service and system events.

For network systems, availability refers to a packet forwarding service. To enforce real-time availability for IP networks, we combine the mechanisms supported by DiffServ and IntServ. However in this case the policy is to *guarantee* that *high* (critical) priority packets are forwarded in real-time when there are faults (caused by Nature or the adversary). For our application we require that [22]:

1) If a packet P with *high* priority is received by an edge router R then: (i) R will drop all packets in its queue and forward P, and (ii) for the next k time intervals R will drop all incoming packets whose priority is not *high*, where $k \geq 1$ is a security parameter.
2) If a packet P is received by an edge router R and its priority is not *high* then: if the priority of P dominates the congestion level of R it is forwarded by R; else it is put in the queue of R (with priority queueing).
3) If the priority of a packet P in the queue Q of R dominates the priorities of all packets in Q, and the congestion level of R then P is forwarded by R (round robin is used if necessary).

This policy favors guaranteed delivery of *high* priority packets, with QoS being a secondary goal. The parameter k is selected to prevent IP packets whose priority is not high from blocking other high priority packets reaching their destination. Essentially, the policy is that when *high* priority packets are being sent, then the communication network will only service such packets when there is network congestion.

V. Real-Time Trusted Substation Automation

From our discussion in Section II-C it follows that the components of a TC-compliant SAS are trusted with trusted interoperability. From our discussion in Section III it follows that we have multicast authentication with message integrity and confidentiality.

This addresses not only passive attacks but also active attacks including insider attacks. The TPM interface will prevent authorized components that get compromised from behaving maliciously, in particular, from contributing to a Destributed DoS (DDoS). Furthermore, the authentication service will prevent external DDoS attacks.

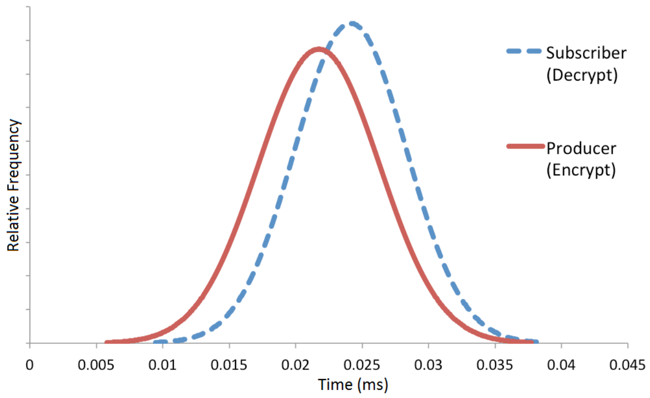

Fig. 3. AES Encryption & Decryption time on the Producer & Subscriber.

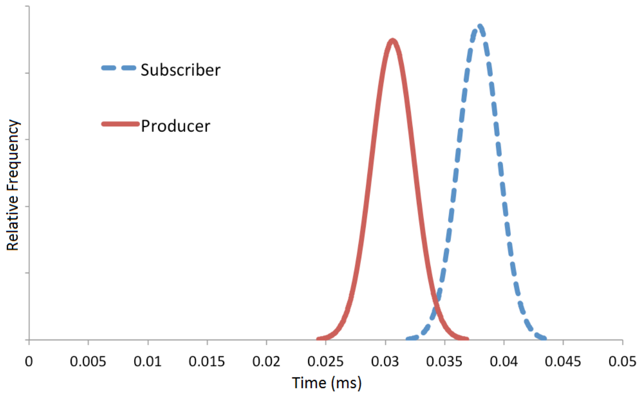

Fig. 4. HMAC Generation & Verification time on the Producer & Subscriber.

If we use real-time access control to manage data feeds as in Section IV-A, then real-time availability is guaranteed for *high* priority packets provided the SAS network has sufficient redundancy to be resilient when only such packets are sent. For our application all packets that contain critical state/control information are assigned a high priority label.

What remains to be shown is that the end-to-end delivery time of IP packets is bounded by the IEC 61850-90-5 constraints that specify a $4\,ms$ threshold for packet processing and delivery. Our performance study indicates that the proposed techniques can be supported within this time constraint.

VI. A Profiler for Trusted IEC 61850-90-5

IEC 61850 offers advanced object oriented semantics for information exchange in power system automation applications, SCADA, system protection, substation automation, distribution automation, etc. IEC 61850-90-5 extends this to provide a reliable communication infrastructure for distributing synchrophasor information in real-time over wide area networks using IP multicast and IP subscription.

SISCO recently released an open source IEC 61850-90-5 profiler [23] that provides routines for five major software sequences: initialization, performed prior to any additional IEC 61580-90-5 function being used; KDC registration, where credentials and communication addressing information required to communicate with KDCs are stored; transmission, performed in order to transmit packets; reception, performed in order to receive packets; and termination, performed during application termination.

We extend the SISCO IEC 61850-90-5 profiler with security features by integrating it with: (i) trusted engines, (ii) Kerberos multicast authentication, and (iii) real-time access control. The enhanced IEC 61850-90-5 profiler is then used to evaluate the efficiency of our security framework architecture.

The Kerberos API [9] is used to handle mutual authentication of principles and establish secure communication channels between clients and the TGS. Then the multicast Kerberos protocol in Section III takes over, with the TGS generating and distributing long term group keys and session group keys. Our profiler interfaces with Kerberos through the API rather than directly by modifying the core Kerberos code. Since the primary role of Kerberos in this protocol is to establish mutual authentication, which the standard API provides, there is no benefit to change the internal workings of Kerberos. We use an open source TPM driver package [24] to access TPM functionalities. The entire implementation of the security extension of the IEC 61850-90-5 is available at http://www.sait.fsu.edu/software.html.

VII. Experimental Results

To demonstrate and quantify performance in a fully functional system, we established a testbed consisting of seventeen machines connected using a Cisco Catalyst 3560G series PoE 48 switch. All machines were TPM enabled running Ubuntu Linux 12.04 64 bit OS; eight had an Intel Xeon 5120 1.86 GHz x2 CPU with 2GB memory, and nine had an Intel Xeon E5506 2.13GHz x4 CPU with 6 GB memory. A dedicated machine running the krb5 API acted as our Kerberos 5 release 1.10 server, which interfaced directly with the Kerberos API.

Using this testbed, we conducted experiments to measure the runtime for the following essential building blocks.

Encryption/Decryption. Both the producer and subscribers performed an AES encryption or decryption on the payload of packets. We measured how long that process took for each. As shown in Figure 3, over a course of 500 packet transmissions the average time required for encryption was $0.02175\,ms$ with standard deviation $0.00457\,ms$. Decryption took on average $0.02413\,ms$ with standard deviation $0.00420\,ms$.

HMAC Authentication. Both the producer and subscribers computed an HMAC using SHA2. As shown in Figure 4, over a course of 500 packet transmissions the average time required by the producer was $0.03063\,ms$ with standard deviation $0.00178\,ms$. The average time required by the subscribers was $0.03786\,ms$ with standard deviation $0.00167\,ms$.

RNG. The average time to generate a random number was $0.0108715\,ms$ with standard deviation $0.000399\,ms$.

TPM lock/unlock. The average time to lock a TPM key was 0.48788*ms* with standard deviation 0.021401*ms*; to unlock a key it was 0.36454*ms* with standard deviation 0.051601*ms*.

A producer must build the packet, encrypt the payload (for confidentiality), calculate an HMAC (for authentication) and finally transmit the packet. Each subscriber that receives the packet, must verify the HMAC and decrypt the payload. The time required for this is: $enc + dec + 2\,mac\ \ ms,$ since the producer must encrypt the payload, authenticate it, and the subscriber must verify it and then decrypt it. Using our earlier average runtime estimates we see that the processing time for a GOOSE packet is $0.11437\,ms$. This does not include the time for network transmission, the IEC 61850-90-5 header generation, the IP protocol formatting, and other general processing requirements. We estimated this to be roughly an additional $0.00069\,ms$ for the producer and $0.54\,ms$ for the subscriber. The relatively high value for the subscriber is due to queuing delays (which also impact on the standard deviation which is $0.302\,ms$).

The total processing time is no more than $1\,ms$, which is well within the $4\,ms$ time limit for GOOSE packets according to the IEC 61850-90-5 specification. This shows that our security architecture provides real-time availability and integrity for the communication of IEC 61850-90-5 compliant SASs.

VIII. Summary and Discussion

In this paper, we proposed a framework to secure an electric grid by integrating real-time attribute-based access, trusted computing engines (e.g., TPM and TNC), and group key management via Kerberos. We demonstrated the sufficiency of our framework by establishing a fully functional profiler for trusted IEC 61850-90-5. Our experimental results show that the profiler provides strong security and at the same time complies with the real-time constraints of IEC 61850-90-5.

While our current experiments involve a relatively small number of nodes (due to practical limitations), the results can be readily generalized to much larger systems. For example, there are an estimated 5,000 substations in the Eastern Interconnection with over 40 intranets. Assuming that each intranet is organized as a network without any hierarchy, due to the scalable nature of multicast, the performance of sending the same packet to 125 nodes should be close to sending it to 16 nodes, as multicasting packets can be done efficiently by routers, without additional computation on the sending machines. When intranets are organized into a hierarchy of the Internet for the Eastern Interconnection, additional delays will be limited to a few hops on the network.

Consequently, our results provide meaningful measurements for the entire grid, not just a small network setting. While our experiments use an IEC 61850-90-5 implementation, the proposed framework can be generalized to other electricity delivery components and more generally cyber-physical systems where there is a critical real-time requirement, including most critical infrastructures.

Acknowledgement. The paper is based upon work partially supported by the National Science Foundation Grant No. 1027217.

References

[1] International Electrotechnical Commission, “IEC/TR 61850, 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.

[2] ——, “IEC/TR 61850-90-5, Edition 1.0 2012-05, Technical Report, Power systems management and associated information exchange – Data and communications security,” *http://preview/webstore.iec.ch/info_iec61 850-90-5% 7Bed1.0%7Den.pdf*, May 2012.

[3] ——, “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.

[4] Trusted Computing Group (TCG), “http://www.trustedcomputinggroup.org.”

[5] TCG, “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.

[6] Trusted Network Connect Architecture for Interoperability (TNC), Specification 1.3, “Revision 6,” April 2008.

[7] R. S. Sandhu, E. J. Coyne, and H. L. Feinstein, and C. E. Youman, “Role-Based Access Control Models,” *IEEE Computer*, vol. 29, no. 2, pp. 38–47, 1996.

[8] J. J. Jenkins, S. Easton, D. Guidry, M. Burmester, X. Liu, X. Yuan, J. Lawrence, and S. T. Ty, “Trusted Group Key Management for Real-Time Cyber-Physical Systems.” In, 7th International Workshop on Critical Information Infrastructures Security (CRITIS 2012), Sept. 2012

[9] D. Moberg and R. Drummond, “RFC4130, Mime-based secure peer-to-peer business data interchange using http, applicability statement 2”, 2005.

[10] E. D. Bell and J. L. La Padula, “Secure computer system: Unified exposition and Multics interpretation,” Bedford, MA, 1976. [Online]. Available: http://csrc.nist.gov/publications/history/bell76.pdf

[11] K. J. Biba, Integrity considerations for secure computer systems,” MITRE Corp., Tech. Rep., 1977.

[12] D. F. Ferraiolo, S. I. Sandhu, R. S.and Gavrila, D. R. Kuhn, and R. Chandramouli, “Proposed NIST standard for role-based access control,” *ACM Transactions on Information Systems Security*, vol. 4, no. 3, pp. 224–274, 2001.

[13] E. Bertino, P. A. Bonatti, and E. Ferrari, “TRBAC: A temporal role-based access control model,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 4, no. 3, pp. 191–233, 2001.

[14] J. Joshi, E. Bertino, U. Latif, and A. Ghafoor, “A generalized Temporal Role-Based Access Control model,” *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, no. 1, pp. 4–23, 2005.

[15] E. Yuan and J. Tong, “Attributed based access control (ABAC) for Web services,” in *Proceedings of the 2005 IEEE International Conference on Web Services (ICWS 2005)*. IEEE, 2005, pp. 561–569.

[16] T. Priebe, W. Dobmeier, C. Schläger, and N. Kamprath, “Supporting attribute-based access control in authorization and authentication infrastructures with ontologies,” *Journal of Software*, vol. 2, no. 1, pp. 27–38, 2007.

[17] S. Blake, D. Black, D. Carlson, M., Z. E., Wang, and W. Weiss, “An Architecture for Differentiated Services,” *RFC 2475*, Dec. 1998.

[18] R. Shenker, S.and Braden and D. Clark, “Integrated services in the Internet architecture: an overview,” *IETF Request for Comments (RFC)*, vol. 1633, 1994.

[19] S. Dhesikan and J. Polk, “Integrated Services (IntServ) Extension to Allow Signaling of Multiple Traffic Specifications and Multiple Flow Specifications in RSVPv1,” *IETF Internet-Draft*, 2012.

[20] F. Baker, B. Lindell, and M. Talwar, “RSVP cryptographic authentication,” *Request for Comments 3097*, 2001.

[21] R. Braden and L. Zhang, “RSVP Cryptographic Authentication–Updated Message Type Value,” RFC 3097, April, Tech. Rep., 2001.

[22] M. Burmester, M. Emmanouil, and V. Chrissikopoulos, “T-ABAC: An Attribute-Based Access Control Model for Real-Time Availability in Cyber-Physical Systems. (http://www.cs.fsu.edu/ burmeste/TABAC.pdf)

[23] SISCO, “Cisco and SISCO Collaborate on Open Source Synchrophasor Framework, Press Release,” *http://www.sisconet.com/downloads/90-5_Cisco_SISCO.pdf*, 2011.

[24] K. Yoder, “TrouSerS The open-source TCG Software Stack,” *http://trousers.sourceforge.net/*, September 2007.