SANDIA REPORT
SAND2007-75527552
Unlimited Release
Printed November 2007

OPSAID Initial Design and Testing Report

Steven A. Hurd, Jason E. Stamp, and Adrian R. Chavez

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd.
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2007-75527552
Unlimited Release
Printed November 2007

OPSAID Initial Design and Testing Report

Steven A. Hurd
Computer and Network Security Department
Sandia National Laboratories
P.O. Box 969
Livermore, California 94551-0969

Jason E. Stamp
Energy Systems Analysis Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-1108

Adrian R. Chavez
Networked System Survivability and Assurance Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-0672

Abstract

Process Control System (PCS) security is critical to our national security. Yet, there are a number of technological, economic, and educational impediments to PCS owners implementing effective security on their systems. OPSAID (Open PCS Security Architecture for Interoperable Design), a project sponsored by the US Department of Energy's Office of Electricity Delivery and Reliability, aims to address this issue through developing and testing an open source architecture for PCS security. Sandia National Laboratories, along with a team of PCS vendors and owners, have developed and tested this PCS security architecture. This report describes their progress to date.

Acknowledgements

The authors acknowledge and thank their colleagues for their assistance with the OPSAID project.

Sandia National Laboratories: Alex Berry, Charles Perine, Regis Cassidy, Bryan Richardson, Laurence Phillips

Teumim Technical, LLC: Dave Teumim

In addition, the authors are greatly indebted to the invaluable help of the members of the OPSAID Core Team. Their assistance has been critical to the success and industry acceptance of the OPSAID project.

Schweitzer Engineering Laboratory: Rhett Smith, Ryan Bradetich, Dennis Gammel

TelTone: Ori Artman

Entergy: Dave Norton, Leonard Chamberlin, Mark Allen

The authors would like to acknowledge that the work that produced the results presented in this paper was funded by the U.S. Department of Energy/Office of Electricity Delivery and Energy Reliability (DOE/OE) as part of the National SCADA Test Bed (NSTB) Program.

Executive Summary

Process control systems (PCS) are very important for critical infrastructure and manufacturing operations, yet cyber security technology in PCS is generally poor. The OPSAID (Open PCS (Process Control System) Security Architecture for Interoperable Design) program is intended to address these security shortcomings by accelerating the availability and deployment of comprehensive security technology for PCS, both for existing PCS and inherently secure PCS in the future. All activities are closely linked to industry outreach and advisory efforts.

Generally speaking, the OPSAID project is focused on providing comprehensive security functionality to PCS that communicate using IP. This is done through creating an interoperable PCS security architecture and developing a reference implementation, which is tested extensively for performance and reliability.

This report first provides background on the PCS security problem and OPSAID, followed by goals and objectives of the project. The report also includes an overview of the results, including the OPSAID architecture and testing activities, along with results from industry outreach activities. Conclusion and recommendation sections follow. Finally, a series of appendices provide more detailed information regarding architecture and testing activities.

Summarizing the project results, the OPSAID architecture was defined, which includes modular security functionality and corresponding component modules. The reference implementation, which includes the collection of component modules, was tested extensively and proved to provide more than acceptable performance in a variety of test scenarios. The primary challenge in implementation and testing was correcting initial configuration errors.

OPSAID industry outreach efforts were very successful. A small group of industry partners were extensively involved in both the design and testing of OPSAID. Conference presentations resulted in creating a larger group of potential industry partners.

Based upon experience implementing and testing OPSAID, as well as through collecting industry feedback, the OPSAID project has done well and is well received. Recommendations for future work include further development of advanced functionality, refinement of interoperability guidance, additional laboratory and field testing, and industry outreach that includes PCS owner education.

—This page intentionally left blank —

Table of Contents

Acknowledgements 4

Executive Summary 5

1 Introduction 9
1.1 Background 9
1.1.1 Description 9
1.1.2 Historical Information 9
1.1.3 Significance 10
1.1.4 Literature Review 10
1.2 Purpose 10
1.2.1 Reason for Investigation 10
1.2.2 Roadmap Challenges 11
1.2.3 Audience 12
1.2.4 Desired Response 12
1.3 Scope 12
1.3.1 Extent and Limits of Investigation 13
1.3.2 Goals 13
1.3.3 Objectives 13

2 Approach 15
2.1 Methods 15
2.2 Assumptions 15
2.3 Procedures 15

3 Results and Discussion 17
3.1 OPSAID Project Results Summary 17
3.2 OPSAID Architecture and Development Activities 18
3.3 OPSAID Testing Activities 21
3.4 OPSAID Industry Outreach Activities 22

4 Conclusions 23

5 Recommendations 25

Appendix A: References 27

Appendix B: Acronyms, Symbols, Abbreviations 29

Appendix C: For More Information 31

Appendix D: OPSAID Reference Implementation –Details and Installation Guide 33

Appendix E: Entergy Testing Procedure 47

Appendix F: Entergy Testing Results 55

Appendix G: Sandia Throughput Testing Results 71

Appendix H: Sandia/Schweitzer Testing Results 73

Table of Figures

Figure E.1. OPSAID Field Trial Schematic....47
Figure E.2. Test Network 1....49
Figure E.3. Test Network 2....50
Figure E.4. Test Network 3....51
Figure E.5. Test Network 4....52
Figure F.1: The overall process of the visualizations....58
Figure F.2: A sample table from the raw database displayed in a PHP web page in a web browser....59
Figure F.3: The user interface to the Java Application....60
Figure F.4: The Test Network in Standard Network Configuraton....61
Figure F.5: Bandwidth data points taken for a one-minute stretch....62
Figure F.6: Network utilization in sequential network tests....63
Figure H.1: First-Round Testing Configuration....73
Figure H.2: Second-Round Testing Configuration....74
Figure H.3: Third-Round Testing Configuration....76

Table of Tables

Table 3.1: OPSAID Critical Deployment and Management Capabilities....20

1 Introduction

1.1 Background

Process control systems (PCS) are very important for critical infrastructure and manufacturing operations. These systems collect and transmit information between sensors, controllers, and central management stations; concurrently they store, process, and analyze information. They have been implemented to work in a number of physical environments using a variety of hardware, software, networking protocols, and communications technologies. Unfortunately, cyber security technology in PCS is generally poor, and not commensurate with the threat.

Technical efforts to design add-on security are neither coordinated nor comprehensive, which negatively impacts their cost, availability, and efficacy. To improve the economic proposition for securing existing control systems, an industry-owned, open and interoperable security architecture to address PCS security is needed.

1.1.1 Description

The OPSAID (Open PCS (Process Control System) Security Architecture for Interoperable Design) program is intended to address these security shortcomings in the short- and medium-term. OPSAID is a research and development project led by Sandia National Laboratories and sponsored by the Department of Energy Office of Electricity Delivery and Reliability (DOE/OE), through the National SCADA Test Bed (NSTB) program.

The OPSAID program is intended to accelerate the availability and deployment of comprehensive security technology for PCS. It will provide a design basis for vendors to build add-on security devices, in order to bring the security of existing PCS up to an acceptable level. Furthermore, the design provides a path forward for the development of inherently secure PCS elements in the future. Finally, the project will work to transition the design to an industry group for ownership to ensure its continued support.

1.1.2 Historical Information

There are many factors that have precipitated the need for the OPSAID project. One common thread among automation systems is that they were developed without adequate regard for security issues. Traditionally, PCS had relatively little in common with typical information technology (IT) systems. PCS communication was typically conducted over serial links and PCS assets were completely segregated from other IT assets. The PCS assets typically were purpose-built and did not incorporate commercial off-the-shelf technology (COTS) found in IT systems.

Changes in the nature of PCS assets and how they communicate have driven the need for the OPSAID project. To reduce costs, PCS manufacturers are increasingly incorporating COTS computer hardware and software components in new devices. This has led to an increased use of PCS communication using the Internet protocol (IP). Yet, IP-based PCS systems

typically lack many of the basic security features ubiquitous in traditional IT systems, such as detailed logging, authentication, and firewall services.

Compounding these problems, many PCS owners need to have significant information sharing between their PCS and their traditional business systems. This potentially exposes the CS to a much wider range of cyber attack, due to the network connections between the two systems.

1.1.3 Significance

It has long been held that due to their lack of security features, PCS are vulnerable to cyber attack. This is even more apparent due to the use of COTS features, IP-based connectivity, and close interconnection with business systems. The OPSAID design provides security controls, that when properly configured, will significantly mitigate these risks.

1.1.4 Literature Review

Given the infrastructure applicability of the OPSAID design, many standards from industry became relevant. A significant percentage of these were reviewed, and a review of the selected group appears in Appendix E: *Entergy Testing Procedure* as part of the testing plan. Overall, the OPSAID team reviewed IEEE standards 399 [1], PC37-1 [2], 1646 [3], P1615/D11 [4], C37-115 [5], 1588 [6], and 1613 [7], and IEC standards 61850 and 62351.

1.2 Purpose

1.2.1 Reason for Investigation

The OPSAID project is based upon previous Sandia-led research in the area of PCS security. Research at Sandia and elsewhere has focused on how to improve security and reliability over the long-term for next-generation PCS. However, as PCS are often attractive targets for adversaries and replacement cycles generally range in decades, rather than months or years, there clearly was a critical need to identify ways to address security shortcomings in the short- and medium-term, yet be complementary to next-generation PCS security solutions.

In Fall 2004, Sandia began a 2-year Laboratory Directed Research and Development (LDRD) project entitled *Applying New Network Security Technologies to SCADA Systems*. This research examined the state of the art and lessons learned in securing conventional IT networks and systems and identified approaches that could be effectively implemented in the PCS/SCADA arena.

The results of this research indicated that securing conventional IT systems and PCS/SCADA systems shared several of the same challenges. Differences were found in the relative priority of different aspects of security. For example, in some cases, a PCS may be less concerned about the confidentiality of network traffic as a conventional IT system, yet more concerned about the availability and integrity of the PCS (versus an IT system). Nevertheless, at some level, virtually all the security concerns found in the IT arena were mirrored in the PCS/SCADA arena.

However, while many of the challenges were shared, there were relatively meager resources in the PCS/SCADA arena to defend against such challenges. As was stated earlier, these systems were developed with little or no consideration for security. Thus, it is not unusual to find systems that are protected by either a single password or have no password at all. Devices typically have little or no capability to log security events. Also, these devices had no firewall to help protect them from network attack.

The research indicated that an important solution for addressing these challenges in the PCS/SCADA arena would be to "inject" security into existing PCS/SCADA networks. Specifically, the concept of "bump in the wire" security, where security appliances (that could provide a variety of needed security services) would be placed in-line between existing communication connections appeared to be the solution of choice.

Accordingly, the research team developed a basic proof-of-concept device called the Secure Linux Appliance for PCS (SLAP). This device, which could work with Ethernet traffic (IP-based) as well as serial traffic (for use with dial-up connections), was developed to provide an array of security services, including encryption, firewall, IDS, centralized logging, forensic capabilities, authentication, device management and configuration session logging. The device was developed using open source software exclusively, including the Linux OS.

Furthermore, very recently more vendors are seeing the area of add-on security for PCS as a market opportunity. However, their solutions do not address the complete set of security requirements as developed by the OPSAID project, and the existing solutions are not interoperable. In contrast, the approach taken by the NSTB OPSAID project will hasten the development of an interoperable design for PCS security.

1.2.2 Roadmap Challenges

The OPSAID project addresses two major goals of the *Roadmap to Secure Control Systems in the Energy Sector* [8]. Specifically, the goals "Develop and Integrate Protective Measures" and "Detect Intrusion and Implement Response Strategies" can be realized through using the open source software components in the OPSAID architecture, which are based on solutions used extensively in conventional IT. Specific priorities addressed by OPSAID include:

- Put non-intrusive, cost-effective, and robust SCADA encryption solutions into production,
- Develop cost-effective gateway security that includes firewalls, intrusion detection, and anti-virus protection with minimum host impact
- Enable automated collection of security information, including incident reports and visualization tools for correlation
- Develop intrusion detection system/intrusion prevention system products for control systems and audit trials for automated reporting,
- Develop and deploy sensors and sensor systems with mechanisms to detect and report anomalous activity.

1.2.3 Audience

The primary audiences for this report are two groups of stakeholders: PCS security technology manufacturers (or vendors) and PCS owners.

The specific interest in this report from manufacturers is expected to vary, depending on whether they have already developed PCS security technology. For new vendors, OPSAID provides proven security functionality through a series of open source software modules (and a collected reference implementation) that they can use to incorporate security functionality into their product. This report enumerates the specific modules used as well as the steps taken to implement the reference implementation.

For manufacturers that have already developed PCS security technology, OPSAID provides available security functionality that they may incorporate into their product. This report should assist that effort. In addition, due to pressure from PCS owners, existing PCS security technology manufacturers are concerned as to how their product may interoperate with products developed using OPSAID technology. This report begins to address this issue; additional work is planned in the coming months.

PCS owners are primarily concerned about the overall reliability of their systems. Accordingly, they are particularly concerned about any loss in reliability or performance resulting from the implementation of PCS security technology. The detailed testing results are likely to be of the most interest to the PCS owner community. However, the entire report is expected to be of interest to PCS owners who want a general understanding as to how new PCS security technology can be applied to their systems.

1.2.4 Desired Response

The desired response to this work includes the creation of a vendor/owner industry organization to sponsor collaboration on the OPSAID project (which has occurred, in the form of a PCSF interest group), as well as the incorporation of OSPAID technology or concepts into security products. Some of the desired response from industry has already occurred. A handful of PCS security technology manufacturers have developed commercially available products, with other manufacturers actively developing products. Hopefully, this report will continue to influence PCS security technology manufacturers to develop new commercially available products. Some vendors with existing security products have expressed interest in the interoperability idea. (To be fair, two have rejected the idea as against their financial best interests. This viewpoint is tightly linked to historical perspectives on automation, which feature tightly stove-piped installations.) Competition in this market is expected to drive down unit prices as well as generating PCS owner interest.

1.3 Scope

Generally speaking, the OPSAID project is focused on providing security functionality to PCS that communicate using IP. The issue of security for legacy[1] PCS equipment is best left for specific solutions, a point which was made most clearly by the peer review panelists during the OPSAID presentation at the 2006 peer review.

[1] PCS equipment ten or more years old is *legacy*.

1.3.1 Extent and Limits of Investigation

The work done between July 2006 and June 2007 primarily focused on developing and testing a prototype field device with basic operational functionality, and conducting industry outreach activities. This field device communicates with other OPSAID devices using the Internet Protocol (IP). At this time, the OPSAID project is not addressing serial communication between OPSAID devices. In addition, due to industry feedback, the project is not addressing environmental hardening of the systems, as well as considering a graphical device management application.

1.3.2 Goals

As mentioned in section 1.2.2, *Roadmap Challenges*, the OPSAID project goals are closely aligned to the overall NSTB goals. Primarily, OPSAID's goals are focused on making encryption and other security services available to PCS/SCADA systems. We believe the OPSAID design and development will help enable the following:

- Understanding impacts of system security features on system operation.
- Shielding hosts with known vulnerabilities from the PCS network.
- Adding monitoring and visualization for PCS security.
- Greatly improving configuration access to PCS devices.
- Adding distributed firewalls to reduce network flexibility.
- Consider issues with deployment complexity on OPSAID device configuration.
- Providing a blueprint for future, inherently secure PCS devices (that can interoperate with OPSAID-protected legacy devices).
- Providing an open, interoperable architecture for vendors to build add-on security devices.

1.3.3 Objectives

Objectives for the OPSAID project included the following:

- Integrate and test fully functional field device
- Test operational impacts of the system
- Develop security database & visualization system capabilities
- Develop configuration and key management tools
- Conduct industry partner outreach
- Conduct field test of OPSAID prototype
- Deliver OPSAID design report to NSTB

—This page intentionally left blank —

2 Approach

2.1 Methods

To meet project objectives, the OPSAID team is involved in system development, integration, and testing. The work on developing an initial prototype would take place at Sandia National Laboratories. In parallel, industry outreach would be conducted, so that feedback could be incorporated into the design and partners would be available to assist in testing the prototype.

Once a working prototype was developed, the OPSAID team would work with partners to test the prototype under a variety of conditions and iteratively refine the technology. And once the industry partnerships started to form, we incorporated their views on capability and interoperability into the OPSAID development.

2.2 Assumptions

The OPSAID project is predicated upon the following assumptions:

- PCS owners are concerned about the security of their PCS assets, and are seeking cost-effective solutions to improve their PCS security posture.
- PCS assets are increasingly being connected beyond the traditional control network (e.g. business networks).
- The security challenges for PCS grow rapidly as PCS assets are more widely interconnected.
- PCS assets are increasingly being deployed using COTS computer hardware and software.
- Industry ownership of an interoperable design will be more effective at promoting cyber security than fragmented competing designs, or alternatively a government standard forced upon industry

2.3 Procedures

The tasks map directly to the completion of the objectives in section 1.3.3:

- Integrate and test fully-functional field device
  1. Complete initial design based on a selected hardware platform and a compatible Linux distribution – the hardware was a mini-ITX-based system [2] with Debian and Ubuntu Linux.
  2. After initial testing, refine the design to incorporate improvements.

[2] Mini-ITX is a 17 x 17 cm low-power motherboard form factor developed by VIA Technologies (http://en.wikipedia.org/wiki/Mini-ITX).

- Test operational impacts of the system
  1. Conduct internal performance trials – testing was between Sandia's New Mexico and California sites.
  2. Conduct testing at a utility site – the OPSAID partner utility Entergy provided the test site; the testing was conducted in Baton Rouge, Louisiana.
  3. Conduct testing with a vendor – Schweitzer Engineering Labs (SEL) in Pullman, Washington provided one end of the test environment and SNL/New Mexico the other.

- Demonstrate security database & visualization system
  1. Demonstrate a prototype security database using open-source software.
  2. Demonstrate a prototype visualization system that leverages the database and also leverages open-source technology.

- Demonstrate configuration and key management tools
  1. Initial development and testing at Sandia
  2. Conduct testing with OPSAID partners at a remote site

- Industry partner outreach
  1. Actively seek industry partners in both the vendor and owner communities – to date, SEL Teltone, Entergy, and the Tennessee Valley Authority have partnered strongly with the OPSAID program, with others interacting as well.
  2. Obtain industry sponsorship for an OPSAID support group – the project was successful at creating a PCSF interest group.
  3. Solicit industry input on OPSAID technology and programmatic development.

- Deliver OPSAID design report to NSTB
  1. Develop a draft report
  2. Release OPSAID report to audience
     a. Obtain approval of Sandia National Laboratories
     b. Obtain approval of DoE Office of Electricity Delivery and Energy Reliability, the sponsor of the OPSAID project

Technical detail for these tasks and procedures are presented in the following section (description of the final task is not included since it is outside the timeline of the report narrative).

3 Results and Discussion

This section is organized in three parts: OPSAID Architecture and Development Activities, OPSAID Testing Activities, and OPSAID Industry Outreach Activities.

3.1 OPSAID Project Results Summary

At a high level, the OPSAID project achieved the following:

1. **OPSAID architecture and development**
   a. Operational OPSAID platform implemented on mini-ITX system with open-source Linux OS.
   b. Configuration and installation guides developed for server and client OPSAID devices (see Appendix D: *OPSAID Reference Implementation*).
   c. OPSAID security database and visualization system demonstrated during testing (see Appendix F: *Entergy Testing Results*).
      i. Open-source security database demonstrated (see section 4.1.3. *MySQL Server Logging* in Appendix F: *Entergy Testing Results*).
      ii. Open-source visualization system demonstrated (see section 4.1.4. *Visualization and Monitoring* in Appendix F: *Entergy Testing Results*).
   d. OPSAID configuration and key management demonstrated (see Appendix H: *Sandia/Schweitzer Testing Results*)

2. **OPSAID testing**
   a. Completed internal performance trials between Sandia's New Mexico and California sites (see Appendix G: *Sandia Throughput Testing Results*)
   b. Completed functional testing at a utility site (see Appendix F: *Entergy Testing Results*)
   c. Completed remote functional testing at vendor site (Appendix H: *Sandia/Schweitzer Testing Results*)

3. **OPSAID industry outreach**
   a. SEL Teltone, Entergy, and the Tennessee Valley Authority are partnered with the OPSAID program
   b. PCSF interest group created (see the OPSAID interest group web page at https://www.pcsforum.org/groups/79/)
   c. Well-attended OPSAID sessions at March 2007 Process Control Systems Forum
   d. Successful operations tests conducted at industry sites (See item 2 of this list, *OPSAID Testing*)

3.2 OPSAID Architecture and Development Activities

The OPSAID architecture has evolved through the course of developing and testing a prototype field device. At one point, OPSAID was envisioned as a very specific standard, such that all devices that were OPSAID-compliant would interoperate completely. However, due to industry feedback, it became clear that OPSAID would not gain widespread success as a monolithic standard. This makes sense, given the large number of security services OPSAID provides and the ever-changing security environment. Accordingly, the vision for OPSAID evolved into the architecture that follows.

OPSAID is fundamentally a security architecture that incorporates several aspects of security functionality. The overall OPSAID architecture is divided into several areas of security functionality. Examples of areas of security functionality include network-based intrusion detection and firewall services.

For each area of security functionality, it is likely that there exist several practical technical approaches to provide that functionality. The basic OPSAID philosophy has been to implement the most commonly used, well-proven, open source solution for each area of security functionality. That way, we address our first goal: Providing an accelerated path for vendors to implement security functionality into their products.

Yet at the same time, it is clear that it is not appropriate for vendors to incorporate a solution for each area of security functionality into each of their products. Furthermore, it is clear that for a specific product and a given area of security functionality, there may be a better solution (today, or certainly in the future) than what was initially chosen for OPSAID. These facts indicate an “all or nothing” standard would not be appropriate, an approach that has been confirmed through industry feedback.

Thus, the OPSAID team has adopted a structure regarding standards that is aimed at maximizing simplicity, flexibility and interoperability among systems. We aim to accomplish this through:

- OPSAID Security Functions
- OPSAID Component Modules (with corresponding interoperability guides)
- The OPSAID Reference Implementation

As mentioned above, an OPSAID Security Function relates to a specific functional role in security, such as network-based intrusion detection or firewall services. The currently defined OPSAID Security Functions are:

- Virtual Private Networking/Encryption
- Firewall Services
- Network Intrusion Detection System
- Host Intrusion Detection System
- Event Logging
- Event Database Storage, Alert Generation & Visualization
- End-device Configuration Session Logging
- Authentication
- Device Management

An OPSAID Component Module is a specific implementation of an OPSAID Security Function. For example, for the OPSAID Security Function "Virtual Private Networking/ Encryption", one OPSAID Component Module would be IPSec. At a later date, an additional OPSAID Component Module for the OPSAID Security Function "Virtual Private Networking/Encryption" could be SSL. Each Component Module will contain detailed information about configuration and interoperability in its interoperability guide. At this time, we are in the process of producing detailed OPSAID Component Module information, which will be available on the OPSAID website.

The OPSAID Reference Implementation is a collection of specific hardware and software (and corresponding configuration information) that implements all OPSAID Security Functions through OPSAID Component Modules. This implementation has its roots in our original "proof of concept" system and has matured to be used as both a point of reference and a platform for testing basic interoperability. However, we do not expect that any vendor will choose to implement the exact OPSAID reference platform as a product. The overall OPSAID Reference Implementation will be iterated each time there is a change, addition or deletion of any OPSAID Component Module. Details regarding the current Reference Implementation as well as an installation guide can be found in Appendix D: *OPSAID Reference Implementation*.

Our hope is that for a particular vendor product, they would complete a checklist as to how they implement or interoperate with each specific OPSAID Component Modules. That way, owner/operators can easily compare functionality among products as well as assess interoperability prospects.

Finally, to help direct future development efforts regarding OPSAID deployment and on-going management, the OPSAID team developed a list of critical deployment and management capabilities. These capabilities, listed in Table 3.1 below, are partitioned into activities specific to an OPSAID component module, an OPSAID system, or a series of OPSAID devices (referred to as a "swarm").

Table 3.1: OPSAID Critical Deployment and Management Capabilities

	Deployment	Management
Module	Modules should be deployed with an easy to use package management utility. The package management utility should ensure modules do not conflict. The package management utility should allow for local and remote installation of packages. An automated or scripted deployment feature should be included in the package management utility. Once installed, the package management utility should initiate an easy configuration utility.	Modules should be managed with a management utility. The management utility should have a simple interface and be easy to use. The management utility should run both locally and remotely. Updates and rollbacks should be implemented in the management utility. The management utility should check for modules that may be configured incorrectly or insecurely. Logging of events occurring in the modules should be displayed in the management utility.
System	Systems should be deployed with an easy to use local and remote installation utility. The installation utility should allow for interactive or scripted deployment. The installation utility should initiate an easy configuration utility for the packages installed on the system.	Management of a system should be accomplished with a utility that provides local and remote administrative capabilities. The logging of system events should be reported to the management utility. The management utility should check for system configurations that may be configured incorrectly or insecurely.
Swarm	A swarm (multiple systems) should be deployed with an easy to use local and remote installation utility. The installation utility should allow for interactive or scripted deployment. The installation utility should initiate an easy configuration utility for the packages installed on the system.	Management of a swarm should be accomplished with a utility that provides remote administrative capabilities for multiple systems. The logging of multiple systems' events should be reported to the management utility. The management system should ensure that all systems are communicating properly and securely.

3.3 OPSAID Testing Activities

To work towards the overall goals and objectives for the OPSAID project, a series of OPSAID testing activities were planned and executed. This section includes summary information regarding each testing activity as well as the location of detailed testing information. In each of the testing activities, most, if not all of the following functions were tested and recorded:

- Normal Operation, after introduction of OPSAID
- Key Management
- OPSAID's VPN Services
- OPSAID Interoperability
- Security Event Logging & Monitoring

Network Metrics

All of the above OPSAID functions were successfully tested in a variety of scenarios. The overwhelming majority of the problems that occurred while testing were resolved by simple configuration changes. This is encouraging, yet does reinforce the need for careful planning in advance of deployment.

The first major testing activity was conducted in partnership with Entergy Corporation at Entergy's testing labs. This testing activity had a number of objectives, including:

- Familiarizing Entergy technical staff with OPSAID and OPSAID operations
- Testing basic implementation and configuration processes

Testing basic operations with OPSAID devices in place

On the whole, this testing was successful. Once the network settings for the OPSAID devices were properly configured, the OPSAID devices did not introduce any errors into the network and the protected devices were able to operate as usual, which is a primary success metric for OPSAID.

Detailed information regarding these tests can be found in Appendix E: *Entergy Testing Procedure* and Appendix F: *Entergy Testing Results*. Please keep in mind that these tests were not aimed at thoroughly testing the security efficacy of the OPSAID system.

The second testing activity was conducted at Sandia, and was solely aimed at determining how the OPSAID devices would operate when in a very high bandwidth environment. Specifically, would the introduction of OPSAID devices result in unacceptable latency or other network delivery problems.

This testing demonstrated that introducing OPSAID into a high-bandwidth environment generally had relatively little performance impact, certainly no more than would be expected. Detailed information regarding these tests can be found in Appendix G: *Sandia Throughput Testing Results*.

The third testing activity was conducted between Sandia and Schweitzer Engineering Laboratory. This activity was designed to test OPSAID capabilities across the Internet, interoperability between Sandia-implemented and Schweitzer-implemented OPSAID devices, and various aspects of IPSec key management.

The testing proved largely successful. Initial connectivity was achieved across the Internet and the Sandia- and Schweitzer-implemented OPSAID devices were able to interoperate. Initially, a few problems were encountered with respect to IPSec operating properly using certificates and a certificate authority as well as a problem with syslog-ng messages not being encrypted between the OPSAID devices. However, after minor configuration errors were corrected, all issues were resolved in further testing with Schweitzer. Detailed information regarding these tests can be found in Appendix H: *Sandia/Schweitzer Testing Results*.

OPSAID Industry Outreach Activities

While the OPSAID Architecture, Development, and Testing activities have produced solid results, OPSAID's outreach to industry has been notable. From the earliest days of the OPSAID project, team members were meeting with potential PCS vendor and owners. These meetings helped shape the OPSAID Core Team. This team includes two vendors (Schweitzer Engineering Laboratories and TelTone) and one PCS owner (Entergy). In addition, an additional PCS owner (Tennessee Valley Authority) is participating in a proposed related commercialization effort. Simply put, the testing activities listed previously would not have been possible without the participation of these partners.

Information regarding OPSAID has been presented in several forums. The largest forum was at the March 2007 Process Control Systems Forum (PCSF) in Atlanta, GA. Two different sessions focusing on OPSAID were well attended, resulting in the formation of an OPSAID PCSF interest group.

These sessions prompted several follow-on discussions with individual PCS owners and vendors. The feedback from these discussions was invaluable, pointing a clear path forward as to how OPSAID can assist them to reach the overarching goal: Reducing the risk of energy disruption through mitigating cyber attack vulnerabilities.

Generally speaking, PCS owners expressed concern about reliability and interoperability issues. They expressed interest in seeing detailed testing information, from an entity without a direct profit motive. They also wanted some assurance that they equipment they purchased today would work with equipment they purchased tomorrow from a different vendor.

PCS vendors that have already developed PCS security products agreed that the availability of advanced functionality would benefit all vendors. Deep protocol analysis for SCADA protocols and creating end-device aware security extensions were given high marks for being functionality they would incorporate into their products. As one might imagine, given the risks of avoiding vendor "lock-in", vendors were not of one mind in their support for interoperability. Several vendors stressed the credibility gained by having OPSAID led by a national laboratory. They stressed the importance of educating the PCS owner community as to the value and drawbacks of the OPSAID approach.

In the interest of full disclosure, it should be noted that the OPSAID team spoke to PCS vendors who were not supportive of the OPSAID project. The OPSAID team is committed to exploring plausible alternative approaches towards reaching the overarching goals suggested by any PCS vendors or owners.

4 Conclusions

The results of OPSAID development and testing are encouraging. The prototype OPSAID device performs well, both in terms of security efficacy and low insertion impact for existing PCS/SCADA networks. The program has also created prototypes or basic specifications for most ancillary capabilities. The outreach activities have been successful in generating interest in OPSAID among PCS security technology manufacturers and PCS owners. One measure of success can be seen through the development and release of first-generation commercial PCS security technology products, providing much of the security functionality identified as part of the OPSAID architecture. Hence, the OPSAID specification has been developed so that it is general enough to fit a variety of customers needs, but also detailed enough to provide effective security features for a PCS.

Thus, the question of whether OPSAID (or OPSAID-like) devices can be developed is no longer in doubt. The reference implementation has been extensively tested and has been proven to be initially successful both with interoperability and providing a higher level of security for PCS. OPSAID has a high potential to become a de facto standard for securing PCS and has generated interest from a variety of vendors.

Yet, user deployment of PCS security technology is slow. To accelerate the deployment of PCS security technology, it is clear that focusing on refining interoperability guidance, developing advanced functionality, continuing test activities, and educating the user community are ways the OPSAID project could continue to be of service.

—This page intentionally left blank —

5 Recommendations

As stated in section 4, *Conclusions*, it is clear that OPSAID (or OPSAID-like) devices can be successfully developed. However, these devices have not yet been widely deployed in the field. The recommendations for the path forward for OPSAID are aimed at eliminating or reducing the barriers to widespread field deployment of OPSAID devices.

PCS owners have expressed concern that PCS security technology must be interoperable. Yet, as PCS security technology is far from "one size fits all", it is difficult for PCS manufacturers to gauge how best to provide security functionality while interoperating with products from other PCS manufacturers. The OPSAID project can serve the community by developing more detailed PCS security interoperability guidance. The most critical area for such guidance is in the area of Virtual Private Networking/Encryption, including key management and public key infrastructure (PKI) support. The OPSAID team recommends further work be undertaken providing more detailed interoperability guidance for VPN and PKI services.

PCS owners have expressed interest in continued testing of OPSAID technology. Continued testing in PCS owner laboratory settings, leading to initial field testing, is an appropriate path forward. The OPSAID team recommends further laboratory and field testing of OPSAID devices.

As stated in the Results and Discussion, there was unanimous support from PCS vendors who have already developed security products for the development of advanced technology. While "deep protocol analysis for DNP3" was suggested, the OPSAID team believes the highest leverage activity in advanced technology development would be to create end-device aware security capabilities. The OPSAID team recommends the development of a prototype in this area.

The OPSAID team did not initially set out with the goal of educating the PCS owner population as to the value of OPSAID (or similar technologies). Nevertheless, this is an area where OPSAID can further aid the adoption of this technology. The OPSAID team recommends continued industry outreach activities, focused at educating PCS owners as well as receiving continuous feedback as to the value of OPSAID. These activities are invaluable in ensuring future OPSAID work is consistent with the needs and wishes of industry.

—This page intentionally left blank —

Appendix A: References

[1] IEEE 399-1997 (Brown Book), *IEEE recommended practice for industrial and commercial power systems analysis*, 1997. http://ieeexplore.ieee.org/ISOL/standardstoc.jsp?punumber=5891

[2] IEEE PC37.1/D5, Sep. 07, *IEEE Draft Standard for SCADA and Automation Systems*, 2007. http://ieeexplore.ieee.org/ISOL/standardstoc.jsp?punumber=4343657

[3] IEEE 1646-2004, *Standard Communication Delivery Time Performance Requirements for Electric Power Substation Automation*, 2004. http://ieeexplore.ieee.org/ISOL/standardstoc.jsp?punumber=9645

[4] IEEE Draft Standard P1615/D11, Nov. 2006, *IEEE Recommended Practice for Network Communication in Electric Power Substations*, 2006. http://ieeexplore.ieee.org/ISOL/standardstoc.jsp?punumber=4040415

[5] IEEE C37.115-2003, *IEEE standard test methods for use in the evaluation of message communications between intelligent electronic devices in an integrated substation protection, control and data acquisition system*, 2003. http://ieeexplore.ieee.org/ISOL/standardstoc.jsp?punumber=9160

[6] IEEE Standard 1588-2002, *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, 2002. http://ieeexplore.ieee.org/ISOL/standardstoc.jsp?punumber=8117

[7] IEEE Standard 1613-2003, *IEEE standard environmental and testing requirements for communications networking devices in electric power substations*, 2003. http://ieeexplore.ieee.org/ISOL/standardstoc.jsp?punumber=9144

[8] *Roadmap to Secure Control Systems in the Energy Sector*, U.S. DOE and U.S. DHS, prepared by Energetics Incorporated, January 2006.

[9] "Linux Kernel 2.6 using KAME tools," *IPsec HOWTO*, September 2006. http://www.ipsec-howto.org/x299.html

[10] "OpenSSH", *Linux Security.com*, September 2006. http://www.linuxsecurity.com/content/view/117274/

[11] Martin, F., "SSL Certificates HOWTO", The Linux Documentation Project *website*, September 2006. http://tldp.org/HOWTO/SSL-Certificates-HOWTO/

[12] Racoon man page: http://www.die.net/doc/linux/man/man8/racoon.8.html

[13] Racoon.conf man page: http://www.die.net/doc/linux/man/man5/racoon.conf.5.html

[14] Setkey man page: http://www.die.net/doc/linux/man/man8/setkey.8.html

—This page intentionally left blank —

Appendix B: Acronyms, Symbols, Abbreviations

BIOS	Basic Input/Output System
CA	Certificate Authority
CD-ROM	Compact Disc Read-Only Media
CIDR	Classless Inter-Domain Routing
COTS	Commercial Off-The-Shelf
CRL	Certificate Revocation List
DHCP	Dynamic Host Configuration Protocol
DNP	Distributed Network Protocol
DOE	Department of Energy
DOE/OE	Department of Energy Office of Electricity Delivery and Reliability
IDE	Integrated Drive Electronics
IED	Intelligent Electronic Device
IEEE	Institute of Electrical and Electronic Engineers
IP	Internet Protocol
IPSec	Internet Protocol Security
IT	Information Technology
LDRD	Laboratory Directed Research and Development
MSS	Maximum Segment Size
MTU	Maximum Transmission Unit
NSTB	National SCADA Test Bed
OCSP	Online Certificate Status Protocol
OPSAID	Open PCS Security Architecture for Interoperable Design
PCS	Process Control System
PKI	Public Key Infrastructure
SCADA	Supervisory Control And Data Acquisition
SCSI	Small Computer System Interface
SEL	Schweitzer Engineering Laboratories
SNL	Sandia National Laboratories
SSH	Secure Shell
TCP	Transmission Control Protocol
TVA	Tennessee Valley Authority
UDP	User Datagram Protocol
USB	Universal Serial Bus
VPN	Virtual Private Network
WAN	Wide Area Network

—This page intentionally left blank —

Appendix C: For More Information

Author	Steven Hurd (sahurd@sandia.gov) Computer & Network Security Department P.O. Box 969 Livermore, California 94551-0969
National SCADA Testbed (NSTB) Project	Jennifer DePoy, Manager (jdepoy@sandia.gov) Critical Infrastructure Systems Department Sandia National Laboratories P.O. Box 5800 Albuquerque, New Mexico 87185
DOE Office of Electricity Delivery and Reliability *Control Systems Security* web page	http://www.oe.energy.gov/randd/css.htm
DOE *Roadmap to Secure Control Systems in the Energy Sector*	http://www.oe.energy.gov/randd/438.htm
National SCADA Test Bed	http://www.oe.energy.gov/randd/487.htm
OPSAID Website	http://www.sandia.gov/scada/opsaid.htm
Sandia Center for SCADA Security	http://www.sandia.gov/scada/

—This page intentionally left blank —

Appendix D: OPSAID Reference Implementation – Details and Installation Guide

This guide provides details on the installation of an OPSAID-compliant Linux system and applications. It assumes that hardware compatibility and other basic operating system issues can be addressed successfully, and does not provide procedures to cover them (reference [9] will be useful if assistance is needed). Please note this procedure includes basic key management setup through a system Certificate Authority.

The following hardware and software (including version information) was used in developing this reference implementation:

Hardware

Field Unit and Central Unit

Case:	Casetronic C159 Rackmount Mini-ITX Case with Rackmount Backplate
Motherboard/CPU:	EPIA EK 10000G / C3
RAM:	DDR-266 1GB
Hard Drive:	IGS 1GB Flash Disk Module
Network Card:	Soekris 1641 Four Port NIC

Central Unit Only

Hard Drive:	Samsung 40GB SATA 5400 rpm Hard Drive
Cable:	SATA Hard Drive Cable

Operating System

GNU/Linux:	Debian 4.0 Etch

Applications

IPSec:	StrongSwan 4.1.2 openssl 0.9.8c
Firewall:	iptables 1.3.6
IDS:	Snort 2.3.3
HIDS:	OPSAID-HID with md5deep, logger, awk and diff
Management:	openssh 4.3
Logging:	syslog-ng 2.0.0
Reports:	apache2 mysql5 php5 php-syslog-ng 2.8

OPSAID Installation Guide

General Notes for OPSAID Installation:

1. In this document the Control unit is referred to as the server and the Field Unit is referred to as the client.
2. These installation instructions assume you have an active Internet connection on the OPSAID device. The document does not provide details regarding the downloading of required files on an Internet-connected system and transferring the files to the OPSAID device.
3. Command line arguments are in a different sans-serif font (e.g. echo “This is an example”).
4. Screen text appears in single quotes.
5. Keystrokes are given as the name of the key in angle brackets (e.g. <enter>).
6. Placeholders are in square brackets (i.e. [Example]).
7. Emacs, nano and vi are installed on the OPSAID devices by default. For editing a file, the example assumes vi; however, you may use any editor you feel comfortable with. For beginners, nano is the easiest editor to use.

I. OPSAID Control Unit (Server) Installation

1 Network information. You will need to know the following information before you continue the installation of the server.
  1.1 Hostname:
  1.2 Domain Name:
  1.3 Internet facing IP address:
  1.4 Internet facing netmask:
  1.5 Internet facing domain name server address(es):
  1.6 Internet facing broadcast address:
  1.7 Internet facing gateway address:
  1.8 Internal IP address:
  1.9 Internal netmask:
  1.10 Internal CIDR block address:
  1.11 Client's Internet facing IP address:

2 Hardware required
  2.1 OPSAID Control Unit
  2.2 USB CDROM

3 Booting from CD
  3.1 Download and burn Debian 4.0 disc 1 from debian.org.
  3.2 Insert CD into a USB CDROM connected to the server.
  3.3 During BIOS checks, press <delete>.
  3.4 Select 'Advanced BIOS Features'.
  3.5 Select 'USB-CDROM for First Boot Device'.
  3.6 Select 'Hard Disk for Second Boot Device'.
  3.7 Press <esc> then choose 'Save & Exit Setup'.

4 Debian install. The install process is performed in a menu-driven environment. Most of the items are self-explanatory or are described in the install.
  4.1 Press <enter> to begin the install.
  4.2 Select a language.
  4.3 Select an area.
  4.4 Select a keyboard layout.
  4.5 Configure network
    4.5.1 Select 'Go Back' if you were not prompted to configure your network.
    4.5.2 Select 'Configure network manually'.
    4.5.3 Set the Internet facing IP address.
    4.5.4 Set the Internet facing netmask address.
    4.5.5 Set the Internet facing gateway address.
    4.5.6 Set the domain name server address(es).
  4.6 Set the host name.
  4.7 Set the domain name.
  4.8 Partition disks
    4.8.1 Select 'Guided – use entire disk`.
    4.8.2 Select 'IDE1 Master'.
    4.8.3 Select 'All files in one partition'.
    4.8.4 Select 'SCSI1'

4.8.5 Select ‘Yes’
4.8.6 Select ‘pri/log’ under SCSI1
4.8.7 Select ‘Create a new partition’
4.8.8 Select ‘Continue’
4.8.9 Select ‘Primary’
4.8.10 Select ‘Mount Point”
4.8.11 Select ‘/var – variable data’
4.8.12 Select ‘Done setting up the partition’
4.8.13 Select ‘Finish’
4.8.14 Select ‘Yes’

4.9 Select the appropriate time zone.
4.10 Set the root (administrator) password.
4.11 Add a user and set the password for the user.
4.12 Configure the package manager
4.12.1 Select ‘Yes’ for network mirror.
4.12.2 Select an appropriate network mirror.
4.12.3 Set a proxy if applicable.
4.12.4 Select ‘No’ for the package survey.

4.13 Software selection
4.13.1 ‘Standard system’ should already be selected. Press <tab> to select ‘Continue’ then press <enter>.

4.14 Boot loader
4.14.1 Select ‘Yes’.

4.15 Finished
4.15.1 Select ‘Continue’ (Note: This will reboot the computer).

5 Remove excess packages
5.1 Log in as root.
5.2 apt-get remove exim4-base nfs-common sysklogd portmap pidentd openbsd-inetd
5.3 Type yes
5.4 rm /var/cache/apt/archives/*

6 Package install
6.1 apt-get install build-essential binutils gcc g++ libc6-dev libgmp3-dev syslog-ng openssh-server openssl tcpdump bzi
6.2 Type yes
6.3 wget http://www.sandia.gov/scada/opsaid/ref-1.0/config.tar.bz2
6.4 tar –xvjf config.tar.bz2 –C /
6.5 rm config.tar.bz2

7 Configure secondary network interface
   7.1 vi /etc/network/interfaces
      Change the line: 'auto lo'
         To: 'auto lo eth0 eth1'
      Add the following to the bottom of the file:
         iface eth1 inet static
         address <Internal IP>
         netmask <Internal netmask>
         network <Internal CIDR block address before the forward slash>
         broadcast <Internal broadcast address>
      Save the file and exit.

8 Install and configure snort
   8.1 apt-get install snort
   8.2 Type yes
   8.3 You will be prompted to enter your home network. This is the internal network CIDR block address.
   8.4 vi /etc/snort/snort.debian.conf
      Change the line: DEBIAN_SNORT_OPTIONS=""
         To: DEBIAN_SNORT_OPTIONS="-s"
      Save the file and exit.

9 Set up syslog-ng
   9.1 cp /root/config/server/syslog-ng.conf /etc/syslog-ng/syslog-ng.conf

10 Set up IP forwarding
   10.1 vi /etc/sysctl.conf
      Change the line: #net.ipv4.conf.default.forwarding=1
         To: net.ipv4.conf.default.forwarding=1
      Save the file and exit.

11 Set up firewall. The firewall provided is very simple and should be modified to fit your network and needs.
   11.1 /bin/sh /root/config/firewall.sh
   11.2 iptables-save >/etc/firewall.conf
   11.3 vi /etc/network/if-up.d/firewall
      Add the following lines:
         #!/bin/sh
         iptables-restore < /etc/firewall.conf
      Save file and exit
   11.4 chmod +x /etc/network/if-up.d/iptables

12 Set up Host-Based Intrusion Detection system. The default settings should be adequate for most. If your configuration requires an alternate setup, notes are provided in the scan.conf file.
   12.1 tar –xvzf /root/config/hid.tgz –C /
   12.2 vi /etc/scan.conf
      Save the file and exit.

13 Install and setup StrongSwan

13.1 Install StrongSwan

13.1.1 wget http://download.strongswan.org/strongswan-4.1.2.tar.bz2

13.1.2 tar –xvjf strongswan-4.1.2.tar.bz2

13.1.3 cd strongswan-4.1.2

13.1.4 ./configure --prefix=/usr --sysconfdir=/etc

13.1.5 make

13.1.6 make install

13.1.7 cd ../

13.1.8 rm –fr strongswan*

13.1.9 cp /root/config/openssl.cnf /etc/ssl/

13.2 Create Certifcate Authority

13.2.1 cd /etc/ssl/

13.2.2 /usr/lib/ssl/misc/CA.pl -newca l

CA certificate filename (or enter to create) <enter>

Enter PEM pass phrase: (Note: This is your certificate authority password)

Verifying password - Enter PEM pass phrase:

Answer the questions posed

13.2.3 cd demoCA/

13.2.4 openssl x509 -in cacert.pem -days 3650 -out cacert.pem -signkey ./private/cakey.pem

Getting Private key

Enter PEM pass phrase: (Note: Use certificate authority password)

13.2.5 cd ../

13.3 Create Server Certificate

13.3.1 /usr/lib/ssl/misc/CA.pl -newreq

Generating a 2048-bit RSA private key

.....++++++

.....++++++

writing new private key to 'newkey.pem'

Enter PEM pass phrase: (Note: This is your server's private key password)

Verifying - Enter PEM pass phrase:

You should write down the answers to the following questions: Country, State, Location, Organization, OganizationUnit, CommonName, Email

These will be referred to as the [ServerCountry], [ServerState], [ServerLocation], [ServerOrganization], [ServerOU], [ServerCommonName], [ServerEmail]

13.3.2 /usr/lib/ssl/misc/CA.pl -sign

Enter passphrase for ./demoCA/private/cakey.pem: (Note: Use your certificate authority password)

<y>

<y>

13.3.3 mv /etc/ssl/certs/newcert.pem /etc/ipsec.d/certs/server_cert.pem

13.3.4 mv /etc/ssl/certs/newkey.pem /etc/ipsec.d/private/server_key.pem

13.3.5 cp /etc/ssl/certs/demoCA/cacert.pem /etc/ipsec.d/cacerts/

13.4 Create Client Certificate

13.4.1 /usr/lib/ssl/misc/CA.pl -newreq

Generating a 2048-bit RSA private key

.....++++++

.....++++++

writing new private key to 'newkey.pem'

Enter PEM pass phrase: (Note: This is your client's private key password)

Verifying – Enter PEM pass phrase:

You should write down the answers to the following questions:

Country, State, Location, Organization, OganizationUnit, CommonName, Email

These will be referred to as the

[ClientCountry], [ClientState], [ClientLocation], [ClientOrganization], [ClientOU], [ClientCommonName], [ClientEmail]

13.4.2 /usr/lib/ssl/misc/CA.pl -sign .

Enter pass phrase for ./demoCA/private/cakey.pem: (Note: Use your certificate authority password)

<y>

<y>

13.4.3 mv newcert.pem client_cert.pem

13.4.4 mv newcert.pem client_key.pem

13.4.5 We will pull the certificates and keys from the server later.

13.5 Set up StrongSwan

13.5.1 cp /root/config/server/ipsec.secrets /etc/

13.5.2 vi /etc/ipsec.secrets

Insert your server private key password in the "[password]" field.

Save the file and exit.

13.5.3 rm /etc/ipsec.conf

13.5.4 vi /etc/ipsec.conf

Add the following lines:

```
version 2.0
conn net-net
    left=[Internet IP]
    leftcert=server_cert.pem
    leftsourceip=[Internal IP]
    leftsubnet=[Server's Internal CIDR block address]
    right=[Client's Internet IP]
    rightsubnet=[Client's Internal CIDR block address]
    rightid="C=[ServerCountry],
            ST=[ServerState],
            L=[ServerLocation],
```

```
                    O=[ServerOrganization],
                    OU=[ServerOU],
                    CN=[ServerCommonName],
                    E=[ServerEmail]”
            compress=yes
            auto=start
```

Save the file and exit.

13.5.5 cp /root/config/ipsec /etc/init.d/

13.5.6 chmod 755 /etc/init.d/ipsec

13.5.7 ln -sf /etc/init.d/ipsec /etc/rc2.d/S99ipsec .

14 Set up SSH

14.1 cp /root/config/sshd_config /etc/ssh .

14.2 su [user created during setup] .

14.3 ssh-keygen –t dsa .

14.3.1 To save the key in the default location press <enter>.

14.3.2 Enter passphrase.

14.3.3 Repeat passphrase.

15 Install apache, mysql, php and php-syslog-ng:

apt-get install apache2 php5 libapache2-mod-php5 mysql-server mysql-client php5-mysql

15.1 Set up mysql:

```
mysql -u root
mysql> use mysql;
mysql> grant all on accounts.* to [new admin]@localhost identified by '[new admin password]';
mysql> flush privileges;
mysql> exit
mysqladmin -u root password [new root password]
mkfifo /var/log/mysql.pipe
```

15.2 Install php-syslog-ng

```
wget “http://downloads.sourceforge.net/php-syslog-ng/php-syslog-ng-2.9.1r10.tar.gz?modtime=1152439438&big_mirror=0”
mkdir /var/www/phpsyslogng/
tar -xvzf php-syslog-ng-2.9.1r10.tar.gz –C /var/www/phpsyslogng/
chown -R root:root /var/www/phpsyslogng/
chmod -R 777 /var/www/phpsyslogng/html/config/
reboot
```

15.3 Log back in as root

15.4 Install php-syslog-ng

15.4.1 On a secondary system, open a web browser to the following address: http://[ServerIPaddress]/phpsyslogng/html/install/

15.4.2 If there are no red marks on the page, click next.

15.4.3 Accept the license and click next.

15.4.4 Step 1
15.4.4.1 Set and verify the MySQL Password to [new root password].
15.4.4.2 Set and verify the Syslog User password.
15.4.4.3 Set and verify the Syslog Admin password.
15.4.4.4 Remove check from the “Install sample data” box.
15.4.4.5 Click next.
15.4.5 Step 2
15.4.5.1 Set site name.
15.4.5.2 Click next.
15.4.6 Step 3
15.4.6.1 Verify settings. The defaults should be correct.
15.4.6.2 Enter an email address.
15.4.6.3 Write down or change password.
15.4.6.4 Click Next
15.4.7 Step 4
15.4.7.1 Click view site
15.4.8 Switch back to the server.
15.4.9 rm -fr /var/www/phpsyslogng/html/install/

15.5 Modify syslog-ng
15.5.1 cp /root/config/server/phpsyslog-ng.conf /etc/syslog-ng/syslog-ng.conf
15.5.2 vi /etc/syslog-ng/syslog-ng.conf
Move to the bottom of the page.
Change the field [Client’s Internal IP] to the appropriate value.
Change the field [Server’s Internal IP] to the appropriate value.
Change the field [Syslog User Password] to the appropriate value.
Save the file and exit

16 Reboot the system
16.1 Type reboot

II. OPSAID Field Unit (Client) Installation

1. Network information. You will need to know the following information before you continue the installation of the client.
   - 1.1 Hostname:
   - 1.2 Domain Name:
   - 1.3 Internet facing IP address:
   - 1.4 Internet facing netmask:
   - 1.5 Internet facing domain name server address(es):
   - 1.6 Internet facing broadcast address:
   - 1.7 Internet facing gateway address:
   - 1.8 Internal IP address:
   - 1.9 Internal netmask:
   - 1.10 Internal CIDR block address:
   - 1.11 Server's Internet facing IP address:
2. Hardware required
   - 2.1 OPSAID Field Unit
   - 2.2 USB CDROM
3. Booting from CD
   - 3.1 Download and burn Debian 4.0 disc 1 from debian.org.
   - 3.2 Insert CD into a USB CDROM connected to the server.
   - 3.3 During BIOS checks, press <delete>.
   - 3.4 Select 'Advanced BIOS Features'.
   - 3.5 Select 'USB-CDROM for First Boot Device'.
   - 3.6 Select 'Hard Disk for Second Boot Device'.
   - 3.7 Press <esc> then choose 'Save & Exit Setup'.
4. Debian install. The install process is performed in a menu-driven environment. Most of the items are self-explanatory or are described in the install.
   - 4.1 Press enter to begin the install.
   - 4.2 Select a language.
   - 4.3 Select an area.
   - 4.4 Select a keyboard layout.
   - 4.5 Configure network
     - 4.5.1 Select 'Go Back' if you were not prompted to configure your network.
     - 4.5.2 Select 'Configure network manually'.
     - 4.5.3 Set the Internet facing IP address.
     - 4.5.4 Set the Internet facing netmask address.
     - 4.5.5 Set the Internet facing gateway address.
     - 4.5.6 Set the domain name server address(es).
   - 4.6 Set the host name.
   - 4.7 Set the domain name.
   - 4.8 Partition disks
     - 4.8.1 Select 'Guided – use entire disk'.
     - 4.8.2 Select 'IDE1 Master'.
     - 4.8.3 Select 'All files in one partition'.
     - 4.8.4 Select 'Finish'
     - 4.8.5 Select 'Yes'

4.9 Select the appropriate time zone.
4.10 Set the root (administrator) password.
4.11 Add a user and set the password for the user.
4.12 Configure the package manager
  4.12.1 Select 'Yes' for network mirror.
  4.12.2 Select an appropriate network mirror.
  4.12.3 Set a proxy if applicable.
  4.12.4 Select 'No' for the package survey.
4.13 Software selection
  4.13.1 'Standard system' should already be selected. Press <tab> to select 'Continue' then press <enter>.
4.14 Boot loader
  4.14.1 Select 'Yes'.
4.15 Finished
  4.15.1 Select 'Continue'. This will reboot the computer.

5. Remove excess packages
   5.1 Log in as root.
   5.2 apt-get remove exim4-base nfs-common sysklogd portmap pidentd openbsd-inetd
   5.3 Type yes

6. Package install
   6.1 apt-get install build-essential binutils gcc g++ libc6-dev libgmp3-dev syslog-ng openssh-server openssl tcpdump bzip2
   6.2 Type yes
   6.3 wget http://www.sandia.gov/scada/opsaid/ref-1.0/config.tar.bz2
   6.4 tar –xvjf config.tar.bz2 –C /
   6.5 rm config.tar.bz2

7. Configure secondary network interface
   7.1 vi /etc/network/interfaces
   Change the line: auto lo
   To: auto lo eth0 eth1
   Add the following to the bottom of the file:
   iface eth1 inet static
   address [Internal IP]
   netmask [Internal netmask]
   network [Internal CIDR block address before the forward slash]
   broadcast [Internal broadcast address]
   Save the file and exit.

8. Install and configure snort
   8.1 apt-get install snort
   8.2 Type yes
   8.3 You will be prompted to enter your home network. This is the internal network CIDR block address.
   8.4 vi /etc/snort/snort.debian.conf

Change the line: DEBIAN_SNORT_OPTIONS=“”
To: DEBIAN_SNORT_OPTIONS=“-s”
Save the file and exit.

9. Set up syslog-ng
   9.1 cp /root/config/client/syslog-ng.conf /etc/syslog-ng/syslog-ng.conf
   9.2 vi /etc/syslog-ng/syslog-ng.conf
   Insert the server’s IP address on the line: tcp([Server IP] port(500));
   Save the file and exit.

10. Set up IP forwarding
    10.1 vi /etc/sysctl.conf
    Change the line: #net.ipv4.conf.default.forwarding=1
    To: net.ipv4.conf.default.forwarding=1
    Save the file and exit.

11. Set up firewall. The firewall provided is very simple and should be modified to fit your network and needs.
    11.1 /bin/sh /root/config/firewall.sh
    11.2 iptables-save >/etc/firewall.conf
    11.3 vi /etc/network/if-up.d/firewall
    Add the following lines:
    #!/bin/sh
    iptables-restore < /etc/firewall.conf
    Save the file and exit.
    11.4 chmod +x /etc/network/if-up.d/iptables

12. Set up Host-Based Intrusion Detection system. The default settings should be adequate for most. If your configuration requires an alternate setup, notes are provided in the scan.conf file.
    12.1 tar –xvzf /root/config/hid.tgz –C /
    12.2 vi /etc/scan.conf
    12.3 Save the file and exit.

13. Install and set up StrongSwan
    13.1 Install StrongSwan
    13.1.1 wget http://download.strongswan.org/strongswan-4.1.2.tar.bz2
    13.1.2 tar –xvjf strongswan-4.1.2.tar.bz2
    13.1.3 cd strongswan-4.1.2
    13.1.4 ./configure --prefix=/usr --sysconfdir=/etc
    13.1.5 make
    13.1.6 make install
    13.1.7 cd ../
    13.1.8 rm –fr strongswan*
    13.1.9 cp /root/config/openssl.cnf /etc/ssl/

    13.2 Create Client Certificate

13.2.1 scp root@[Server's Internet IP]:/etc/ssl/client_key.pem /etc/ipsec.d/private/

13.2.2 scp root@[Server's Internet IP]:/etc/ssl/client_cert.pem /etc/ipsec.d/certs/

13.2.3 scp root@[Server's Internet IP]:/etc/ssl/demoCA/cacert.pem /etc/ipsec.d/cacerts/

13.3 Set up StrongSwan

13.3.1 cp /root/config/client/ipsec.secrets /etc/

13.3.2 vi /etc/ipsec.secrets

Insert your client private key password in the "[password]" field.
Save the file and exit.

13.3.3 rm /etc/ipsec.conf

13.3.4 vi /etc/ipsec.conf

Add the following lines:

```
version 2.0
conn net-net
    left=[Internet IP]
    leftcert=client_cert.pem
    leftsourceip=[Internal IP]
    leftsubnet=[Client's Internal CIDR block address]
    right=[Server's Internet IP]
    rightsubnet=[Server's Internal CIDR block address]
    rightid="C=[ClientCountry],
        ST=[ClientState],
        L=[ClientLocation],
        O=[ClientOrganization],
        OU=[ClientOU],
        CN=[ClientCommonName],
        E=[ClientEmail]"
    compress=yes
    auto=start
```

Save the file and exit.

13.3.5 cp /root/config/ipsec /etc/init.d/

13.3.6 chmod 755 /etc/init.d/ipsec

13.3.7 ln -sf /etc/init.d/ipsec /etc/rc2.d/S99ipsec

14. Set up SSH

14.1 cp /root/config/sshd_config /etc/ssh

14.2 su [user created during setup]

14.3 ssh-keygen –t dsa

14.3.1 To save the key in the default location press <enter>.

14.3.2 Enter passphrase.

14.3.3 Repeat passphrase.

14.4 scp ~/.ssh/id_dsa.pub [Server's User]@[Server's Internet IP]:~/.ssh/authorized_keys

14.5 scp [Server's User]@[Server's Internet IP]:~/.ssh/id_dsa.pub ~/.ssh/authorized_keys
14.6 Type exit
14.7 When you ssh in to the server, do the following:
    14.7.1 Log in to the client as [Client User]
    14.7.2 ssh [Server User]@[Server Internet IP]
    14.7.3 If you need root access on the server, type su -
14.8 When you ssh from the server to the client, do the following:
    14.8.1 Log in to the server as [Server User]
    14.8.2 ssh [Client User]@[Client Internet IP]
    14.8.3 If you need root access on the client, type su -

15. Reboot the system
    15.1 Type reboot

Appendix E: Entergy Testing Procedure

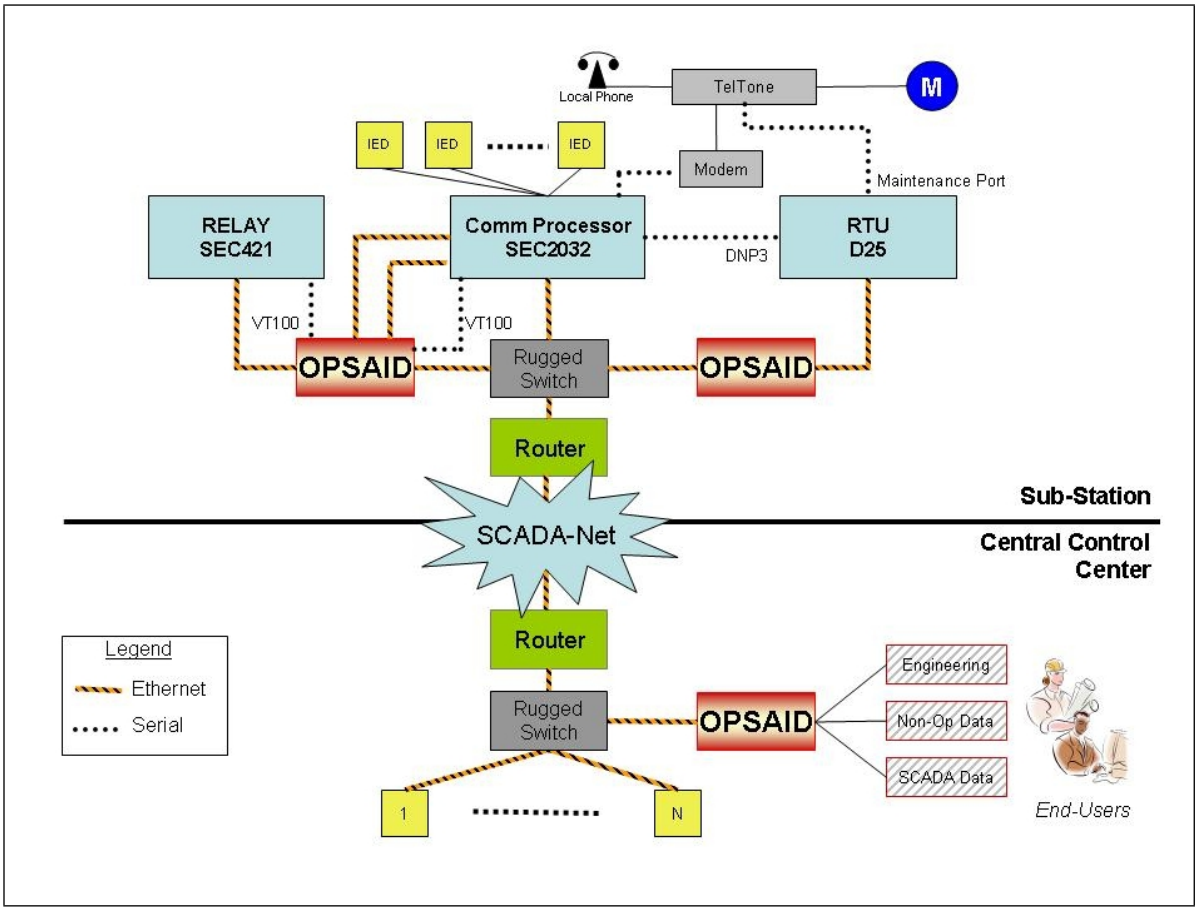

Figure E.1. OPSAID Field Trial Schematic

Joint Entergy/Sandia SLAP (OPSAID) Test Procedure

1. Test Objectives
   a. Lab test (in Entergy laboratory):
      i. The objectives of lab testing are to:
         - Allow Entergy technical people to get familiar with OPSAID operation
         - Identify and fix any issues that would prevent a successful field test.
         - Identify any problems to be fixed in future refinements to OPSAID and generate data to base future refinements on.
      ii. Type of testing to be performed:
         - Prove out OPSAID network in normal anticipated operation.
         - Stress test software
         - Cryptographic testing as applicable/feasible
         - Test interoperability, per Entergy procedures, with different routers, multiplexers, etc
         - Standards conformance/compatibility (as applicable/feasible)
   b. Field test
      i. The objectives of the field test are to:

- Simulate real-time administration and communication required by OPSAID administration / Entergy work demands
- Generate data useful to project partners, DOE, and industry.

2. Equipment Under Test
   a. Hardware:
      i. 2 Rackmount "Substation" OPSAID devices
         - 800/1000 MHz VIA ITX system
         - 1 GB Ram
         - 1 GB Flash hard drive
         - 2x 10/100/1000 Network Ports
         - 4x 10/100 Network Ports
         - 4x Serial Ports
      ii. 1 Rackmount "Control Center" OPSAID
         - 1.86 GHz Dual Core Processor
         - 1 GB Ram
         - 250 GB Hard Drive
         - 3x 10/100/1000 Network Ports
         - 1x Serial Ports
   b. Software:
      i. Ubuntu Linux Server (Debian-Based)
      ii. OPSAID Visualization tool
      iii. mySQL Backend
3. Features Under Test
   a. Three facets of OPSAID operation will be tested:
      i. OPSAID administration functions
         - Remote login, key exchange with SSH
         - SQL server logging
         - Visualization and monitoring
   b. Monitoring and control data exchange with DNP over IP using IPSec
      i. Without IPSEC
      ii. With IPSEC
      iii. Mixed-mode communication
   c. Administration functions while exchanging monitoring and control data
4. Test Networks – The diagrams below illustrate the four tested field test setups.

a. Test Network 1 (Figure E.2)

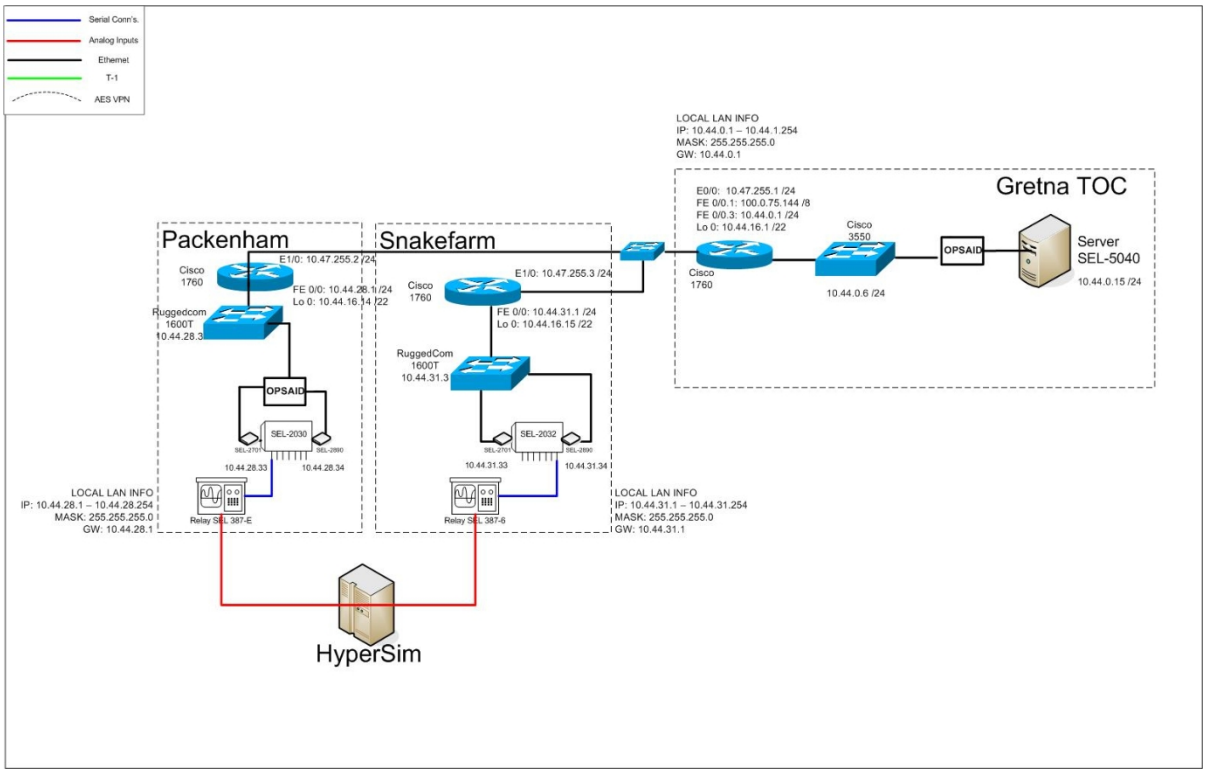

Figure E.2. Test Network 1

The system in Test Network 1 was used for initial OPSAID feature testing and application integration testing. It adds specificity Field Test Trial schematic. The SCADA applications under test using this system include D25 polling of various OPSAID-connected IEDs (the SEL-2030, RTU, and 387-E), as well as remote administrative connections to these field IEDs. The network allows for comparison of the basic system with OPSAID insertion effects through its use of a second communications leg, with the SEL-2032 and 387-6 connections. This network provides a baseline for the stress-testing scenarios.

b. Test Network 2 (Figure E.3)

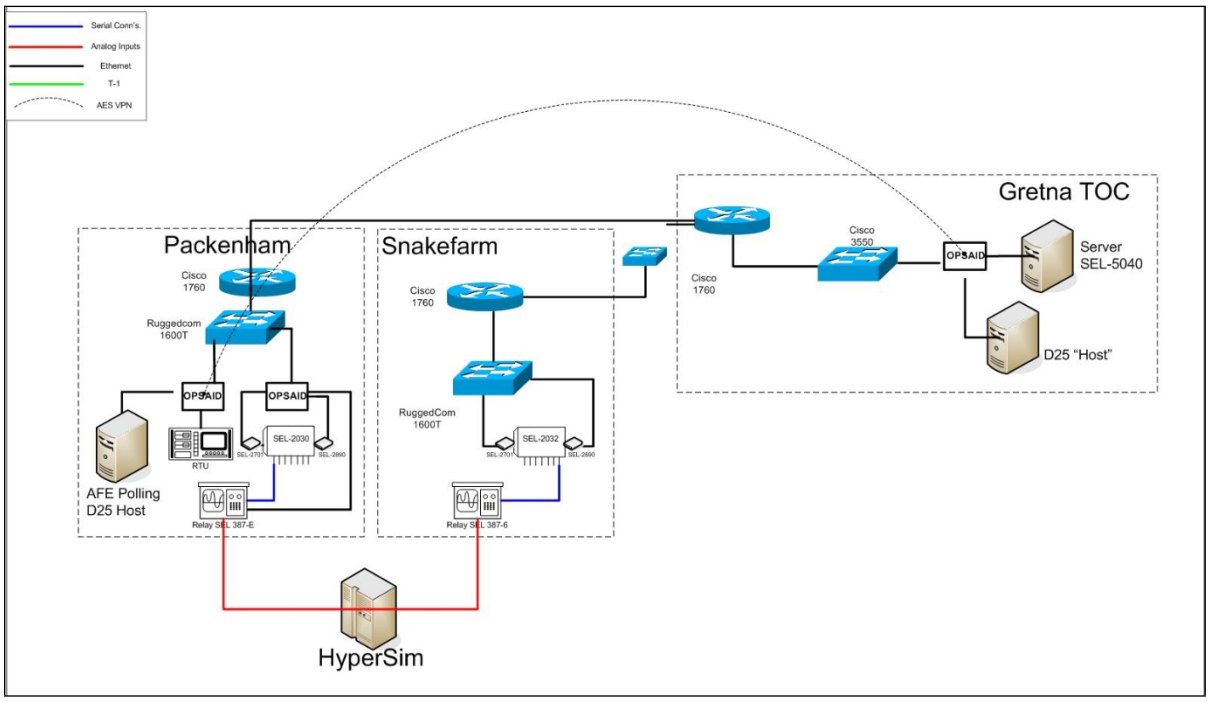

Figure E.3. Test Network 2

Test Network 2 enables further application integration testing based on the addition of the fractional T1 communication link, which is a typical element of Entergy automation communication networks. It is also used for the stress testing profile, in order to understand the effects of the reduced T1 bandwidth on OPSAID performance.

c. Test Network 3 (Figure E.4)

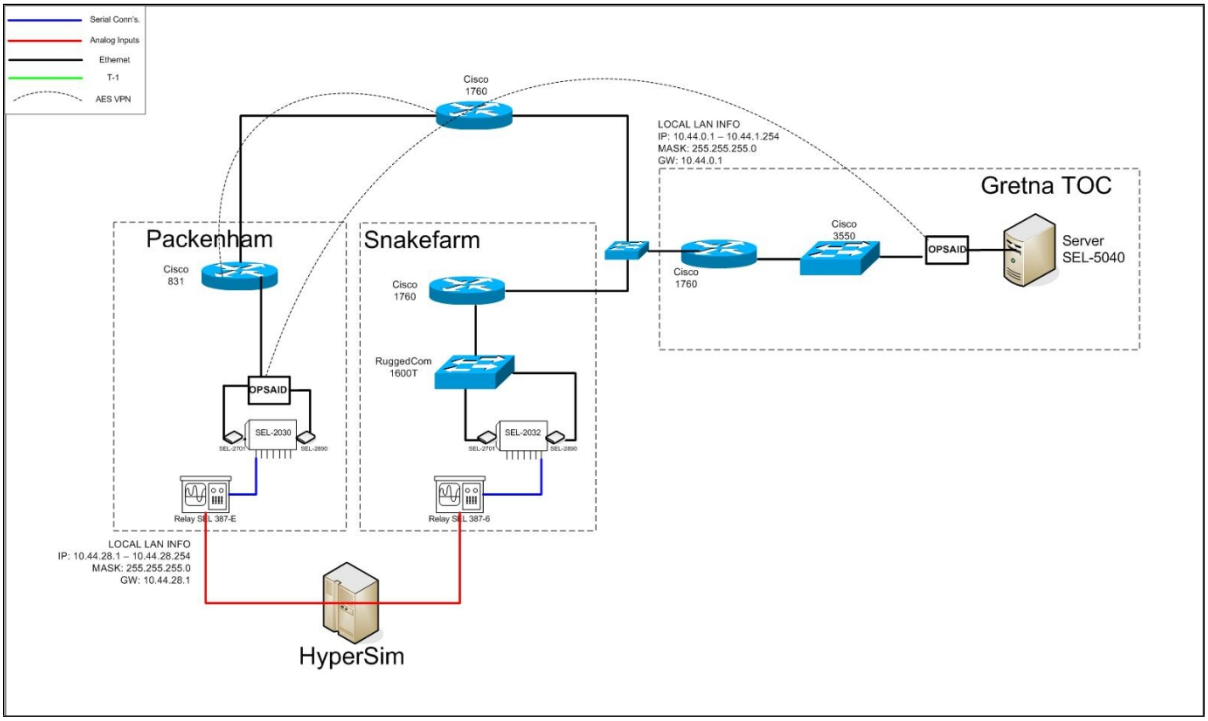

Figure E.4. Test Network 3

Test Network 3 adds a VPN connection, modeling automation connectivity through non-utility-owned assets (e.g. leased Telco lines). This serves as a feature test, given its use in actual automation networks, and enables testing the ability of the OPSAID connection to operate as a VPN-within-a-VPN. Performance impacts are also considered.

d. Test Network 4 (Figure E.5)

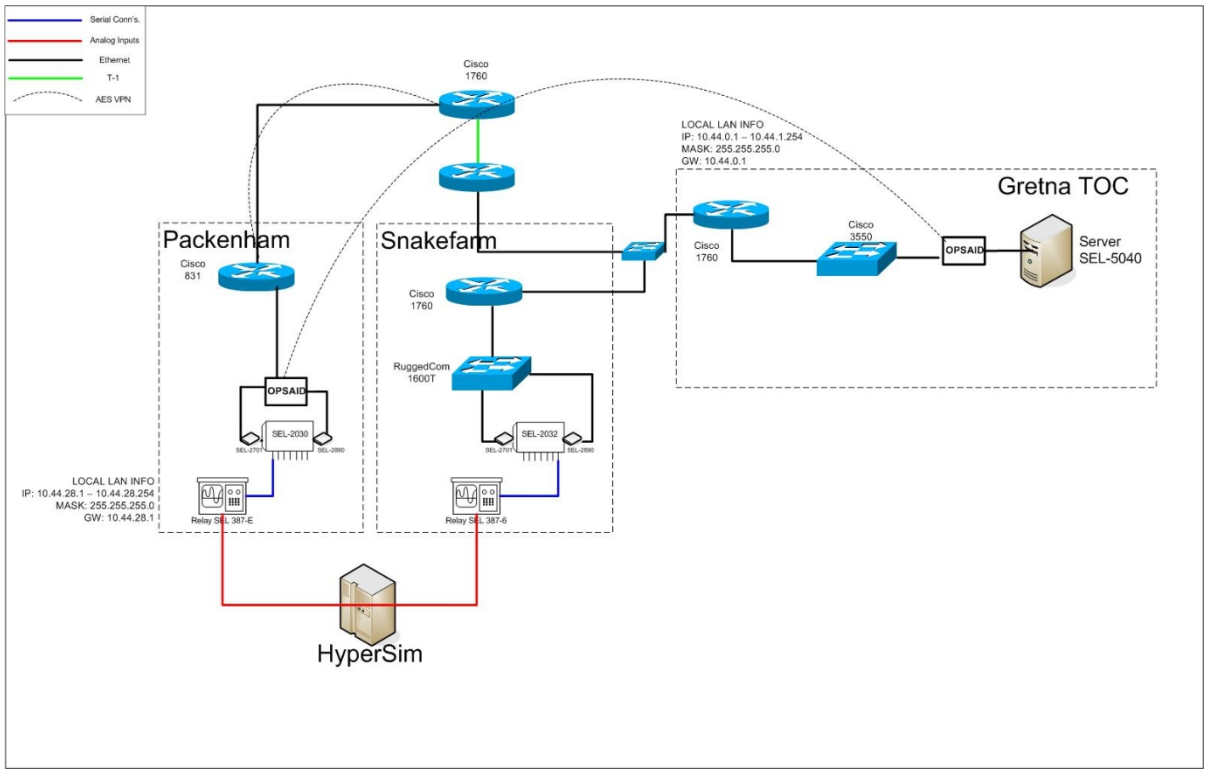

Figure E.5. Test Network 4

This system returns to the Ethernet connection, but adds a meshed topology to the automation WAN. This enables testing failover performance for OPSAID connections during rerouting, when communication paths are severed.

5. High Level Test Procedure Summary for each test type per section 1 of this outline, *Test Objectives*.
   a. ***Test Scenario 1 – OPSAID Feature Testing***
      i. Remote login, key exchange with SSH for OPSAID administration
      ii. VPN IPSec Setup
      iii. Firewall Operation
      iv. SSH Console Operation
      v. Logging Facilities (write to OPSAID logging server using syslog-ng and IPSec)
      vi. Tunneling IP Traffic using VPN
      vii. Visualization and monitoring
   b. ***Test Scenario 2 – Application Integration Testing***
      i. Test configuration session functionality to end devices
      ii. Test OPSAID compatibility in original utility communications systems and applications
   c. ***Test Scenario 3 – Stress and Performance Testing***
      i. Perform high and low volume data transfers between the OPSAID devices to analyze the following network features:
         - Address translation,
         - Packet delay, loss, or timing (jitter), for end-to-end SCADA applications
         - Application response time (with and without OPSAID),
         - Loss of communications, including intermittent connectivity and rerouting
         - Reliability, and
         - Throughput.
   d. ***Test Scenario 4 – Standards Conformance / Compatibility (as applicable/feasible)***
      i. Consideration of applicable standards (involves no actual lab or field testing):
         - IEEE 399, Recommended Practice for Power Systems Analysis [1]
         - IEEE C37.1/D1.5, Draft Standard for SCADA and Automation Systems [2]
         - IEEE 1646-2004, Standard Communication Delivery Time Performance [3]
         - IEEE P1615/D6, Draft Recommended Practice for Network Communication in Substations [4]
         - IEEE C37.115, 2003, *Evaluation of Message Communications* [5]
      ii. At the time of the publication of this test plan, Sandia personnel did not have a copy of IEEE 399, so its relevance was not evaluated with respect to this field test. However, once obtained, it can be examined similarly to the other four.
      iii. IEEE C37.1/D1.5[3] In section 5, tables 1-3 define important temporal parameters for automation systems. They include minimum performance requirements for measurement and algorithmic latency, to which integrated control systems must conform. The introduction of an OPSAID device overlay for security must not interfere with this standard conformance; therefore, an important element of the device testing should be to understand the typical latency introduced by OPSAID devices (presumably under the most stressful case).

[3] "Section" and "table" remarks in the discussion of IEEE C37.1/D1.5 refer to elements of that document, not of this report.

iv. Unfortunately, this standard places security beyond its scope (referencing section 5.3), so it adds little guidance on conformance testing for the prototype OPSAID. In the long run, the material in sections 5.6 (regarding availability) and 7 (environmental requirements) will be important to vendors developing branded OPSAID solutions. Possibly subsequent field tests could evaluate potential technologies for OPSAID availability. The team will work to ensure electrical source compatibility summarized in sections 6.2 (Grounding) and 6.3 (Electrical Power).

v. IEEE P1615/D6 contains several sections relevant to the test exercise. For example, it may be useful to consider the security connections of the OPSAID test system in conjunction with the data category definitions in section 6.1. In section 6.2, to document delves into security recommendations concerning threats to automation systems. The OPSAID design and application should be mapped to the relevant security issues in this section, and this section lends itself to attack vectors that can be applied to the SLAP-secured test system. Production OPSAID components should conform to the Recommendation in section 7.3.1 (concerning electrical noise parameters), and the recommended practices in Section 10 describing LAN requirements for the substation environment (in the latter case, the tested design should support as many of the requirements as is feasible).

vi. In IEEE Standard 1646, the issue of temporal performance requirements in automation systems is amplified, in this case particularly for electric power substations. Figure E.1 depicts the role of an OPSAID as a "Comm Processor". Tables 1 through 3 provide specific maximum delivery times, which tend to indicate a range of allowable processing delays that can be introduced by an OPSAID security overlay (depending on the application, i.e. tripping, load shedding, etc.). In the envisioned application (SCADA), this requirement amounts to "up to 200 ms." Section 5 adds to other sections from previous standards concerning communication capabilities for on-time data delivery.

vii. Finally, IEEE C37.115-2003 specifies, after a fair bit of introductory material for networking in sections 4 and 5, requirements for testing communications for substation devices. The measurement application shown in Figure E.5 may serve as a potential configuration for monitoring OPSAID tests, with the role of "Communication Network Analyzer" filled by a sniffer. As described in sections 6.1 to 6.2, the OPSAID field test should evaluate performance relating to:

1. Address translation,
2. Encapsulation and decapsulation,
3. Packet delay, loss, or timing (jitter),
4. Application response time (with and without OPSAID),
5. Application feature testing (to ensure OPSAID does not preclude functionality),
6. Reliability, and
7. Throughput.

Appendix F: Entergy Testing Results

1. Summary

During the week of Monday, October 16, a Sandia/Entergy Lab Test was performed on the OPSAID reference implementation in Baton Rouge, LA at the PCS 2000 facility. The purpose of the lab test was to perform the initial tests lined out the in the *Joint Entergy/Sandia SLAP (OPSAID) Test Procedure* described in *Appendix E: Entergy Testing Procedure* of this report and to give Entergy technical experts a basic understanding of the configuration and use of the OPSAID. The tests that were performed showed that the OPSAIDs worked correctly in the network of IEDs, specifically providi
ng encryption and authentication. Future lab tests will enable us to finish the testing needed to prove the complete functionality of the OPSAID.

1.1. Goals

Test the OPSAID in a lab configuration of IEDs and other related PCS equipment following the *Joint Entergy/Sandia SLAP (OPSAID) Test Procedure* (*Appendix E: Entergy Testing Procedure*).

Familiarize Entergy personnel with the operation and configuration of the OPSAID for further testing.

1.2. Participants

Sandia National Laboratories:	Alex Berry Adrian Chavez
Entergy:	Leonard Chamberlin Mark Allen

2. Background

The OPSAID system provides automation networks and hosts with many of the security capabilities found in traditional IT elements. In contrast to some security solutions, the OPSAID system is based entirely on open-source software and standardized hardware. Furthermore, each OPSAID device is designed to protect either single hosts or small network segments. The OPSAID system also provides logging to a centrally located hardened security historian server for forensic analysis and control systems visualization.

3. Test Timeline

Monday 10/16/2006

- Morning
  - Configure OPSAIDs for the PCS Laboratory/Leonard's Configurations.
  - Test basic OPSAID functionality
  - Verify operation of IEDs without OPSAID Protects
- Afternoon –
  - Testing Network Configurations 1-4

Tuesday 10/17/2006

- Morning
  - Further Configuration of OPSAIDs and IEDs for testing PCS traffic.
  - Performance testing of the network under load for configurations 1-4
- Afternoon
  - Discussions on results, generating documentation for configuration use

Wednesday 10/18/2006

- Morning
  - Walk Entergy personnel (Leonard) through the configuration and operation of the OPSAID devices

4. Hardware

Rackmount "Substation" OPSAIDs

- 800/1000 MHz VIA ITX System
- 1 GB RAM
- 1 GB Flash Hard Drive
- 2x 10/100/1000 Network Ports
- 4x Serial Ports

Rackmount "Control Center" OPSAID

- 1.86 GHz Dual Core Processor
- 1 GB RAM
- 250 GB Hard Drive
- 3x 10/100/1000 Network Ports
- 1x Serial Ports

One of the "Substation" OPSAIDs was dead on arrival and we are working to find out what happened to it during travel. Since this is a reference implementation the hardware failure was not a critical failure.

5. Software

Ubuntu Linux Server (Debian Based)
OPSAID Visualization Tool
MySQL Backend

6. Entergy Lab Tasks

6.1. Completed Tasks

6.1.1. Remote Login

Each of the OPSAIDs have been configured with open SSH ports for remote configurations. Two factor public key authentication is required to log into the SSH connections. Start and stop time, username, and auth type are all data points recorded for each incoming SSH session and sent to the central system log. This shell is provided for complete configuration of the Linux subsystem and worked without error.

6.1.2. Remote Terminal Access

SSH was also configured to perform remote terminal access IEDs using a SSH session with a minicom (terminal program) shell. This allows a user to SSH to the OPSAID and connect to the configuration port on the IED. The remote terminal required the following:

- two-factor authentication to log in
- restricted user access
- no applications beyond minicom could run

All of the remote terminal access sessions we configured for the SEL-2032 and SEL-2890 worked correctly.

6.1.3. MySQL Server Logging

Each of the OPSAIDs has been configured to run the syslog-ng service on startup. The syslog-ng service reports and generates an audit trail of events to the OPSAID master in the network. The events generated have different severities associated with them along with a descriptive message about each of the events. The OPSAID master is configured as a centralized system in the network and has been setup to receive syslog-ng messages from all other OPSAIDs in the network. The OPSAID master has the most computing power and the largest amount of disk space in order to support the visualizations and the MySQL server database, which stores all events received. The MySQL database on the OPSAID master is comprised of a single table storing all of the events received from all of the OPSAIDs within the network. The events received by the OPSAID master are stored in the raw format in which they are received from the syslog-ng service. All logging over the network was performed within the VPN tunnel.

In order to normalize the raw database, a second database with a new schema is also available. The advantages of the second database are that it is more organized, it has a more flexible table structure, and it also nicely drives one of the visualizations talked about in the section 4.1.4. *Visualization and Monitoring*. The formatted database is populated by exporting and parsing the raw syslog-ng database into a new schema. The overall process of both databases being populated is summarized in Figure F.1 below. Each OPSAID in the network generates syslog-ng events and sends them to a raw database on the OPSAID master, which can be viewed in a browser. The raw database is then exported into a formatted database that drives the Java application displaying the network view visualization.

Each of the OPSAIDs successfully sent syslog-ng events to the OPSAID master machine and the OPSAID master successfully received the syslog-ng events. Once the events were received by the OPSAID master, the events were successfully populated into both MySQL databases to aid both visualizations discussed in the next section.

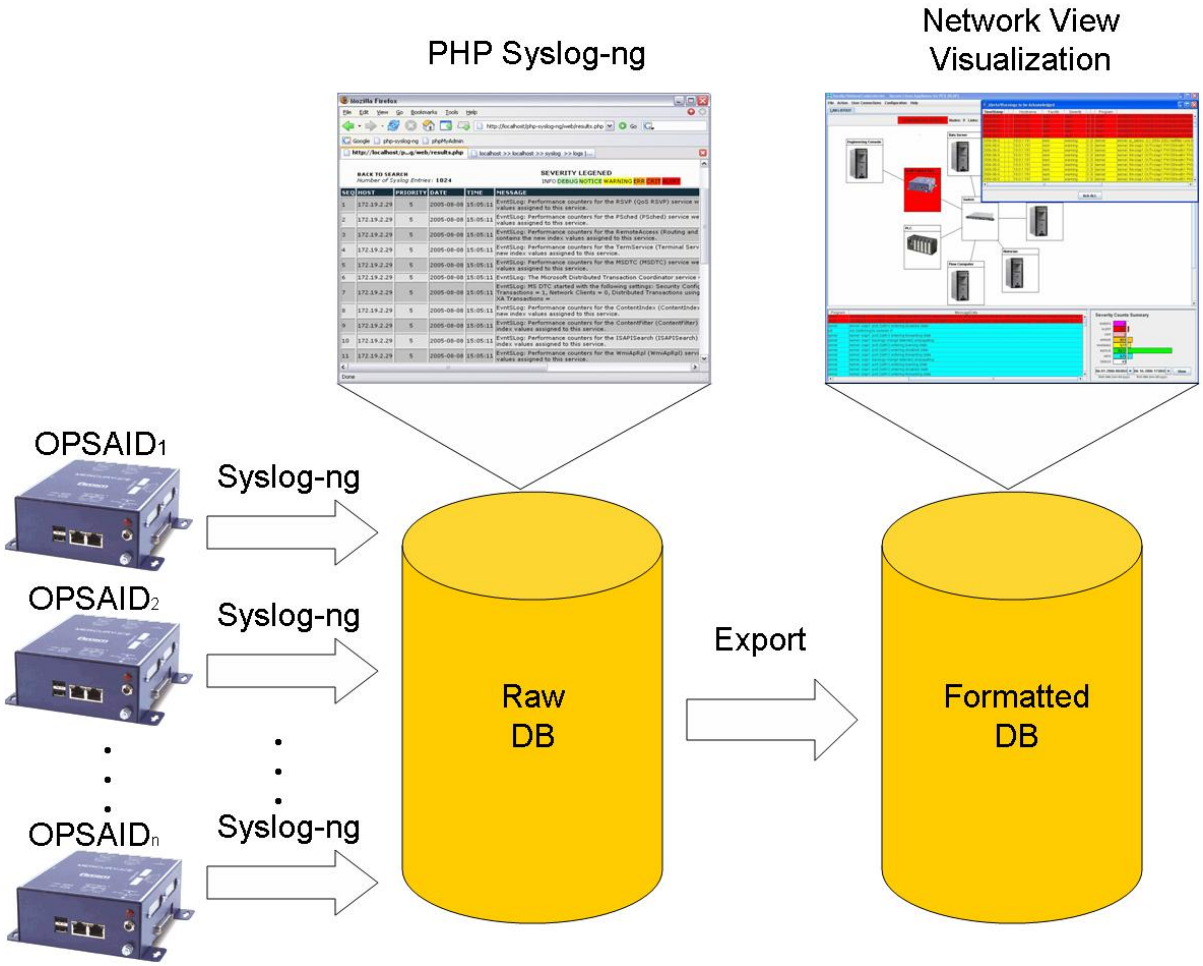

Figure F.1: The overall process of the visualizations

6.1.4. Visualization and Monitoring

The OPSAID master is configured to run a web server so users can display the raw syslog-ng events in a tabular format via a PHP web page. The data displayed in the PHP web page can be filtered and searched based on any criteria specified by the user (severity level, hostname reporting event, date/time, service, etc). The PHP web page is useful for submitting queries to the database and viewing the results in a table format.

Alternatively, a graphical view of the network can be viewed via a Java application. The database containing the raw syslog-ng events is first exported into a structured database with a much more flexible design in the table structures. New events inserted into the raw database are continuously exported to the formatted database in order to synchronize the two databases. The formatted database is then continuously queried by the Java application for new events to update the displayed table at the bottom of the application. The table entries are colored based on the severities of the events. The bottom right corner of the application also displays a bar chart of the event counts based on severity levels. The application will distinguish events at the severity level of ‘alert’, the highest severity level generated by syslog-ng, in a noticeable way. When an alert is generated by an OPSAID in the network, the icon of the OPSAID reporting the alert blinks red in the network view of the application and continue to blink red until an operator acknowledges and fixes the flagged alerts.

The overall process of both visualizations is summarized in Figure F.1 above. The OPSAIDs first send raw syslog-ng events to a MySQL database on the OPSAID master. The raw database can then be viewed and queried via a web browser as shown in the figure. The raw database is then exported to a separate MySQL database with a more organized and formatted table structure. The Network View visualization tool then queries the formatted database, and alerts are visualized by OPSAIDs icons blinking red. The alerts can then be acknowledged and corrected by an OPSAID administrator. Figure F.2 shows the second visualization tool and Figure F.3 shows the user interface of the Java Application. The Java application is started and the most current events are displayed at the bottom of the application along with a network view at the top.

Both visualizations were successfully tested and worked as expected. Both the PHP web page and the network visualization tool were updated for each syslog-ng event generated by each of the OPSAIDs. The visualizations can assist an operator in verifying and maintain an audit log of notable events occurring on each of the OPSAIDs in the network.

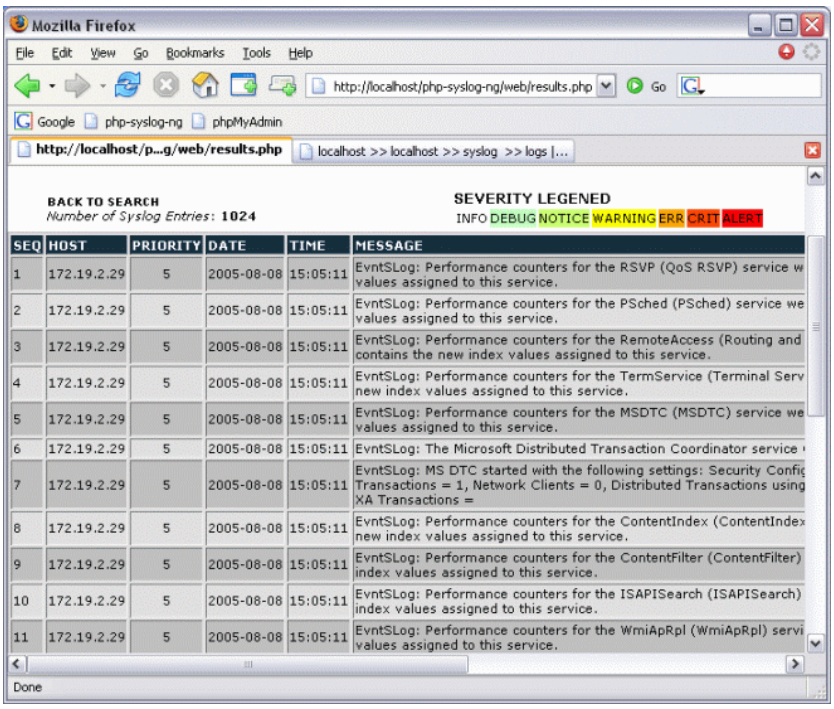

Figure F.2: A sample table from the raw database displayed in a PHP web page in a web browser

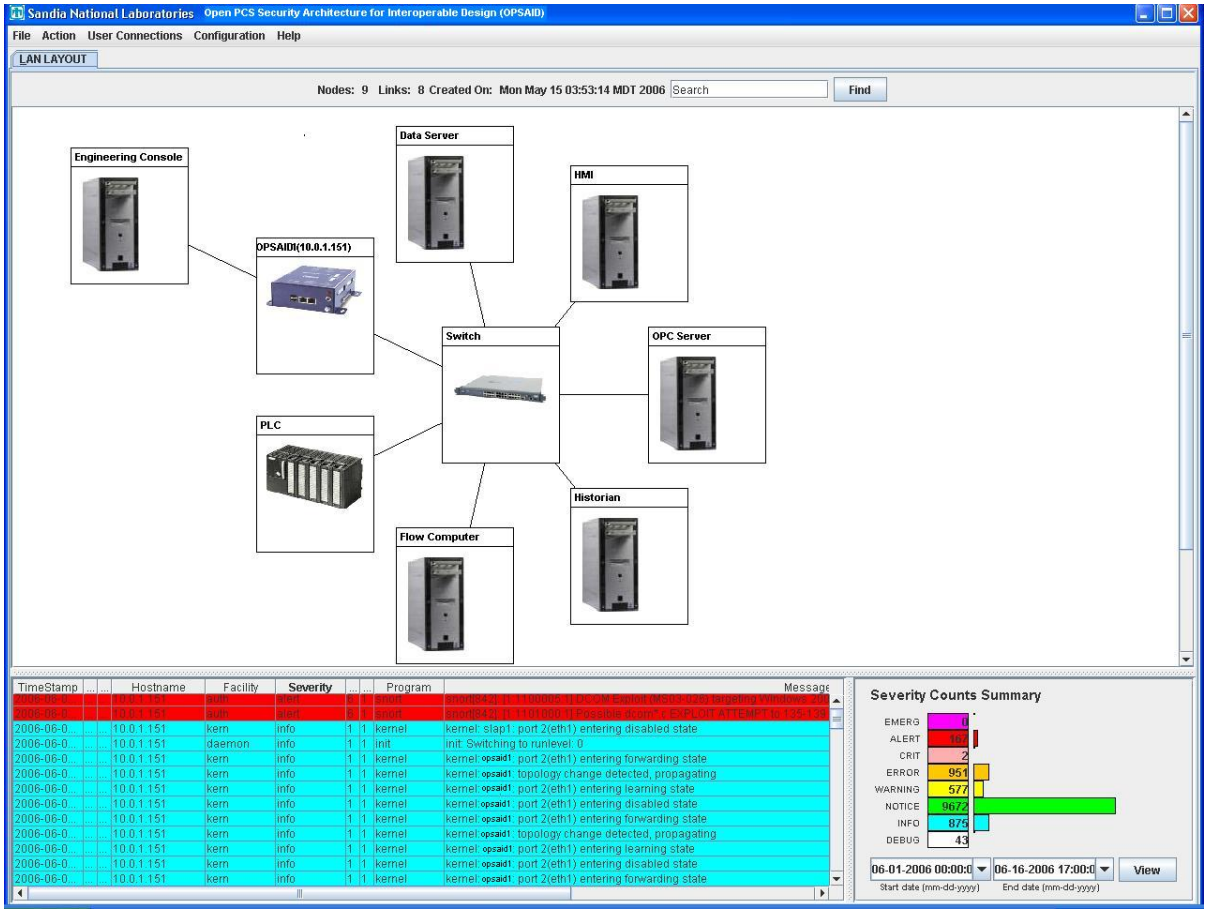

Figure F.3: The user interface to the Java Application

6.1.5. Prove out OPSAID network in normal anticipated operation

For reference, the standard network configuration is shown below in Figure F.4. The standard configuration enables further application integration testing, based on the addition of the fractional T1 communication link. This is a typical element of Entergy automation communication networks. It is also be used for the stress testing profile, in order to understand the effects of the reduced T1 bandwidth on SLAP performance.

Everything worked as expected in the standard network configurations. The OPSAIDs did not introduce any errors into the network and therefore worked as expected. Once the network was properly configured with the correct IP addresses and subnets, the OPSAIDs worked transparently to the network. This proved that the OPSAIDs are capable of operating in the normal circumstances of a typical network configuration for a PCS system.

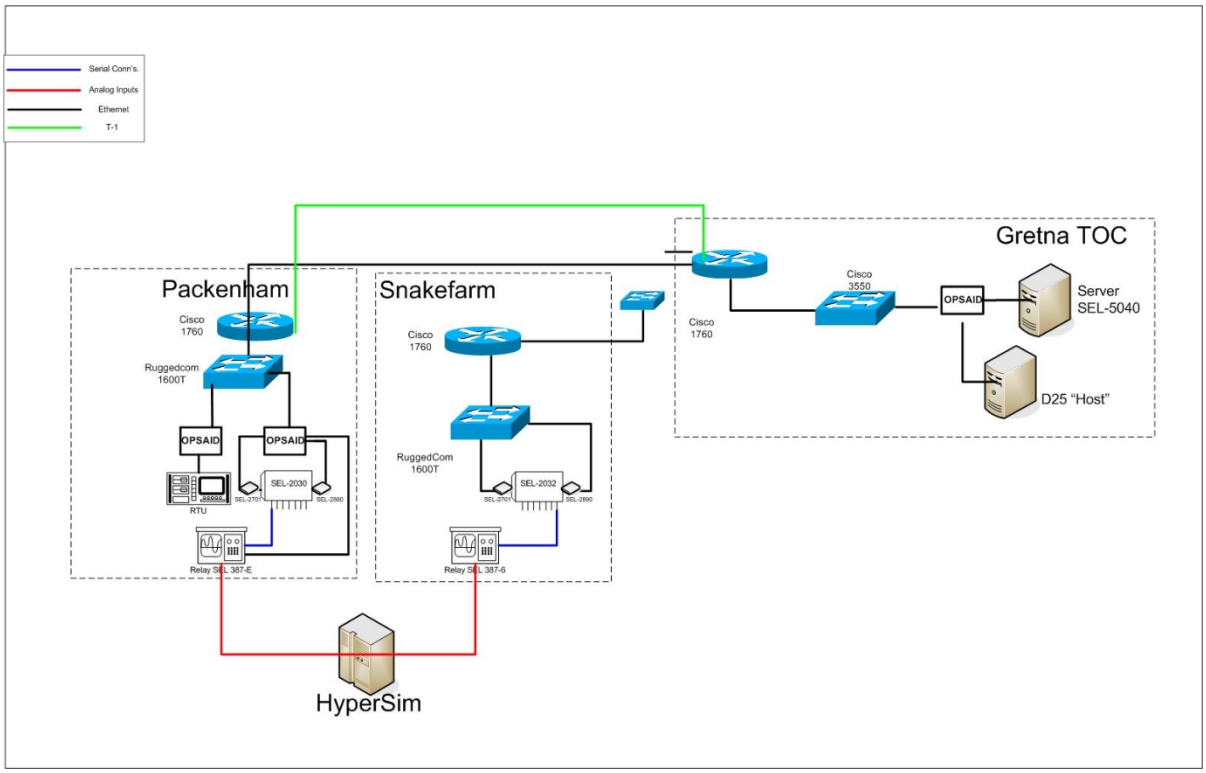

Figure F.4: The Test Network in Standard Network Configuration

6.2. Performance Test

IPerf is an open source network performance tool designed to measure various network performance values such as the maximum TCP/UDP bandwidths, MSS/MTU size, jitter, and packet loss. The bandwidth measurement results are shown in Figure F.5 below. The averages for the different network scenarios are shown in the legend at the bottom. The captured data originated from a TCP connection established using the IPerf tool.

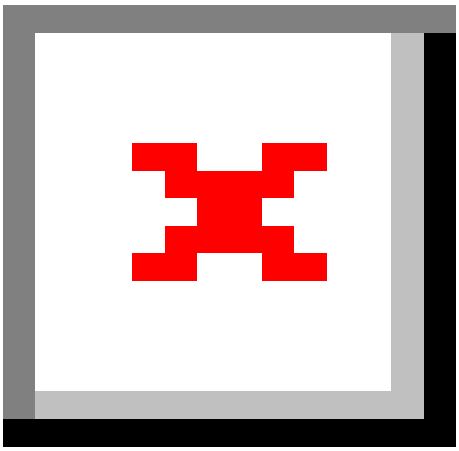

Figure F.5: Bandwidth data points taken for a one-minute stretch

Figure F.6 shows the same data in a different format. Each test is shown linearly by time for a different view of the data. Each bar shows a different segment of the test as follows:

1. Unsecured 10MBit network with standard packet overhead.
2. VPN over 10MBit network
3. VPN in failover T1 network
4. VPN over 10MBit network
5. VPN over VPN in 10MBit network, the intermediate VPN router induced the change in network utilization
6. VPN over VPN in failover to T1
7. VPN over VPN on 10MBit network after recovery
8. VPN over 10MBit network

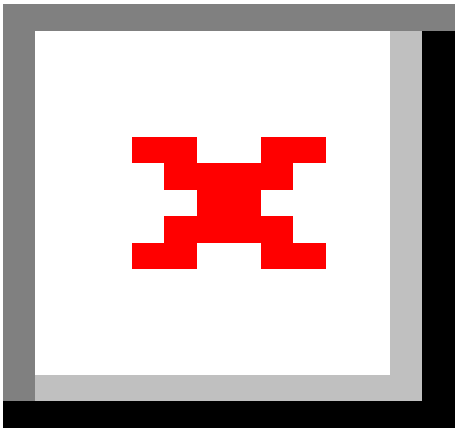

Figure F.6: Network utilization in sequential network tests.

6.2.1. Cryptographic Testing

The IPSec tunnel worked as expected in each of the network test scenarios. Racoon was used for the IPSec configuration along with x.509 certificates for authentication. Pre-shared keys were also tested and also worked as expected. 128bit AES was used for all of the VPN tunnels. Since we used a known tested AES implementation, we did not test the correctness of the algorithm itself.

6.2.2. Interoperability Testing

Testing was performed using available equipment and personnel. Three types of traffic were tested in the network: SCADA non-operational data, engineering configuration data and IT traffic. Network was completely operational for all of the traffic types. The network failover took about thirty seconds, once the new routes were established, the VPN resumed operation.

6.2.3. Network Diagrams

All four of the network diagrams outlined in Appendix E: *Entergy Testing Procedure* were tested. The OPSAID correctly functioned in all of these configurations without affecting the communication between IEDs and between IEDs and hosts on the process control network.

7. Remaining Tasks

- Stress Test with HYPERSIM
- D25 Polling
    - DNP over IP
    - IPSec Disabled
    - IPSec Enabled
    - Mixed Mode

8. Future Tasks

- Traffic Playback
- Snort
    - Sanity Checking of packets
- Network Metrics
    - Packet Delay
    - Packet Loss
    - Jitter
    - Reliability
    - Application Response Time
- Firewall Policies (IPTables)

9. ENTERGY TEST Setup & Configurations

IPSec Tunnel Configuration Using Racoon

The easiest way to understand the Racoon file is by looking at references [9]-[14] above. They are short and very descriptive about how to configure Racoon.

Starting an IPSec tunnel

The IPSec tunnel is configured to start on bootup, but it can also be started manually if desired. To manually start the IPSec tunnel, enter the following command on both tunnel gateways:

```
/etc/init.d/racoon start
```

This will automatically load the policy to allow the IPSec tunnel to be created and will negotiate a shared key to be used for the tunnel using Racoon.

If you would like to load the policy and start the Racoon service separately, enter the following two commands at both gateways of the tunnel:

```
setkey -f /etc/setkey_racoon_tunnel.conf
racoon -F -f /etc/racoon/racoon_cert.conf
```

These are the two commands executed in the /etc/init.d/racoon script.

Stopping an IPSec tunnel

The IPSec tunnel is currently configured to start on bootup but can be stopped as desired easily. To kill the IPSec tunnel, enter the following command on both tunnel gateways:

```
/etc/init.d/racoon stop
```

This will stop the IPSec Tunnel along with the policy used. If you want to stop the policy and the Racoon service separately, enter the following two commands:

```
setkey -F -P
killall -9 racoon
```

These are the two commands executed in the /etc/init.d/racoon script.

Files of Interest

All the files listed below appear in both the OPSAIDmaster and OPSAID1. A description of the first two files is given below along with configuration information and example configurations.

1. /etc/setkey_racoon_tunnel.conf
2. /etc/racoon/racoon_cert.conf
3. /etc/ssl/certs/newcertOPSAIDx.pem
4. /etc/ssl/certs/newkeyOPSAIDx.pem
5. /etc/ssl/certs/cacert.pem

1. /etc/setkey_racoon_tunnel.conf

Description: These files are mirror images of each other on both gateways. The only difference between them is that in/out is swapped between files.

The general syntax for this file is as follows for Gateway A:

```
# Gateway A
spdadd X.X.X.X/X Y.Y.Y.Y/Y any -P out ipsec
esp/tunnel/a.b.c.d-e.f.g.h/require
ah/tunnel/a.b.c.d-e.f.g.h/require;
spdadd Y.Y.Y.Y/Y X.X.X.X/X any -P in ipsec
esp/tunnel/e.f.g.h-a.b.c.d/require
ah/tunnel/e.f.g.h-a.b.c.d/require;
```

Similarly, Gateway B's configuration is set as follows:

```
# Gateway B
spdadd Y.Y.Y.Y/Y X.X.X.X/X any -P out ipsec
esp/tunnel/e.f.g.h-a.b.c.d/require
ah/tunnel/e.f.g.h-a.b.c.d/require;
spdadd X.X.X.X/X Y.Y.Y.Y/Y any -P in ipsec
esp/tunnel/a.b.c.d-e.f.g.h/require
ah/tunnel/a.b.c.d-e.f.g.h/require;
```

where `X.X.X.X/X` is the source range of IPs, `Y.Y.Y.Y/Y` is the destination range of IPs, `-P` is the policy, in/out is the direction of traffic, `a.b.c.d` is the source of the tunnel, and `e.f.g.h` is the destination of the tunnel. If you do not want an Authentication Header (AH), simply omit the `ah/tunnel//require` from the policy files on both OPSAIDs.

Additional information about the configuration of this file can be found by retrieving the man page of setkey.

Here are example tunnel configuration files with IP address information, as they would appear on the two gateways:

/etc/setkey_racoon_tunnel.conf (Gateway A):

```
#!/usr/sbin/setkey -f
#
# Flush SAD and SPD
flush;
spdflush;

spdadd 10.44.1.0/24 10.44.28.0/24 any -P out ipsec
esp/tunnel/192.168.0.15-10.44.28.10/require
ah/tunnel/192.168.0.15-10.44.28.10/require;
spdadd 10.44.28.0/24 10.44.1.0/24 any -P in ipsec
esp/tunnel/10.44.28.10-192.168.0.15/require
ah/tunnel/10.44.28.10-192.168.0.15/require;
```

/etc/setkey_racoon_tunnel.conf (Gateway B):

```
#!/usr/sbin/setkey -f
#
# Flush SAD and SPD
flush;
spdflush;

spdadd 10.44.28.0/24 10.44.1.0/24 any -P out ipsec
esp/tunnel/10.44.28.10-192.168.0.15/require
ah/tunnel/10.44.28.10-192.168.0.15/require;
spdadd 10.44.1.0/24 10.44.28.0/24 any -P in ipsec
esp/tunnel/192.168.0.15-10.44.28.10/require
ah/tunnel/192.168.0.15-10.44.28.10/require;
```

2. /etc/racoon/racoon_cert.conf

Description: This file describes the Racoon configuration. The remote section specifies the endpoint of the IPSec tunnel along with the certificate used to initiate the IPSec tunnel and the encryption/authentication algorithms used in the key negotiation step. The path to these certificates is specified through the path certificate statement on the first line. The sainfo section describes the source/destination networks that the IPSec tunnel will be encrypting traffic between. The encryption/authentication algorithms are also specified in the sainfo section for the IPSec tunnel. Additional information about this file can be found in the man page of racoon.conf.

/etc/racoon/racoon_cert.conf (Gateway A):

```
path certificate "/etc/ssl/certs";

remote 10.44.28.10 {
     exchange_mode main;
     certificate_type x509 "newcertOPSAID1.pem"
"newkeyOPSAID1.pem";
     verify_cert on;
     my_identifier asn1dn;
     peers_identifier asn1dn;
     proposal {
          encryption_algorithm 3des;
              hash_algorithm sha1;
                  authentication_method rsasig;
                dh_group modp1024;
         }
}

sainfo address 10.44.1.0/24 any address 10.44.28.0/24 any {
        pfs_group modp768;
        encryption_algorithm 3des;
        authentication_algorithm hmac_md5;
        compression_algorithm deflate;
}
```

/etc/racoon/racoon_cert.conf (Gateway B):

```
path certificate "/etc/ssl/certs";

remote 192.168.0.15 {
     exchange_mode main;
     certificate_type x509 "newcertOPSAID1.pem"
"newkeyOPSAID1.pem";
     verify_cert on;
     my_identifier asn1dn;
     peers_identifier asn1dn;
     proposal {
          encryption_algorithm 3des;
                hash_algorithm sha1;
                authentication_method rsasig;
                dh_group modp1024;
         }
}

sainfo address 10.44.28.0/24 any address 10.44.1.0/24 any {
        pfs_group modp768;
        encryption_algorithm 3des;
        authentication_algorithm hmac_md5;
        compression_algorithm deflate;
}
```

Editors

Nano is the editor of choice. Most things will have to be edited by root, so use "sudo nano filename" or "sudo -s" to give you a root shell

Configuration of network interfaces.

File: /etc/network/interfaces

`auto [interface]` – automatically bring up interface at boot time

`iface [interface] inet [type]` – begin section to define interfaces

Type can be manual, static or dhcp. You shouldn't need anything but static and dhcp configurations.

```
auto lo         #should always be in the interfaces file
iface lo inet loopback   #should always be in the interfaces
file

auto eth0       # Bring up interface at boot
iface eth0 inet dhcp     # Define a dhcp interface
```

This will bring up eth0 on boot in dhcp mode.

```
iface eth0 inet static   #Define a static-ip interface
<tab>     address xxx.xxx.xxx.xxx
<tab>     netmask xxx.xxx.xxx.xxx
<tab>     broadcast xxx.xxx.xxx.xx
```

`pre-up` and `post-up` can be used to run commands before and after bringing up and interface. `pre-down` and `post-down` can be used to run commands before and after shutting down an interface.

Bridging

Build a bridge for a 192.168.1.0/24 subnet across 3 interfaces (eth0, eth1 and eth4) with the OPSAID IP address 192.168.1.1. Then anything addressed inside that subnet can be plugged in those ports, including a hub/switch/router. It is also generally good practice to remove all other interface definitions for interfaces inside the bridge from the interfaces file. All of the auto lines for those interfaces must be removed.

```
iface OPSAIDnet inet static  #Define static-ip interface
<tab>     pre-up    brctl addbr OPSAIDnet #create bridge
#configure the interfaces for ipv4 multicast
<tab>     pre-up    ifconfig eth1 0.0.0.0
<tab>     pre-up    ifconfig eth4 0.0.0.0
<tab>     pre-up    ifconfig eth0 0.0.0.0
#add the interfaces to the bridge
```

```
<tab>     pre-up    brctl addif OPSAIDnet eth1
<tab>     pre-up    brctl addif OPSAIDnet eth0
<tab>     pre-up    brctl addif OPSAIDnet eth4
<tab>     address    192.168.1.1 #define the bridge ip info
<tab>     netmask 255.255.255.0
<tab>     broadcast 192.168.1.255
#after bringing down the bridge delete it
<tab>     post-down brctl delbr OPSAIDnet
```

`ifup` and `ifdown` are used to control the network interfaces from the command line. So you can `ifup eth0` to bring it up and `ifdown` eth0 to bring it down. To bring the bridge up/down, you would use `ifup`/`ifdown` OPSAIDnet. `ifconfig` at a shell prompt will list all interfaces that are currently up and their network information. `ifconfig -a` will list all network interfaces on the system.

`man interfaces` on the masterOPSAID will further describe how to use this file.

SSH to OPSAIDs

SSH'ing to the OPSAIDs from a UNIX command prompt

For OPSAID-ng-1:
`ssh -i /path/to/OPSAID-ng OPSAIDuser@[ip of OPSAID-ng-1]`
Password for the private key "`AsDfGhJkL`"
Once connected for root shell "`sudo -s`" password "`letmein`"

For masterOPSAID:
`ssh -i /path/to/OPSAID-master OPSAIDuser@[ip of OPSAIDmaster]`
Password for the private key "`AsDfGhJkL`"
Once connected for root shell "`sudo -s`" password "`letmein`"

For dropping to a serial port (more on this configuration later):
`ssh -i /path/to/ttySX OPSAIDdrop@[ip of OPSAID-ng-1]`
Password for the private key "`ZxCvBnM`"

With a putty configuration:
To convert a RSA private key to a putty private key,
start puttygen
Click "load", pick the rsa private key file and type in the pass phrase, then save the ppk file

To set up the session start putty
--> Session
HostName = <ip address of the OPSAID>
Saved Sessions = <name that you want to save it as>
--> Connection
Autologin username = <OPSAIDuser for admin sessions / OPSAIDdrop for tty drop>
ins

--> Connection --> SSH --> Auth
Browse to the appropriate private key file
--> Session
Save Session

Syslog-ng Configuration

in `/etc/hosts`
add a line for the OPSAIDmaster ip
`ip.of.OPSAID.master<tab> OPSAIDmaster`
delete all of the other OPSAIDmaster lines.
After the change,
`sudo /etc/init.d/syslog-ng restart`

Web Access to Syslog-ng

http://ip.of.SLAPmaster/
username: `slapuser`
password: `letmein`

Dropping to serial port configuration

`cd /etc/minicom`
`minirc.ttySX` contains the configurations for each of the serial ports to set the speeds. Alternatively you can do `minicom ttySX` and then do a `ctrl+a, o` to edit the settings from within minicom. Then save the new settings back to the file.

Appendix G: Sandia Throughput Testing Results

These tests compare three different configuration scenarios. In the first scenario (Crossover), the source and destination computers are connected directly by a network crossover cable. This would represent the most direct method of connection. In the second scenario (Routed), a pair of OPSAID devices has been inserted between the source and destination computers, providing the functionality of a router. In the third scenario (Routed with IPSec), the OPSAID devices are again inserted between the source and destination computers, this time encrypting and decrypting all network traffic.

It is important to note that all of the following scenarios include network bandwidth considerably greater than virtually any real-world PCS/SCADA network. Nevertheless, it is important to show how an OPSAID device's performance can degrade under load.

The first test sends 1.25MB of data over 10 seconds (1.05 Mb/s), using the UDP protocol. Transfer rate, jitter and packet loss are measured. The results show that the "Routed with IPSec" scenario performs identically to "Crossover".

```
IPERF UDP
Crossover
------------------------------------------------------------
Client connecting to 192.168.10.1, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:   103 KByte (default)
------------------------------------------------------------
[  3] local 192.168.10.2 port 32768 connected with 192.168.10.1 port 5001
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[  3] Sent 893 datagrams
[  3] Server Report:
[ ID]  Interval      Transfer     Bandwidth       Jitter    Lost/Total
Datagrams
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec  0.003 ms  0/893 (0%)

Routed
------------------------------------------------------------
Client connecting to 192.168.1.2, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:   103 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.2 port 32768 connected with 192.168.1.2 port 5001
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[  3] Sent 893 datagrams
[  3] Server Report:
[ ID]  Interval      Transfer     Bandwidth       Jitter    Lost/Total
Datagrams
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec  0.041 ms  0893 (0%)

Routed with IPSec
------------------------------------------------------------
Client connecting to 192.168.1.2, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:   103 KByte (default)
```

```
------------------------------------------------------------
[  3] local 192.168.2.2 port 32768 connected with 192.168.1.2 port 5001
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[  3] Sent 893 datagrams
[  3] Server Report:
[ ID]  Interval       Transfer     Bandwidth       Jitter    Lost/Total
Datagrams
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec  0.016 ms  0893 (0%)
```

The second test sends as much data as possible over 10 seconds, using the TCP protocol. The amount of data transferred and corresponding throughput rate are calculated. The results show that adding the routing between the source and destination devices reduce the effective throughput from 90.4 Mb/s to 82.9 Mb/s. This 8.3% reduction in throughput is typical of any routed network. As the routed network traffic is encrypted using IPSec, a 4.2% reduction in throughput is seen (82.9 Mb/s to 79.4 Mb/s). This reduction is well within typical tolerance limits, especially as PCS/SCADA devices would be unlikely to communicate at anything like 79.4 Mb/s.

```
IPERF TCP
Crossover
------------------------------------------------------------
Client connecting to 192.168.10.1, TCP port 5001
TCP window size: 24.5 KByte (default)
------------------------------------------------------------
[  3] local 192.168.10.2 port 51496 connected with 192.168.10.1 port 5001
[ ID]  Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   108 MBytes   90.4 Mbits/sec

Routed
------------------------------------------------------------
Client connecting to 192.168.1.2, TCP port 5001
TCP window size: 16.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.2 port 57303 connected with 192.168.1.2 port 5001
[ ID]  Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  98.9 MBytes  82.9 Mbits/sec

Routed with IPSec
------------------------------------------------------------
Client connecting to 192.168.1.2, TCP port 5001
TCP window size: 16.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.2 port 57899 connected with 192.168.1.2 port 5001
[ ID]  Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  94.7 MBytes  79.4 Mbits/sec
```

Appendix H: Sandia/Schweitzer Testing Results

First round of testing: SNL & SEL OPSAID Interoperability Lab Tests
Wednesday, January 10, 2007

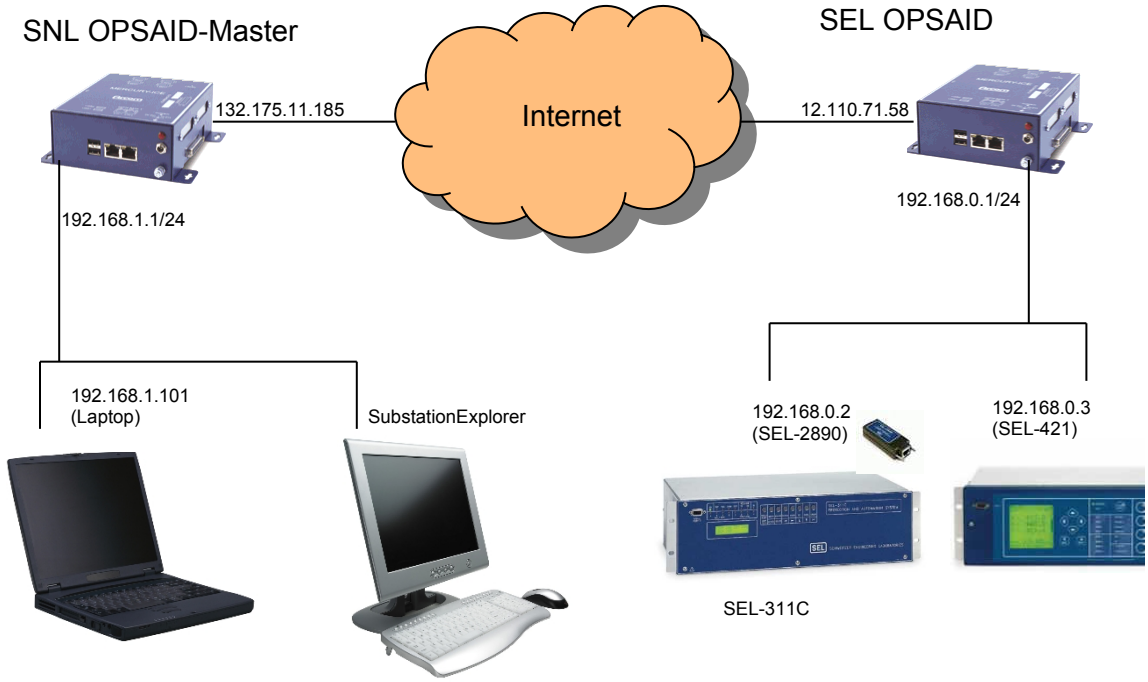

Figure H.1: First-Round Testing Configuration

What Worked

- SSH between SNL OPSAID-Master & SEL OPSAID
- Syslog-ng messages from SEL OPSAID logged at SNL OPSAID-Master (Unencrypted)
- Racoon IPSec tunnel (shared keys) between subnets 192.168.0.0/24 and 192.168.1.0/24
- Telnet session through IPSec tunnel from laptop (192.168.1.101) to SEL-311C (via SEL-2890 192.168.0.2)

What Did Not Work

- Racoon IPSec tunnel using Certificates/Certificate Authority
- Syslog-ng messages NOT encrypted between OPSAID devices

What Needs To Be Done

- Test Modbus / DNP Data between SubstationExplorer and SEL-421
- Fix Racoon IPSec tunnel using Certificates/Certificate Authority
- Fix unencrypted Syslog-ng messages between OPSAID devices by directing Syslog-ng messages to 192.168.1.1 instead of to 132.175.11.185

- Use StrongSwan and configure OCSP protocol

Second Round of Testing: SNL & SEL OPSAID Lab Tests Number 2

Friday, Jan 26, 2007

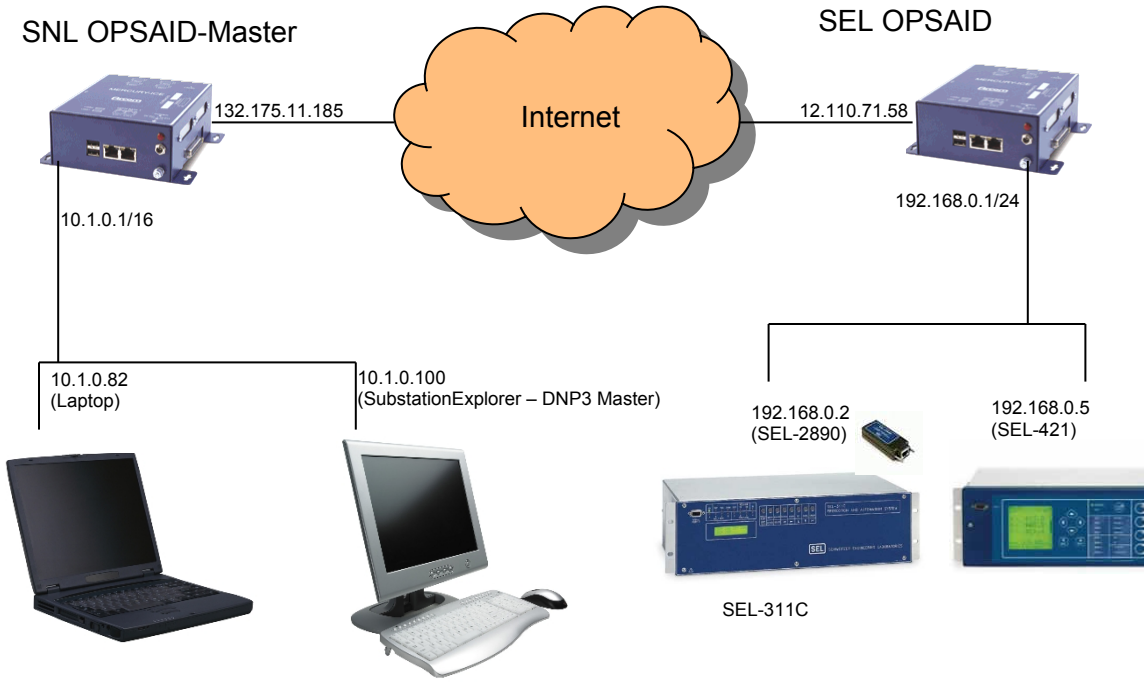

Figure H.2: Second-Round Testing Configuration

Attendees

Sandia (SNL): Regis Cassidy, Adrian Chavez, Bryan Richardson

Schweitzer (SEL): Ryan Bradetich, Dennis Gammel, Rhett Smith

What Worked

- SSH between SNL OPSAID-Master & SEL OPSAID
- Syslog-ng Messages from SEL OPSAID logged at SNL OPSAID-Master – Encrypted syslog-ng messages when using 132.175.11.185 interface of OPSAID-Master
- Strongswan IPSec Tunnel using (SNL created) Certificates Between Subnets 192.168.0.0/24 and 10.1.0.0/16
- DNP3 Polling/Responding via Strongswan IPSec Tunnel

What Did Not Work

- Strongswan IPSec tunnel using SEL-created certificates between subnets 192.168.0.0/24 and 10.1.0.0/16
- After the lab was completed, troubleshooting was performed on the SEL-created certificates and the problem was resolved (possible openssl bug). SEL-created certificates are now working in the SNL lab and need to be tested again via the network.
- Syslog-ng messages NOT encrypted between OPSAID devices using 10.1.0.1 interface of OPSAID-Master

What Needs To Be Done

- Test Modbus data between substationExplorer and SEL-421
- Fix Strongswan IPSec tunnel using certificates created by both SNL and SEL
- Fix unencrypted Syslog-ng messages between OPSAID devices by directing Syslog-ng messages to 10.1.0.1 instead of to 132.175.11.185
- Use OCSP Protocol to obtain Certificate Revocation List (CRL)
- Timing tests using IPerf, which can be installed with: `apt-get install iperf`

Third Round of Testing: SNL & SEL OPSAID Lab Tests Number 2
Thursday, February 8, 2007

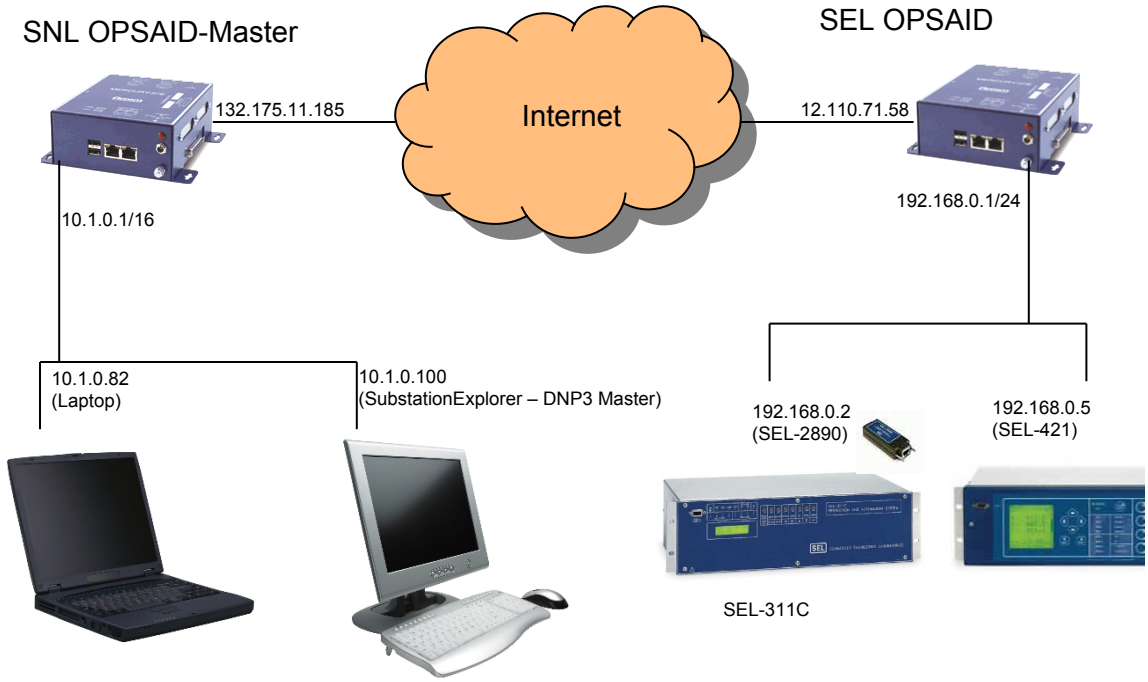

Figure H.3: Third-Round Testing Configuration

Attendees

Sandia (SNL): Regis Cassidy, Adrian Chavez, Bryan Richardson
Schweitzer (SEL): Ryan Bradetich, Dennis Gammel, Rhett Smith

What Worked

- SSH between SNL OPSAID-Master & SEL OPSAID
- Syslog-ng messages from SEL OPSAID logged at SNL OPSAID-Master – Encrypted syslog-ng messages when using 132.175.11.185 interface of OPSAID-Master
- Strongswan IPSec tunnel using SNL-created certificates between subnets 192.168.0.0/24 and 10.1.0.0/16
- DNP3 polling/responding via Strongswan IPSec tunnel
- Strongswan IPSec tunnel using SEL-created certificates between subnets 192.168.0.0/24 and 10.1.0.0/16
- Syslog-ng messages NOT encrypted between OPSAID devices using 10.1.0.1 interface of OPSAID-Master

What Did Not Work

- Everything tested worked this iteration

What Needs To Be Done

- Test Modbus Data between SubstationExplorer and SEL-421
- Use OCSP protocol to obtain Certificate Revocation List (CRL)

- Timing tests using IPerf, which can be installed with: `apt-get install iperf`