Spring 2011

Use of Service Oriented Architecture for Scada Networks

Scott H. Beavers
Regis University

Regis University
College for Professional Studies Graduate Programs
Final Project/Thesis

Disclaimer

THE USE OF SERVICE ORIENTED ARCHITECTURE FOR SCADA NETWORKS

A THESIS

SUBMITTED ON 24th OF May 2011

TO THE DEPARTMENT OF INFORMATION SYSTEMS

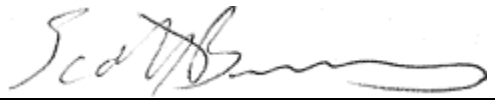
OF THE SCHOOL OF COMPUTER & INFORMATION SCIENCES

OF REGIS UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF MASTER OF SCIENCE IN

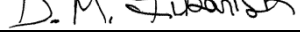
INFORMATION TECHNOLOGY MANAGEMENT

BY



Scott H. Beavers

APPROVALS



Daniel Likarish, Thesis Advisor

Nancy Birkenheuer

Richard L Blumenthal

Abstract

Supervisory Control and Data Acquisition (SCADA) systems involve the use of distributed processing to operate geographically dispersed endpoint hardware components. They manage the control networks used to monitor and direct large-scale operations such as utilities and transit systems that are essential to national infrastructure. SCADA industrial control networks (ICNs) have long operated in obscurity and been kept isolated largely through strong physical security. Today, Internet technologies are increasingly being utilized to access control networks, giving rise to a growing concern that they are becoming more vulnerable to attack. Like SCADA, distributed processing is also central to cloud computing or, more formally, the Service Oriented Architecture (SOA) computing model. Certain distinctive properties differentiate ICNs from the enterprise networks that cloud computing developments have focused on. The objective of this project is to determine if modern cloud computing technologies can be also applied to improving dated SCADA distributed processing systems. Extensive research was performed regarding control network requirements as compared to those of general enterprise networks. Research was also conducted into the benefits, implementation, and performance of SOA to determine its merits for application to control networks. The conclusion developed is that some aspects of cloud computing might be usefully applied to SCADA systems but that SOA fails to meet ICN requirements in a certain essential areas. The lack of current standards for SOA security presents an unacceptable risk to SCADA systems that manage dangerous equipment or essential services. SOA network performance is also not sufficiently deterministic to suit many real-time hardware control applications. Finally, SOA environments cannot as yet address the regulatory compliance assurance requirements of critical infrastructure SCADA systems.

Acknowledgements

In deepest gratitude to my wife, Janis Blanchard, for her loving encouragement always and her gracious patience throughout the course of this project. To my mother, Beverly, for her enthusiastic support. And to Daniel Likarish and Shari Plantz-Masters, my thesis advisors, with special appreciation for their guidance,.

Table of Contents

List of Figures

List of Tables

List of Appendices

Chapter 1 – Introduction

The acronym SCADA refers to Supervisory Control and Data Acquisition, a type of process control system (PCS) used to monitor and control processes that are distributed geographically (IBM Internet Security Systems [IBM-ISS], 2007, p. 3). Examples of such processes include power generation and distribution, water and waste treatment, or railway track management – systems that are often critical to a country's infrastructure. Because SCADA systems are integral to maintaining human health and safety, commercial productivity, and emergency or military responsiveness, they are often operated in a strict regulatory environment (Stouffer, Falco, & Scarfone, 2008). After safety, availability has always been the top design goal for these systems (Dzung, Naedele, Von Hoff, & Crevatin, 2005). In the context of control systems, availability is tightly intertwined with reliability as continuous operation is the overall performance driver (Fabro & Maio, 2007; Fenrich, 2007; National SCADA Test Bed [NSTB], 2010). Security is an essential aspect of reliable operations, as noted by Stouffer et al.: "Improved control systems security and control system specific security policies can potentially improve control system reliability and availability" (p. 51). Accordingly, the security of SCADA systems has been recognized as a key element of national defense. In 2002, a U.S. Department of Energy (DOE) report recommending greater adoption of control network security measures announced that "…SCADA networks provide great efficiency and are widely used. However, they also present a security risk. SCADA networks were initially designed to maximize functionality, with little attention paid to security" (p. 2).

While the origin of many SCADA systems predates the Internet, an increasing majority of these systems are now connected using Internet technologies. Fenrich (2007) clearly presents the nature of this change:

> Historically, process control systems…were typically operated in an isolated or stand-alone environment, and did not share information or communicate with other systems. These systems were normally comprised of proprietary hardware, software, and protocols designed specifically to control and monitor sensitive processes. Since access to these control systems was greatly limited, and knowledge of these protocols was limited to a small population, control system network…security efforts were minimal, and focused primarily on physical measures.
>
> Today…stakeholders are demanding real-time plant information be readily available from any location. This has led many previously stand-alone control systems to become part of the "always connected" world, where real-time control system information can be easily accessed…via corporate networks or Internet technologies. This increased connectivity, coupled with the adoption of standardized technologies, protocol implementations, and operating systems, has dramatically increased the focus on control system security. (pp. 1-2)

This new connectivity is eliminating the past security benefits of using proprietary systems that were almost unknown outside of industrial circles and instead brings to control networks a fresh exposure to standard IT network security concerns (Centre for the Protection of National Infrastructure [CPNI], 2008). Increased connectivity has also elevated certain additional and distinctive security issues for these systems. Some of these issues are technologically based as automation controllers have not typically been endowed with enough computing capacity to support the processing overhead of authentication, encryption, or other security measures (Stouffer et al., 2008). Further issues are based in performance reliability requirements as even the momentary disruption of a controlled process may not be acceptable. "Both the SCADA systems and the underlying physical systems have strict survivability

requirements on a twenty-four-hours-a-day, seven-days-a-week (24x7) basis. Here survivability means the capability of a system to fulfill its mission in a timely manner, even in the presence of attacks, failures, or accidents" (Xiao, Ren, & Kwiat, 2010, p. 1). Accordingly, "the use of typical IT strategies such as rebooting a component, are usually not acceptable solutions due to the adverse impact on the requirements for high availability, reliability and maintainability of the ICS" (Stouffer et al., p 29). The potential consequences of a control system failure extend beyond typical IT dimensions. As serious as it can be for a corporate network to be compromised by a worm or hacker, there are far more dire consequences inherent in the breach of a control system regulating operations at a nuclear power plant.

Cloud computing is based on service oriented architecture (SOA). It offers compelling business benefits for many consumers of enterprise IT services such that its use is growing increasingly common. As with SCADA, the cloud computing model is rooted in distributed processing and is possessed of certain security characteristics that differ from those of conventional networks. Unlike SCADA, cloud computing has mostly evolved within the past decade and is based on leading edge Internet technologies that offer tantalizing prospects for the future. The central question of this thesis concerns whether these prospects include the potential to improve SCADA control networks.

1.1 Research Questions

The central question this research seeks to answer is whether or not present-day cloud computing technologies can provide a worthwhile improvement over the conventional networking methods currently used to meet the distributed processing needs of SCADA industrial control systems. Improvement in this case can largely be rated as an increase in reliability and availability or a decrease in cost with no sacrifice to either. The best opportunity

for improvement may not come in operational performance but through increasing the security of control networks in the Internet age.

SOA and SCADA systems are both based on distributed processing models but it is not immediately obvious how readily one may be substituted for the other. Of concern is how these models map to each other and whether SOA's unique attributes are suited to addressing SCADA's distinctive operational requirements. Research is required to determine if the technological advances making SOA development feasible are also suited to improving process control systems.

In order to evaluate the applicability of cloud computing to improving control networks, the research for this thesis addresses the following questions:

- What are the benefits of cloud computing?
- To what degree do these benefits accrue to control networks as opposed to enterprise networks, particularly in regards to security?
- What are the performance requirements for SCADA control networks?
- Can SOA be expected to meet SCADA performance requirements?
- Are SOA information assurance capabilities (security and compliance) sufficient to meet the regulatory standards required of many SCADA systems?

1.2 Assumptions

It is assumed that the application of cloud computing to control networks will not be commercially viable unless SOA implementations at least match the operational performance of legacy SCADA systems. Any cloud-based solution can be no less stable or robust than current systems, should address the full range of control requirements presently satisfied by conventional implementations, and will otherwise meet or exceed present-day performance in all critical areas.

Chapter 2 – Review of Literature and Research

The literature review for this project encompassed three primary areas including the security and performance requirements of SCADA systems; the technological nature and performance aspects of cloud computing; and the application of design science methodology in research. Research into SCADA security amalgamating general enterprise IT security risks and mitigations with the differentiating security considerations specific to control networks. Security was given strong emphasis in researching the merits and constraints particular to SOA implementations and their potential operational impact on – and suitability for – SCADA systems.

A significant majority of literature reviewed for this project was drawn from two types of sources: peer reviewed works from journals and proceedings or publically reviewed compendia of best practices developed by industry groups and governmental or quasi-governmental bodies such as the Department of Homeland Security, NIST, and various national laboratories. These latter guidelines are frequently produced in the form of mandated standards that must be applied to systems developed for use by federal programs. Due to the current rapid evolution of SOA in deployment, peer reviewed material was supplemented by gleaning consensus opinions from timely industry sources. These sources included white papers and references produced by professional organizations and consortia such as the Cloud Security Association that are spearheading the latest efforts to develop standards and best practices for cloud computing.

Peer reviewed literature concerning the design science methodology was also reviewed to establish what criteria to apply in determining sufficiency with regard to the work performed for this project.

2.1 SCADA Systems and Industrial Process Control

SCADA systems constitute one class of automation control systems more generally referred to as Process Control Systems (PCS) or Industrial Control Systems (ICS). A control system may be as simple as the volume control knob on a radio or a thermostat that regulates the temperature maintained by an oven or air conditioner. SCADA control systems are used to manage hardware typically associated with large-scale operations such as factories, utilities, transportation systems and the like.

The fundamental nature of process control is different than that of enterprise computing. Weiss (*Control systems cyber*, 2009) spells out two central distinctions. First, tasks in the business IT model generally have a defined beginning and end whereas the process control model is built around the continuous loop. While the IT community generally avoids the continuous loop, it is the continuous loop that enables an ICS to operate efficiently and safely. Second, the end user of an enterprise system is usually a person whereas the end user in a SCADA system is most likely to be a computer or other control device.

One of the most frequently cited distinguishing properties of control systems that it that they are deterministic, that is, process control often has strict real-time or near real-time requirements for monitoring and response (Byres & Hoffman, 2004; *Control systems cyber*, 2009; Dawson, Boyd, Dawson, & Nieto, 2006; Dzung et al., 2005; Fabro & Cornelius, 2008; Fenrich, 2007; Fernandez & Fernandez, 2005; Miller, 2005; Naedele, 2007; Stouffer et al., 2008). In contrast, the standard for “real-time” in enterprise systems typically reflects the amount of delay that is acceptable before a user loses patience.

These characteristic differences between systems result in a certain subset of performance requirements that separate process control from enterprise applications. This separation in turn

leads to diverging implementation standards by impacting areas such as system design and security orientation.

SCADA Components and System Organization

SCADA systems are multi-tiered and may be viewed hierarchically from two parallel perspectives, control and communications. Control may be considered in terms of both logical and physical distance from the endpoint hardware components or machinery while communication relates to being closest to the top-level SCADA controller, sometimes referred to as a master terminal unit (MTU) or a real-time processor (RTP). At the lowest communication level, the components of a control system will consist of somewhat intelligent "field devices" such as intelligent electronic devices (IEDs), or programmable automation controllers (PACs) (IBM-ISS, 2007). These components may control somewhat basic processes and sub-processes but often simply regulate a limited collection of very fundamental hardware endpoints such as valves, relays, motors and solenoids or monitor sensors for temperature, pressure, pH, current and so on.

While a field device may communicate directly with a SCADA controller, it will frequently be managed or coordinated along with other field devices by intermediate controllers such as remote terminal units (RTUs), programmable logic controllers (PLCs), or distributed control systems (DCSs) that serve to control complex processes or sub-processes (Stouffer, et al., 2008). Some intermediate controllers have extensive capabilities that may rival those of the SCADA controller. A DCS, for example, controls an entire site, factory, or process that is geographically situated so that all components can be connected on a LAN. Although a typical RTU is often described as a being a ruggedized field device on the order of some IEDs (Patel, Bhatt, & Graham, 2009), they can often provide fairly complex control of an entire process or

field station. PLCs have evolved to offer a great deal of flexibility and may be configured to function as an RTU or be sophisticated enough to serve as a low-end DCS. Common usage of terms for the various types of process controllers shows some fuzziness of definition and there is, at all levels, a certain amount of overlap in the range of capabilities for each.

A SCADA controller consists of a SCADA server, most often coupled to a separate computer that provides the human machine interface (HMI). An HMI normally offers graphical view of system conditions and events in real time while serving as the terminal through which an operator enters commands, run tests, and respond to alarms. The other key component of a SCADA system is the data historian. This is server used to keep a time line record of process states and activities. Multiple or redundant instances of any SCADA component will exist according to the operational requirements of the system.

The distinguishing characteristic of a SCADA controller is that it manages processes over a wide enough area that some form of WAN is required to connect all of the constituent elements (IBM-ISS, 2007). The process may not actually be as complex as one typically managed by a DCS but the system will be geographically distributed. The WAN may utilize any manner of communication links including wired, wireless, or satellite and follow a mix of standard process automation protocols such as ICCP, DNP3, or Modbus, as well as TCP/IP (Giani, Karsai, Roosta, Shah, Sinopoli & Wiley, 2008; Graham & Maynor, 2006; IBM-ISS, 2007). Technical precision is often waived as the term “SCADA” is used interchangeably with “control” to refer to any control network or system, regardless of the actual network span.

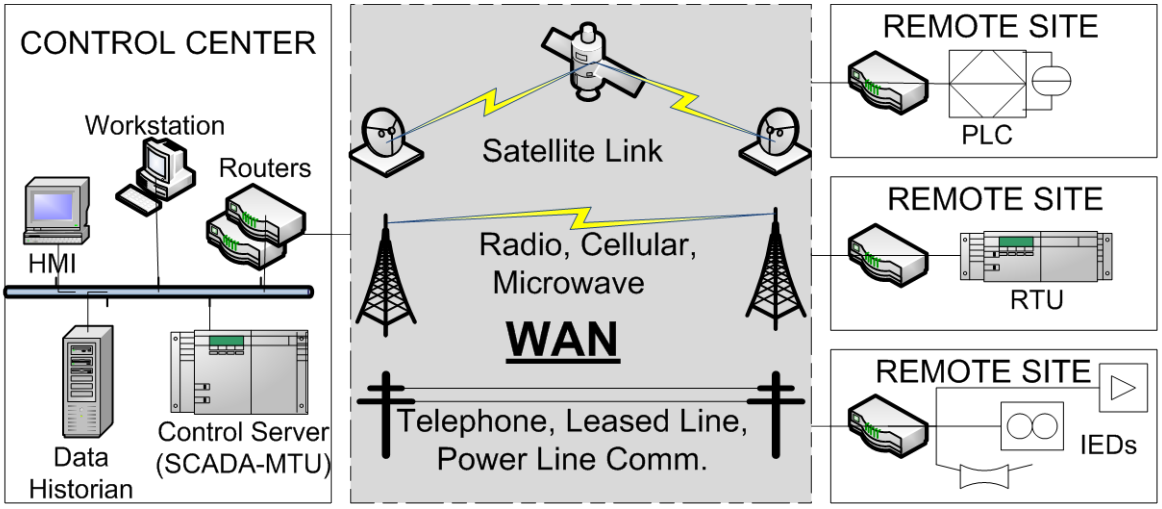


Figure 1: General SCADA System Layout

The SCADA controller may oversee a local site in conjunction with remote operations, in which case it is also connected to a production system LAN. LANs in an automation control environment typically employ a diverse set of channels and protocols. The mix utilized is less often driven by bandwidth requirements than by considerations such as immunity to electrical noise or poor grounding, signal attenuation over the distance required, and the choice of communication interfaces provided by the vendors of particular specialized equipment. Serial communications via RS-232 or RS-485 are not uncommon and, along with previously identified process automation protocols, may utilize proprietary protocols mixed with standards such as GPIB or, more recently, USB. This LAN, or control network, may also be referred to as the process control or industrial control network (PCN or ICN), as distinct from the enterprise, corporate, business, or IT network. In the case of SCADA systems, the PCN extends to include the SCADA WAN as well.

A good application to use as model for envisioning SCADA systems is that of electrical production and distribution. An individual electrical generation plant will be controlled by a DCS that manages the plant's many processes, sub-processes, and components. A SCADA

system will provide for the centralized monitoring and coordination of a number of generation plants within a region as well as overseeing the power distribution network and its associated substations (each of which has its own RTU). In this way, the SCADA system manages power generation to match total system load requirements and parcels out that load to different locations according to current demand in each area.

OPC

In the world of PCNs, OLE for process control (OPC) plays a significant role. OPC is a very widely used set of protocols for integrating the multifarious communications protocols utilized by automation hardware vendors and has, in turn, received a great deal of vendor acceptance and support (Byres Research, 2007; Dzung, et al., 2005; Tu, Cuong, Tan, & Thang, 2010). OPC utilizes a client-server communication architecture to deliver one of the key features also found in SOA: a simplified means to link heterogeneous systems, of which there are multitudes in the automation and process control industry. The recommended architecture for highest-reliability systems includes redundant OPC servers with a "heartbeat" signal to keep each server alert to the other's status, allowing the backup server to know when to assert control (Dzung et al.; Tu et al.). Figure 2 illustrates the differences between conventional control network communications and OPC communications.

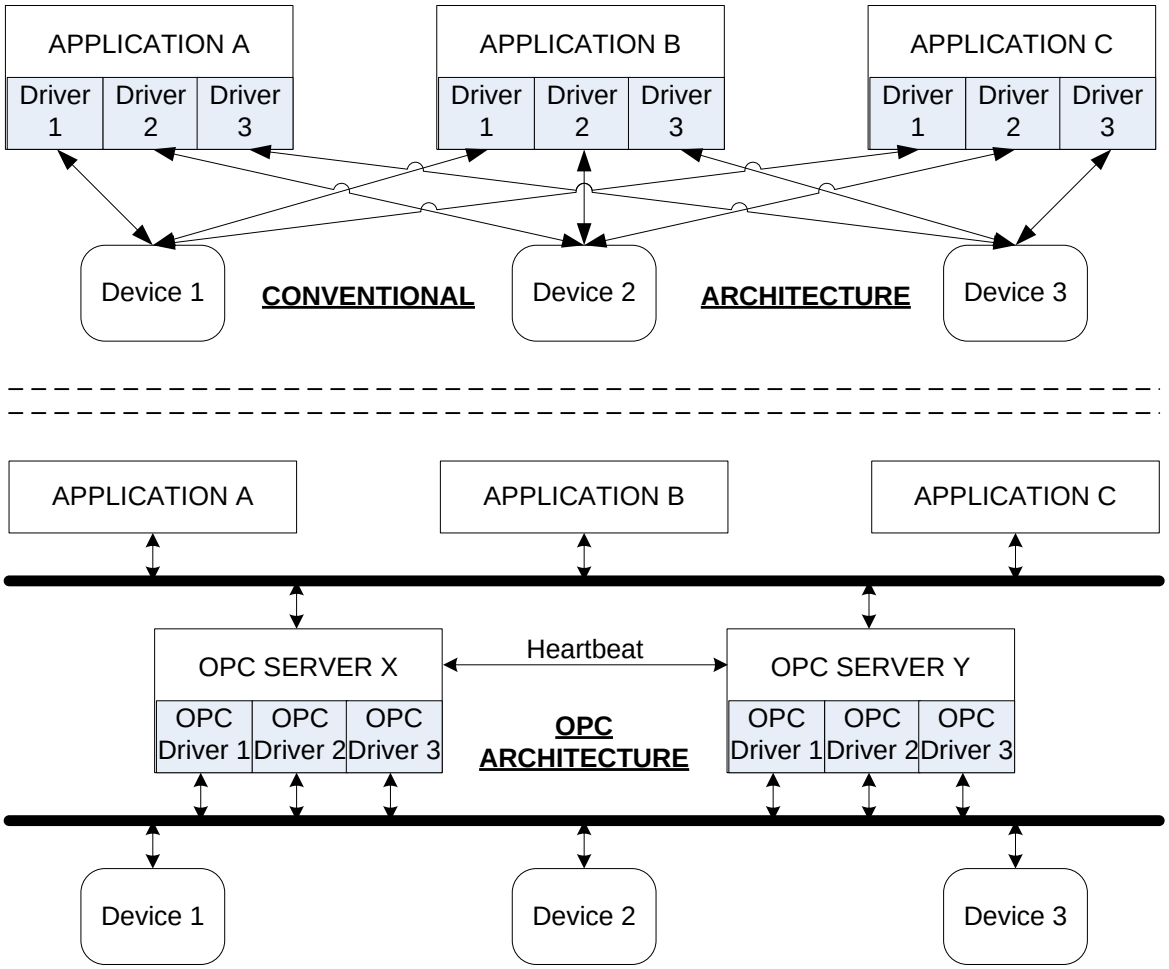


Figure 2: Conventional vs. OPC Communication Architectures

OPC is built on Microsoft's remote procedure call (RPC) and distributed component object model (DCOM) protocols central to its Object Linking and Embedding (OLE) technology (Byres Research, 2007; Tu, 2010). The OPC Foundation has more recently developed OPC-UA (OPC universal access), a backwardly compatible OPC based on XML and .NET Web services that provide a foundation for the potential adoption of a service oriented architecture. The intent is to address, among other things, better OPC security and Microsoft's plans to retire DCOM "but it may be a number of years before many companies actually convert their systems" (Byres, 2007b, p. 4) so OPC "classic" is very much the prevailing technology.

Threat Evolution with SCADA Systems

SCADA systems pre-date the Internet but are now largely connected to TCP/IP networks. These connections often occur in unintentional ways that are not recognized by system managers and are therefore overlooked for their security risks (Graham & Maynor, 2006). According to Naedele (2007), "direct…and indirect…links between the automation system and external public networks is often underestimated" (p. 2).

Recognized or not, these connections expose previously insulated control networks to the entire host of mainstream enterprise network vulnerabilities (Cagalaban, Kim, & Kim, 2008; Dzung et al.; Fenrich, 2007; Landau, 2008; Taylor, Krings, & Alves-Foss, 2002). Even though the rate of Internet adoption in process control environments has lagged that of business and personal computing, by 2004 studies were showing that the overall rates of control systems incidents had increased five-fold in the preceding10 years while between 2002 and 2004 the percentage of external incidents more than doubled to 66% from a 20-year baseline of 29% (Miller, 2005).

There are distinctive consequences of an ICS security breach above and beyond those applying to enterprise networks due to standards for fault tolerance, downtime, operational safety, equipment damage, etc. As itemized by Fabro & Cornelius (2008, p. 23):

> The consequences associated with cyber incidents in a control systems environment can vary and can include:
>
> - Loss of localized or remote control over the process
> - Loss of production
> - Compromise of safety

- Catastrophic cascading failures that affect critical infrastructure and can extend to peer sites and other critical infrastructure sectors
- Environmental damage
- Injury or loss of human life

For historical reasons, “controls systems that are not Windows-based generally have poor security designs and weak protection” (Byres, 2004, p. 5). Even so, these systems have largely escaped external attacks and, until fairly recently, reports of process systems succumbing to an Internet attack have been rare. Until PCNs were connected to the Internet, physical security provided the greatest measure of protection and incidents were largely the result of insider actions, both accidental and deliberate. The proprietary nature and rarely understood workings of automation systems software also created a barrier to outside attack, often referred to as “security through obscurity” – potentially effective protection but only under conditions that are seldom realized in practice today (Udassin, 2008b). It is not clear that external attacks have yet superseded internally induced events but the trend is in the making. Appendix A provides a summary of many of the best known industrial control security incidents organized by date to help highlight the progression from events being internally generated to instead being externally induced.

A SCADA server may provide an attacker entrée to a corporate network and vice versa. Increasingly, SCADA systems have been impacted by common network attacks, not because these attacks specifically targeted the lower process control tiers but because hardware control was incidentally affected when control network servers or pc-based controllers were disrupted. One of the best known incidents occurred when the safety monitoring system of the Besse-Davis nuclear power plant was disabled by the Slammer worm because a contractor plugged his laptop

into a local service port behind the firewall while being remotely logged in to his company's infected network (Naedele & Dzung, 2005).

Malware moved closer to directly threatening automation controllers with the exploitation of vulnerabilities in the RPC DCOM interface that is at the core of the widely used OPC standard (Byres, 2007a). In 2007, for the first time ever, the OPC network server of a popular SCADA system used to control railroads, oil refineries, dams, and nuclear power plants was discovered to harbor a remotely exploitable vulnerability that could crash the system and potentially even allow it to be taken over (Vaas, 2007).

Since then, not only have PCNs been challenged but the proprietary operating systems used for process control have come under increasing scrutiny and even attack. In 2008, an Israeli SCADA security firm, C4, was the first to document and demonstrate a remotely exploitable vulnerability that would definitely allow the takeover of an industrial control system (Udassin, 2008a). Subsequently, the Stuxnet worm was discovered in June 2010. Stuxnet was the first rootkit to specifically target a manufacturer's proprietary operating system for industrial control (Falliere, 2010). Not only did Stuxnet use sophisticated techniques to infiltrate Windows systems running targeted SCADA software, it also could identify a variety of PLCs under SCADA control and then download and hide blocks of code on those PLCs capable of secretly impacting hardware activities. The Stuxnet-targeted SCADA system supplied by Siemens is in common use running nuclear facilities and the worm, though widely distributed geographically, was found to be clustered primarily at nuclear installations in Iran. Analysis of Stuxnet code revealed that an early form of the worm existed for roughly a year prior to its discovery and that a newer upgraded version circulated undetected for several months.

Security Impact of Differences between Control and Enterprise Networks

Control networks and enterprise networks coexist within organizations that are engaged in process control. The primary point of contact between the two networks is usually the Data Historian, which is a server like many that would commonly be found on the corporate network. Workstations are generally the most visible computers the SCADA controller level. The rather ordinary nature of these most visible PCN components contributes to the impression held by many IT groups that the distinctions between control and corporate networks are minimal. While much of the computational hardware that is employed may be similar, the performance requirements for control networks are at strong variance to those for enterprise networks. Weiss made the point in his 2009 testimony before the Senate "that IT encompasses a large realm, but does not include ICS processes" (*Control systems cyber*, 2009). Table 1 presents a summary of the most significant differences in performance expectations between control systems and enterprise IT systems.

Table 1

Major differences between performance expectations for enterprise IT systems and PCS systems

Category	Enterprise IT Systems	Process Control Systems
Performance Requirements	▪ Consistent response time required ▪ Demand is for high throughput ▪ High delay and jitter may be acceptable	▪ Real-time, response is time-critical ▪ Modest throughput is acceptable ▪ High delay and/or jitter is not acceptable
Availability Requirements	▪ Occasional failures may be tolerated ▪ Depends on operational needs ▪ Acceptable problem responses include actions such as rebooting ▪ Field beta testing can be acceptable as is timely – or even automated – software change management	▪ Maximum availability is essential ▪ Redundant systems may be required ▪ Rebooting likely not acceptable and is a slow painstaking process ▪ Downtime planned well ahead of time ▪ Changes, including patching, are infrequent and require exhaustive pre-deployment testing to assure they do not disrupt ICS processes
Risk Management Requirements	▪ Top priorities are data confidentiality and integrity ▪ Fault tolerance is less consequential	▪ Top priorities are human safety, then process protection/continuity ▪ Fault tolerance is critical – even

	– transient downtimes generally not a big risk ▪ Major risk impacts are delay of business operations and exposure of proprietary information or methods ▪ Outsourcing viable and utilized extensively, increasingly so with the advent of cloud computing	momentary downtime may not be acceptable ▪ Major risk impacts include loss of life, environmental damage, equipment damage, regulatory non-compliance, and lost production ▪ Outsourcing rare except for vendor maintenance contracts
Security Orientation	▪ Emphasis on protection of IT assets and the information stored or shared among these assets ▪ A security breach may make system unavailable and result in lost data integrity or confidentiality	▪ Emphasis on protecting edge clients (e.g., field devices that monitor and control processes) ▪ A security breach may result in a dangerous or lethal outcome for many people
Computational Resource Constraints	▪ Systems are specified with enough resources to support additional third-party applications such as security solutions	▪ Automation control systems often lack capacity to support or cannot tolerate the latency of added security measures like encryption, IDS, or anti-virus ▪ Penetration testing network may disrupt operation
System Components	▪ Typical lifetime of 3-5 years ▪ Usually local and easy to access	▪ Typical lifetime of 15-20 years ▪ Often remote and can be very difficult to access

While security remediation for enterprise and control networks will overlap, there are consequential differences that must be recognized. This can be demonstrated by highlighting just a couple of the distinctions between the two types of systems: computational resources and security orientation. IT systems are consistently specified with the communication bandwidth and enough memory and processor speed to accommodate security measures like anti-virus programs and encryption protocols. Control systems, by contrast, typically communicate at a fairly pedestrian pace and have little in the way of excess computational resources over and above what is required for their hardware management duties (*Control systems cyber,* 2009; Dawson et al., 2006; Dzung et al., 2005; Naedele, 2007; Stouffer et al., 2008). Common enterprise security measures like software patching or network scanning can easily disrupt process controllers (Duggan, Berg, Dillinger, & Stamp, 2005; Naedele; Stouffer et al.).

When considering the three core attributes of security – confidentiality, integrity, and availability – IT systems generally rank their importance in just that order, with integrity possibly tying with confidentiality for first place (Fabro & Cornelius, 2008). For control systems, the rank order is reversed with availability being far and away of the highest importance, and then integrity of the data/message stream followed at some distance confidentiality (*Control systems cyber*, 2009; Control Systems Security Program [CSSP], 2009; Dzung et al., 2005; Fabro & Cornelius; Fabro & Maio, 2007; Fenrich, 2007; McQueen, Boyer, Flynn, & Beitel, 2006; Miller, 2005; Naedele, 2007; Stouffer et al., 2008). "Whether this is correct, it is indeed a result of historically non-cyber security cultures having to make system uptime the primary operational activity" (Fabro & Maio, p.18). While there are other contrasts between enterprise systems and control systems, these distinctions in computational resources and security posture are the predominant drivers of differences in security remediation decisions between the two types of systems.

As with enterprise IT security, securing SCADA systems requires a defense-in-depth approach. In the ICS security realm, certain misinformed beliefs have been widely held that form an impediment to enhanced security. According to Xiao et al. (2008), these beliefs include (in italics with discussion following):

- *Control network inaccessibility: PCNs are physically segregated from the Internet and access from corporate networks is protected through strong access control measures.* In practice, many control systems are externally accessible by modems that may be discovered by "war dialing" then compromised by password cracking utilities (Stouffer et al., 2008). Graham and Maynor (2006) document many cases where ICS networks were subject to compromise because they were not as well isolated as operators believed them to be. Wireless accessibility is

also a potentially vulnerable feature of many control systems. Too often, employee access to a control network relies on IT network security while the link between networks utilizes weak authentication protocols and grants privileges to too many users. (Fernandez & Fernandez, 2005; Permann, Hammer, Lee, & Rohde, 2006; Stouffer et al., 2008)

- *Security through obscurity: process systems consist of niche applications and proprietary networks that require specialized knowledge to access and control.* As Byres (2004) points out, HMI controllers in SCADA systems rely heavily on Windows platforms with commonly known vulnerabilities. This leaves them subject to disruption at least, if not a targeted attack on operational controls. Appendix B summarizes certain key findings of Homeland Security's ICS vulnerability assessments performed by the Idaho National Labs (NSTB, 2010) which reveals that the 10 most critical ICS vulnerabilities all ranked "High" for attacker awareness of their existence. Eight of the 10 vulnerabilities were ranked "Easy" or "Moderate" for attacker ease of detection, "High" or "Widespread" for prevalence of being found ICSs, and had CVSS v2 scores between 9.0 and 9.8 out of 10 for impact/damage potential.
- *A control system network breach will not confer end point control: a process' hardware operations will not be impacted if the ICN is compromised due to the specialized knowledge needed to exert control over field units like PLCs or IEs.* In practice, field units communicate often in clear text with message banners that identify what units are on the network while the information needed to gain control of these units is freely available on the Internet (Graham & Maynor,

2006). Often the necessary information can be found on the compromised network (Udassin, 2008b). It has further been demonstrated that an unknown control protocol may be reverse-engineered within a matter of days (Naedele, 2007). Stuxnet provided further proof that the need for proprietary knowledge is not a sufficient barrier to prevent field unit attacks.

Even given the corporate will to secure control systems, strategies for securing enterprise systems do not always translate well to a control network environment. "Factors such as operational isolation, legacy networking, and inflexible roles in job activities may not be conducive to creating environments that are rich with cyber security capability, functionality, or interest" (Fabro & Maio, 2007, p. 1). For this reason, there is a need for enterprise and control systems specialists to work jointly on ICN security issues rather than relying overmuch on standard IT systems expertise (Stouffer et al., 2008).

One aspect of ICS security that does correlate strongly with enterprise systems is that of utilizing secure network architecture. Isolation of control networks is a major objective and wherever possible an "air gap" is employed such that there are no external network connections to systems that directly control equipment. This is a requirement for core nuclear plant control systems and appears to be one reason that the Stuxnet worm was designed to propagate through both USB and network connections.

Sometimes vendors will have external connections to a control network for maintenance purposes but the primary vulnerability to external intrusion is via corporate network connections that are established for a variety of useful and convenient business reasons (Stouffer et al., 2008).

Both Homeland Security (CSSP, 2009) and Canada's National Infrastructure Security Coordination Centre (Byres, Karsh, Carter & Savage, 2005) reviewed network architectures that

have been used over time to promote securing the connections between enterprise and corporate networks. Solutions using only routers or dual homed servers are no longer considered to have any effective security value and even placing a single firewall between networks is deemed only marginally useful when a low-risk control network is involved. The most worthwhile network architectures use multiple firewalls to place a DMZ between the control and enterprise networks as illustrated in Figure 3. This topology ensures that communication between networks is never direct but occurs primarily through data exchanged via the SCADA Data Historian located in the DMZ, in effect buffering the control network from the corporate network as if it was an open conduit to the Internet. The same conceptual network segregation might also be enforced with the use of VLANs to implement the firewall zones rather than multiple firewalls, though VLANs come with additional security vulnerabilities that must be carefully addressed (Dzung, et al., 2005; Leischner & Tews, 2007).

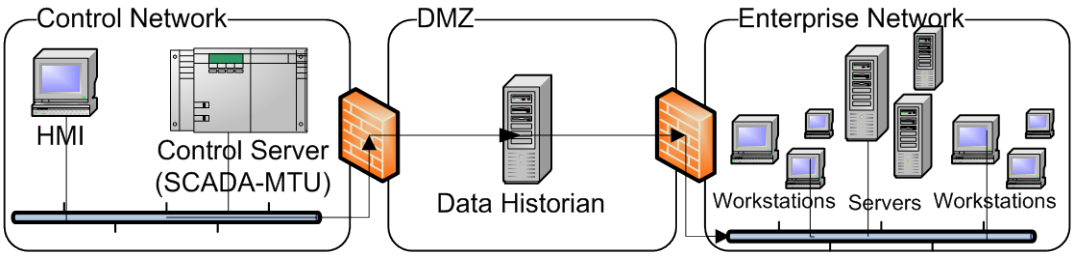


Figure 3: DMZ Segregation of Control and Enterprise Networks

Company networks are often thought to be more secure than they are so the policies controlling connections to the control network from inside the enterprise are not suitably stringent. While it is necessary to understand technical solutions that extend into the control system space, there is also a great need to focus on the "soft side" of security – issues such as policy, training, and organizational security culture. As Naedele (2007) notes: "Security is in

the first place not a technical issue. In consequence, some of the largest challenges in making control systems more secure relate to human behavior and the perception of the problem" (p.2).

2.2 Cloud Computing

The commonly used term "cloud computing" is an informal label applied to networks using Service Oriented Architecture (SOA) for which NIST provides this broad working definition: "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" (Mell & Grance, 2009, p. 9). The terms "cloud computing", "SOA", and "the cloud" are used interchangeably in this thesis.

One primary aspect of SOA is that the various parts of the system are integrated with each other by means of messaging. Since the messaging mechanisms used are o/s- and platform-independent, heterogeneous systems are encapsulated behind their messaging interface and can be made to effectively interoperate with each other over the network. And, since a cloud is network based, one of its key characteristics is that its component elements may be widely distributed geographically in a fashion similar to SCADA systems.

Units of software functionality, called Web services in SOA, are conceptually related to methods in object-oriented programming in that the inputs and outputs are public but the manner of execution remains private. "A web service is basically a collection of related operations, with each operation (e.g. placeOrder) being associated with a message or a pair of messages (placeOrderRequest, placeOrderResponse)" (Kearney, 2005). Alternatively, Mell & Grance provide a more general picture of Web services as "self-describing and stateless modules that perform discrete units of work and are available over the network" (2009, p. 71). Also known

simply as “services”, Web services are frequently constructed to be “multi-tenant” so that simultaneous use by multiple requestors is possible (Cloud Security Alliance [CSA], 2009). One or more instances of each service may exist at various locations within a cloud.

A cloud’s services are catalogued in a service registry. This registry may be publically accessible or kept private (within a single organizational entity). The registry is a database containing a roster of web services, with a formal description of their functions (methods, inputs, and outputs), and their location in the cloud (Sinha, Sinha, & Purkayastha, 2010). Service requestors query the registry to “discover” web services that meet their needs. The entire process of connecting requestors to services is brokered using a Web services protocol stack consisting – in the most basic view – of four protocols that provide for transport, messaging, description, and discovery.

XML and the Web Services Protocol Stack

XML (eXensible Markup Language) provides the foundation for three of the four Web services protocols: messaging, description, and discovery of Web services (Rosenberg & Remy, 2004, chap. 1). Only the transport protocol (i.e., SMTP, FTP, HTTP, etc.) is not rooted in XML. XML is also being applied to extending trust and security capabilities for Web services through such means as the Security Assertion Markup Language (SAML), the eXtensible Access Control Markup Language (XACML), and a group of SOAP extensions (WS-Security, WS-Trust, etc.) identified collectively as WS-* (Martino & Bertino, 2006).

XML uses a text-based format to provide a structured self-describing means of representing data in a manner independent of the platform, operating system, programming language or protocol used by any system. Different languages, including the document type definition (DTD) language that is part of the XML specification, are used to create XML

schemas that define the rules for an enormous variety of XML documents and domain-specific languages (Rosenberg & Remy, 2004). It is the flexibility of XML that has allowed its use for the creation of Web services protocols.

The Simple Object Access Protocol (SOAP) is built on an XML schema that defines the predominant Web services message protocol. "It allows a program running in one system to call a program running in another system and it is independent of any programming model" (Rahaman, Schaad, & Rits, 2006, p. 78). SOAP messages exchange structured information, principally XML data, between Web services, most frequently utilizing HTTP/HTTPS as the transport protocol.

The Web Services Descriptor Language (WSDL) is an XML implementation of the Web services description protocol used to define the set of operations provided by Web services along with the structure of their related SOAP messages (Rosenberg & Remy, 2004). This description, known as a "WSDL", is used by one service to tell other services how to interact with it, where the service is located, what the service can do, and how it is invoked.

The service discovery protocol is another XML offshoot, UDDI (Universal Description Discovery and Integration). UDDI is used to compose the Web services registries where WSDLs are published. "It defines a set of standard interfaces for accessing a database of Web services. The purpose of UDDI is to allow users to discover available Web services and interact with them dynamically. The process can be divided into three phases: Searching (discovery), Binding and Executing" (Sinha et al., 2010, p. 135).

Figure 4 illustrates the relationship between SOAP, WSDL, and UDDI, the XML-derived protocols utilized for the Web services model.

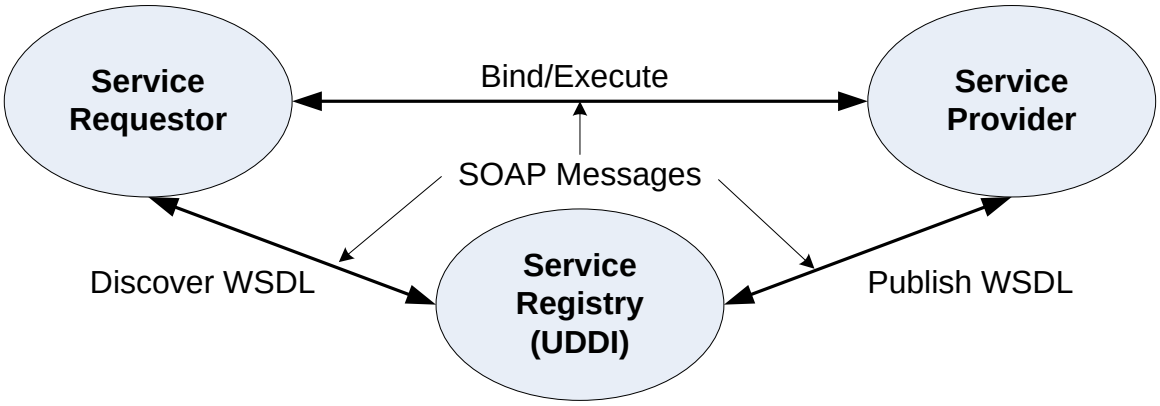


Figure 4: XML Protocols and the Web Services Model

Web Services Performance Considerations

While flexibility and system agnosticism make XML a powerful and useful tool, it also has its drawbacks. Martino and Bertino (2006) observe that use of XML for Web services may hamper SOA system performance due to the amount of processing overhead involved, lead to interoperability issues caused by deficiencies or incompatibilities of evolving standards, and adversely impact security through a variety of inherent vulnerabilities.

Security vulnerabilities in XML begin with the fact that it is a text-based standard, which creates a requirement to encrypt sensitive data to avoid compromising confidentiality in the event it falls into the wrong hands (Martino & Bertino, 2006). For SOAP messages, routine transport layer encryption (SSL/TLS) is not entirely sufficient for this purpose because it only provides point-to-point security. SOAP messages may be transmitted in multiple "hops" through a series of services, each of which is an endpoint. This means that transport layer security ends at the first hop rather than at the ultimate destination. Although TLS security may be separately applied to each hop, what is really required is message level security that can provide end-to-end validation that each separate leg of the transmission was secured (Hongzhao, 2010; Martino & Bertino; Rahaman et al., 2006; Tan, Yoo, & Yi, 2009). End-to-end message security is evolving

through the use of XML-Encryption and XML-Signature in accordance with WS-Security and other related SOAP extensions such as WS-Policy and WS-Trust (Rodrigues, Estrella, & Branco, 2011; Rahaman et al.).

Researchers have documented numerous other concerns regarding XML vulnerabilities including XML schema poisoning (Mehta, 2009), XML rewriting attacks (Rahaman et al., 2006), and WSDL scanning and enumeration attacks (Breytenbach, 2005; Mehta). Sinha et al. (2010) raise the additional issue of "how to ascertain that the description of the service provided by the registry and the actual service provided by the provider for binding are the same" (p. 137) and note that tampering with a WSDL in transit or storage can cause a variety of malfunctions.

Standards for Web services can be problematic because they are not always fully compatible. Martino & Bertino (2006) point to three types of standards that are in effect. *De facto* standards are those in wide use that may or may not be officially recognized as *de jure* standards by legal international standards bodies such as ISO while a third kind of standard, the *consortium recommendation*, is prevalent in SOA, particularly as to access control and messaging security extensions. Consortia recommendations may eventually be adopted as *de jure* standards, but in the mean time "the Web service standards community such as OASIS, IETF, and W3C among numerous others are producing a plethora of specifications which, in principle, could offer potential solutions…However...there is still considerable fluidity in these developments and a variety of implementations that exist" (Jie, Arshod, Sinnot, Townend, and Lei, 2011, p. 12:24). Further, many Web service standards are developed using a layered approach where upper layer standards can use and extend the standards from lower layers. As a result, the standards for different layers are often developed by different standardization bodies such that "standard specifications are not always compatible" (Martino & Bertino, p. 21). As

illustrated by Carlson & Himler (2005): "Most Web services specifications provide a number of mechanisms for accomplishing the same thing, which can lead to interoperability issues. As an example, a sender can encrypt data using Triple DES, AES 128, AES 192, or AES 256 algorithms, but the receiver might only be able to decrypt AES 128" (pp. 28-29). This example typifies "the need of a flexible negotiation approach that enables the system to dynamically adapt to changing conditions" according to Martino & Bertino (p. 23), who also express concerns about "the real interoperability between the standard implementations by different manufactures [*sic*]" (p. 22).

There are additional Web services interoperability issues caused by semantic incompatibilities. "One of the biggest stumbling blocks in the grand vision proposed by SOA is data heterogeneity between interoperating services. By data or message level heterogeneities, we refer to incompatible formats of messages exchanged by the services" (Nagarajan, M., Verma, K., Sheth, A. P., Miller, J., & Lathem, J., 2006, p. 373). Simple examples might include an instance where one service defines test performance as GRADE(A-F) versus another service's SCORE(0-100), or GRADSTUDENT(ID, Name, Addr) versus STUDENT(ID, Name, Addr, StudentType[GRAD]). Addressing issues of this nature is one of the factors driving work on the Semantic Web and the Web Ontology Language (OWL) (Nagarajan, et al).

As issues of Web service security and interoperability requirements have become better understood, the processing overhead required to address them has also raised performance concerns (Hinton et al., 2005; Martino & Bertino, 2006; Rahaman et al, 2006; Tu et al., 2010). Studies to evaluate the transmission and processing times of secured versus unsecured SOAP messages indicate a three- to over 10-fold increase for invoking security measures (Rodrigues et al, 2011; Tan et al., 2009). Some manufacturers have responded to these concerns with the

development of XML appliances. Martino and Bertino describe the volume of processing that may be offloaded to these appliances, providing further insight into the potential amount of overhead involved:

> Such products are commonly referred to as XML appliances and include the XML accelerators and the XML firewalls. A XML accelerator appliance is a customized hardware and software where the following processing consuming tasks are performed: XML/SOAP parsing, XML schema validation, XPath processing and XSLT transformation functions. XML firewalls, also known as XML security gateways, are devices that, in addition to the functions of a XML accelerator, support the WS-Security standards and a range of security-related functions such as: content or metadata-based XML/SOAP filtering functions; XML messages encryption/decryption at the message or element level; XML signature verification and XML message signing according to XML Encryption standard; Authentication and authorization functions (that in some XML appliance can be based on local or on on-board repositories); Auditing and accounting functions. (p. 22).

Even without adding overhead for security processing and the like, the fact that XML is text-based innately increases system overhead because text does not transmit as compactly as binary data (Martino & Bertino, 2006; Tan et al., 2009; Tu et al., 2010). This issue was addressed through the creation of a set of standards for using SOAP messages to transmit large binary data (Martino & Bertino), resulting in at least a six-fold decrease in message size (Tan). However, regarding XML in the control network world of OPC, “the binary encoding routines are of particular concern, especially in embedded controllers, which have been especially prone to parsing errors in the past” (Byres Research, 2007a, p. 26). This issue could take some time to

fully resolve as only time will tell when the Web services binary data standards are stable and then allow for these standards find their way into fully validated control systems which are, in turn, deployed at very low turnover rates.

Virtualization

Virtualization is not a part of cloud computing by definition but it is so often present as a feature that it is warranted for inclusion in any consideration of the cloud (CSA, 2010). Virtualization adds a key cost benefit to cloud computing through the part it plays in enabling utility computing. As multi-tenant services allow units of code to be shared, virtualization allows units of hardware – servers and storage systems in particular – to be shared. Like a multitasking operating system that oversees the sharing of processor cycles and I/O access between several tasks on a single machine, the virtual machine hypervisor shares the many processors and other resources of a high-power server among many "virtual" computers. Unlike a task, which is a single software component with limited functionality, a virtual machine has the capability of running its own operating system and distinct set of applications as if it was contained in a separate standalone box.

Benefits of Cloud Computing

There are a number of potential benefits to cloud computing for consumers of computing services. The message-based nature of SOA provides a means to loosely couple both local and remote network elements (Flurry, 2007). This loose coupling enables interoperability between heterogeneous systems, meaning that new upgrades and installations may be deployed without simultaneously replacing essential, but incompatible, legacy systems (Frievald, 2008). Loose coupling can further allow for interoperability between different cloud infrastructures (Badger & Grance, 2010).

Cloud computing also enables the acquisition and use of computing services in a fashion that approaches the model common to utilities, where – as with household power or water – only the scale of services required is utilized and paid for (CSA, 2010; Mell & Grance, 2009). Shared web services support renting software on an as-used basis rather than owning it. Virtualization facilitates utility computing by enabling incremental expansion through the addition of a single virtual machine that utilizes only a fraction of a complete physical server that would need to be acquired otherwise. Underutilized capacity is minimized by making a single pool of resources available to meet the needs of multiple user groups. Through resource sharing, processing capacity is idle less often and only a modest reserve capacity is needed to support the peak needs of all users compared to what is required when each user group provisions independently to meet occasional peak demand.

What cloud computing means for small and medium-sized businesses (SMBs) is that they can have utility-like access to sophisticated computing capabilities without making an otherwise prohibitive investment in equipment and IT staff (Frievald, 2008). For large companies that might afford full IT departments, cloud computing provides a means to outsource a substantial amount of mundane IT activity and focus on strategic issues.

There are three deployment models for cloud services: Infrastructure-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS) (Badger & Grance, 2010). In the order listed, there is an increasing shift of control from the service consumer to the cloud provider. This shift of control also represents a cost benefit from increasing reliance on outsourced IT staff functions. In all cases, the customer is leasing hardware with no direct control of the underlying rack server or hypervisor and can only negotiate management policies through service level agreements (SLAs) with the provider (CSA, 2010).

With IaaS, the customer retains full control of the service layers above the hardware level, i.e., the operating system, middleware and, at the top level, the user applications. IaaS is targeted to providing customers with fundamental computing resources like processing, storage, and network capacity in a manner highly analogous to typical pc leasing. IaaS augments PaaS to the extent of deploying customer-created or -specified applications to the cloud. In this role, service providers will additionally assume full control of the operating system while sharing with customers the administrative control over middleware and applications. The customer will also retain a certain amount of programmability control for these layers as well. The intent of SaaS is to enable the customer to use the provider's applications over a network (Mell & Grance, 2009; CSA, 2010). Accordingly, the customer cedes almost all control to the service provider except for retaining limited administrative capabilities like adding and deleting accounts.

Badger & Grance (2010) also identify four basic deployment models for clouds: private, community, public, and hybrid. The infrastructure of a private cloud is owned or leased to be operated solely for use by a single organization. It may be managed by a third party and located either on or off premises. A group of organizations sharing common interests or concerns may maintain a community cloud infrastructure, again applying any mix of management and location options. A public cloud is entirely owned and managed by a single organization for the purpose of providing cloud services to the public. The hybrid cloud infrastructure consists of two or more unique clouds that are bound together by technologies that enable data and application portability between them.

Details and distinctions aside, the promise of cloud computing boils down to three core advantages for service customers: agility, control and cost (Frievald, 2007; see also Baer, 2008). The distributed nature of the cloud provides agility because components may be added, removed,

or modified without impacting other areas while additional functionality and resources may be taken on in very discrete, affordable increments, also a cost benefit (Chatarji, 2004). Further cost benefits include streamlining the IT workforce and the application of standards across the cloud that leverage rather than compartmentalize skill sets (SOATutorial.net, 2010). Control is enhanced in two ways, the first being that users may gain better control of the features they need because solution delivery is componentized in the cloud rather than being tightly coupled to large interdepartmental systems like ERPs. Loose coupling also gives IT more freedom, and therefore better control, over how the required functionality is delivered technically, facilitating the selection of "best in breed" solutions (SOATutorial.net). The desire to obtain these advantages is driving the adoption of cloud computing at an increasing rate.

Cloud Performance

Virtually all cloud computing infrastructures involve a great deal of complexity. A good sense of this complexity can be gained by exploring one aspect of SOA performance: quality of service (QoS). QoS manifests in concerns regarding critical issues such as reliability, performance, interoperability and security (Balasubramaniam, Lewis, Morris, Simanta, & Smith, 2009). The nature of SOA itself creates additional issues in these areas beyond those previously identified for Web services and their underlying reliance on XML.

The distributed nature of the cloud makes reliability and performance difficult to design and test (Balasubramaniam; Roch, 2006). Component services are too often black boxes so that basics like the error models employed are not understood between decoupled developers. There is a lack of universal semantic standards for describing service functionality for the discovery process (Nagarajan, Verma, Sheth, Miller, & Lathem, 2006). This can confound interoperability and, at the least, make it difficult to distinguish differences between related services.

Applications are composed of a collection of services that are not linked until an application is deployed and not all services will be under a given developer's control, so thorough end-to-end testing is impossible. Further, the performance of multi-tenant services can be very dependent on the number of active requestors at any given time while any number of other system load factors can unpredictably affect SOA applications. "In service-oriented systems, which involve multiple independent capabilities, or services, combined into an application or composite capability, validating and monitoring QoS is challenging" (Balasubramaniam et al., 2009, p.1). These challenges are even greater when significant portions of a cloud are leased, which puts a large number of services, policies, and procedures in the hands of vendors who are not necessarily amenable to granting customers access to their systems for audit, let alone control, purposes (CSA, 2010).

Cloud Security and Compliance

Security and compliance loom large among cloud computing QoS issues. One 2008 survey by the IDC Enterprise Panel found security to be the preeminent concern with respect to the on-demand model of cloud computing (Mell & Grance, 2009, p. 17). Aside from those vulnerabilities inherent in SOA's XML infrastructure, a cloud's distributed nature creates additional security issues.

Security in the SOA environment is confounded by some very fundamental factors. Farkas & Huhns (2008) note: "Some of the characteristics that make service-oriented architectures appealing for enterprise applications also make them vulnerable to security breaches. The vulnerabilities are primarily due to the openness of the service-execution environment, to the dynamic run-time selection and composition of services, and to the autonomy of the individual services" (p. 428).

Access control is cornerstone issue with SOA as “Web services do not have a clear notion of a security perimeter” (Carlson & Himler, 2005, p. 2) and permissions for accessing web services do not map well to a role-based security model like RBAC (Alam, Hafner, & Breau, 2008). Related considerations of federated identity management, single sign-on (SSO), authorization and authentication are also significant factors with respect SOA usability as well as security and compliance (Carr, 2008; Jei et al., 2011; Martino & Bertino, 2006). A cloud may encompass multiple security domains, making it inconvenient to the point of impracticality to require users to separately sign-on in each domain that contains an essential Web service. On the one hand, authorization and authentication must be sufficient to allow users access to the most secure domain hosting a Web service they require. Balasubramaniam et al. (2009) point out that a “service can potentially be reused by multiple consumers in different contexts that have their own security requirements” (p. 104), meaning that some Web services may be hosted in less secure environments than the service requestor’s context and accessing those services may represent an unacceptable security risk.

The degree of awareness required in multiple areas presents an additional difficulty to implementing effective security. Often security approaches focus on application-to-application interactions and overlook the interstices between infrastructure, platform, and application levels – issues like hypervisor vulnerabilities and purging latent data from temporary storage or the shared memory of multi-tenant services (CSA, 2009).

Compliance is a concern because verification can be confounding for the same reasons that reliability and performance testing is difficult, a situation that can be aggravated when assets are under the control of cloud services providers. SOA is dynamic, system binding occurs at execution time and operational performance can be very dependent on the current resource

demands within the cloud (Balasubramaniam et al., 2009). Since control systems can be critically sensitive to real-time response, it is not enough to assure that the correct steps occur in the proper order. To a degree seldom required in enterprise networks, compliance for a control network involves the assurance that event recognition and response unfailingly occurs within strict time limits (Dzung et al., 2005; Fernandez & Fernandez, 2005).

The Enterprise Service Bus

In every area, the user convenience provided by distributed on-demand computing services is fraught with complexity for the developers and providers of those services. The enterprise service bus (ESB) is a software infrastructure that facilitates application integration and helps manage this complexity (Tibbling, 2007). The ESB accomplishes this through simplifying the implementation of SOA's most critical feature, loose coupling, acting as an intermediary between heterogeneous elements in a cloud and forming a backbone for Web service transactions. "As an intermediary, the ESB performs service virtualization to mediate the differences between service requesters and service providers, and offers aspect-oriented connectivity to act as an enforcement point for SOA policies, such as management and security. The loose coupling permits a clean separation of concerns (temporal, technological, and organizational) between the parts in a solution to enable flexibility and agility in both business processes and IT systems" (Flurry & Reinitz, 2007). Given that the major role of the ESB is simplification, it only makes sense to adopt an ESB only after the system has been built up to a tipping point of 25 production services or so (Lawton, 2010).

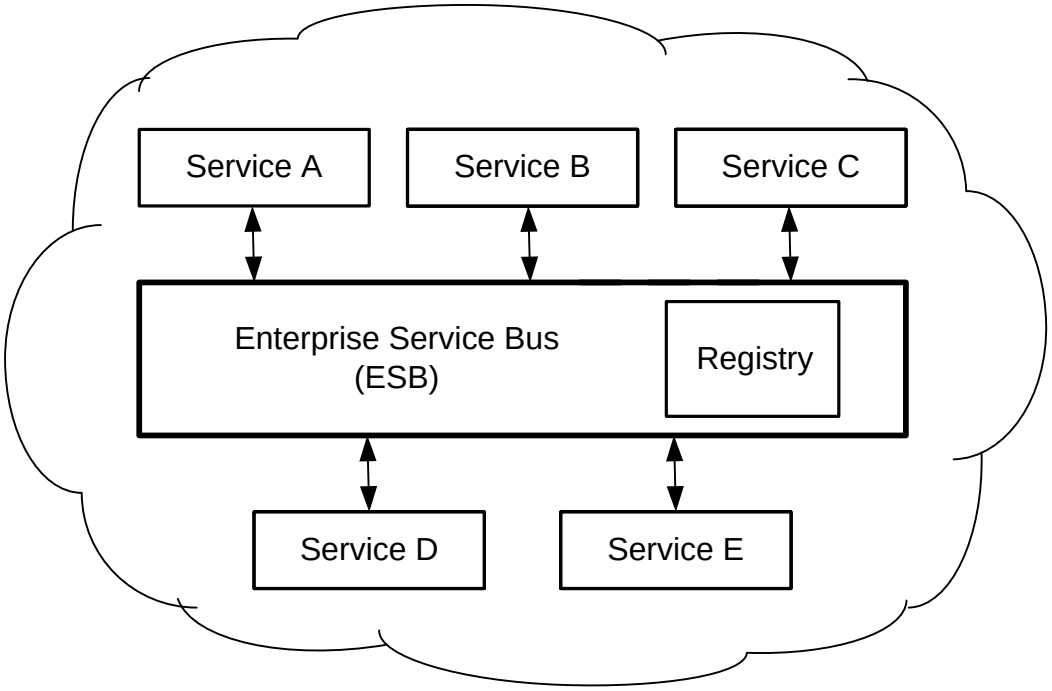


Figure 5: The Enterprise Service Bus

Figure 5 depicts the ESB's position in a cloud as a message broker between services. In this capacity, the ESB has the primary responsibility to translate between the different message types and transport protocols native to the various services it connects. To simplify this work, businesses will often adopt a Canonical Message Model (CMM) for use with the ESB (Selvage, Flurry, Sauter, & Lane, 2008). Consider the ESB operating like a group of translators at the United Nations, the Web services being U.N. diplomats. Potentially, each translator would need to be able to directly convert their native speech to dozens of other languages spoken by different diplomats. This is clearly an impractical expectation. However, if every translator was bilingual in Esperanto, all native speech could be translated into Esperanto and then retranslated into any other language required by the appropriate translator Like Esperanto, the CMM serves as a language common to all, eliminating the need to convert any Web service message to more than one other format.

Since the Web services registry is the target of a great deal of messaging activity, it is typically incorporated in the ESB. Besides serving as message broker and translator, an ESB can also provide a central point to enforce elements of security policy by virtue of its being the conduit for all messages exchanged between services (Flurry, 2007; Hinton, Hondo, & Hutchison, 2005; Tibbling, 2010). The downside of its role as a conduit is that it can also become a choke point through poor design, insufficient resourcing, or cyber attack. Despite its centralized role, SOA design means that the ESB itself is most likely to be a distributed entity. This diminishes the prospects of it becoming a single point of failure but also weakens the chances it will provide uniform security policy enforcement as implementation quality will vary among scattered development teams.

2.3 Design Science Research

A design science approach was adopted for this project (Hevner, March, Park, & Ram, 2004; March & Smith, 1995). "The sciences of design are a relatively new entrant to the set of methodologies, paradigms and orientations that have been dominated by debates previously only positioned as positivist versus interpretive and quantitative versus qualitative" (Purao et al., 2008). While design science has become an accepted research method in architecture and various engineering fields, it has been slower coming into its own in the IS/IT arena (Peffers, Tuunanen, Rothenberger, & Chaterjee, 2008). This lag may be somewhat attributable to conflict over the dualistic nature of scientific interest in IT as being both descriptive and descriptive. March and Smith describe the issue in these terms:

> Though not intrinsically harmful, this division of interests has created a dichotomy among IT researchers and disagreement over what constitutes legitimate scientific research in the field. Such disagreements are common in fields that encompass both

> knowledge-producing and knowledge- using activities. They are fostered in part by the prestige attached to science in modern societies and the belief that the term "science" should be reserved for research that produces theoretical knowledge. The debate in IT research is similar to that between engineering and the physical sciences. Knowledge-producing, "pure" science normally has the upper hand in such debates. In IT, however, the situation is different. It could be argued that research aimed at developing IT systems, at improving IT practice, has been more successful and important than traditional scientific attempts to understand it. With the issue undecided, the field is left in an uneasy standoff. (p. 252)

While design science may generate the sort of general or theoretical insights associated with the basic research of “pure” science, the approach has a definite applied science orientation. “The design-science paradigm seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts” (Hevner et al, 2004, p. 75). March and Smith (1995) observe that in contrast with natural science “*design science* attempts to create things that serve human purposes. It is technology-oriented. Its products are assessed against criteria of value or utility - does it work? is it an improvement?” (p. 253). So, where the Wright brothers drew from practical success in wing design to form a generalized theory of lift, merely achieving flight was a sufficient outcome for design science purposes.

March and Smith (1995) identified four research activities: building, evaluating, theorizing and justify. Theorizing and justifying were excluded from design science as involving a natural science rather than design science intent. Design research, therefore, involves building and evaluating artifacts where artifacts were identified as constructs, models, methods, and instantiations. “Conceptually, a design research artifact can be any designed object in which a

research contribution is embedded in the design." (Peffers et al., 2008, p. 55). While the term "build" in this context always refers to the construction of an artifact, it means "implement" only if the artifact is an instantiation. "An instantiation is the realization of an artifact in its environment.... Instantiations operationalize constructs, models, and methods" (March & Smith, p. 258).

Artifacts, then, are the outputs of design science's first activity, building, which are then subject to its second, evaluation. "An important aspect of design science research is the evaluation of the proposed artifacts; in other words, the utility of the proposed artifacts must be demonstrated" (Adomavicius, Bockstedt, Gupta, & Kaufman, 2008, p. 781). A cross-section of design science processes found in the literature reveals some bias towards creating instantiations for evaluation artifacts as they allow concrete testing by doing (Offermann, Levina, Schönherr, & Bub, 2009). However, an artifact may also be a model or design that may be evaluated through expert judgment or that may be depicted with sufficient rigor to enable implementation for evaluation purposes (Hevnor, et al. 2004). The value in this is that the expense of implementation is avoided when artifact fails to pass muster on evaluation criteria such as cost, performance, or excessive complexity.

This same value is realized in design science through the use of *ex ante* evaluation, which "is well developed for the purpose of deciding whether or not to acquire or develop a technology.... When regarded from the perspective of design research, ex ante evaluation provides models for theoretically evaluating a design without actually implementing the material system or technology. In other words, the artefact [*sic*] is evaluated on the basis of its design specifications alone" (Pries-Heje, Baskerville, & Venable, 2008, p. 256). Ex ante evaluation notwithstanding, artifact generation predominates in design science and wide scope is given for

their construction according to the utility required. “Such artifacts are represented in a structured form that may vary from software, formal logic, and rigorous mathematics to informal natural language descriptions” (Hevner et al., p. 77).

Chapter 3 – Methodology

A design science methodology was adopted for this research project. As stated by Hevnor (2007), “Good design science research often begins by identifying and representing opportunities and problems in an actual application environment” (p. 89). The overarching purpose of this research was to produce some result that could be of practical applied value. "Engineering disciplines accept design as a valid and valuable research methodology because the engineering research culture places explicit value on incrementally effective applicable problem solutions" (Peffers et al., 2008, p. 47).

An extensive review of the literature was performed to determine if the distributed processing model of cloud computing could be effectively applied to improve upon conventional networking technologies currently utilized in distributed SCADA process control networks. The key performance requirements for control networks were ascertained along with the technologies currently deployed. Research was also undertaken to uncover current solution deficiencies that might provide areas of opportunity to focus on improvement.

Further research was performed on the present state of cloud computing and the capabilities manifest in its component SOA technologies. These capabilities were assessed against the key control network requirements identified. An architectural model was constructed to embody the current potential for SOA application in a process control network. The descriptive narrative accompanying the model artifact augments the depiction at a greater level of detail.

Chapter 4 – Project Analysis and Results

Cloud computing and SCADA systems utilize two different architectures to implement distributed processing. Cloud computing has been developed on enterprise networks to meet IT needs for providing business services. It is based on SOA and typically also incorporates the use of virtualization. SCADA control networks were developed using pre-cloud technologies to provide automated control of physical rather than virtual processes. Control networks share certain commonalities with enterprise networks but also have differences related primarily to system performance requirements and the nature of some of the hardware clients residing on the network. Findings on the two types of systems are spotlighted below to determine the potential applicability of cloud computing for improving the implementation of SCADA systems.

4.1 Findings on SCADA

When attempting to qualify enterprise network mechanisms for use in a process control environment, it is important to recognize key operational differences between the two types of systems. Although similarities predominate, the distinctions that do exist impact the orientation, approach, and capabilities that are factored into the design and implementation of each of network.

Many SCADA systems have real-time response requirements that are seldom present in enterprise systems. “IT systems are ‘best effort’ in that they get the task complete when they get the task completed. ICS systems are ‘deterministic’ in that they must do it NOW and cannot wait for later as that will be too late” (*Control systems cyber*, 2009, p. 4). The need for deterministic behavior fits hand in glove with other key distinctive process control requirements. “ICS typically have many unique characteristics—including a need for real-time response and extremely high availability, predictability, and reliability” (Ross, 2007, p. I-1).

Security is an essential component of reliability (Stouffer, 2008) but as a consequence of ICS requirements, the order of importance for the "three pillars of security" in enterprise networks – confidentiality, integrity, availability – is reversed for control networks. Fenrich (2007) summarizes the reasons for this reversal: "Availability and fault tolerance are paramount – 99.99% uptime is required – because the process being controlled is continuous and can be unstable if not supervised; Integrity remains a necessity to ensure end-to-end data accuracy; and confidentiality – except for the protection of proprietary product recipes and plant security data – is of lower importance" (p. 6).

Regulatory compliance routinely plays a larger role in control networks than in enterprise systems. The nature of many processes under ICS control is such that human health and safety can be threatened by process instability, which may also result in equipment and environmental damage (Duggan et al., 2005; Stouffer et al., 2008). Failure of such processes can also cause widespread social losses as when utility failures disrupt transit, medical, or business operations, diminishing the welfare and productivity of a region. For all these reasons, the scope of compliance in SCADA systems extends beyond the assurance of consistent and correct process execution ordained by regulatory acts such as Sarbanes-Oxley. Regulations involving health, safety, and other aspects of social welfare also come into play.

Even given the above differences, recommended security measures for SCADA systems follow a depth-in-defense approach very much like that of enterprise networks but with some adjustment of emphasis. Whereas protecting core servers is usually the top objective in enterprise system security, SCADA systems are oriented towards isolating the control network, protecting edge clients, and generally enhancing the safety, reliability, and availability of the process control function (Dzung, et al.; Fabro & Maio, 2007; Fenrich, 2007; NSTB, 2010).

Initially, physical security was the primary requirement of control networks because those networks were established before the advent of the Internet, so the potential for external intrusion was minimal. "As such, the culture that has been developed as it pertains to system security is one that is tied to availability of the systems, as well as to the reliability of the system to perform its function. Measures are taken to protect the system in this regard and often do not include consideration for cyber security" (Fabro & Cornelius, 2008). Growing recognition of the increased vulnerability of control systems to intrusive threats is leading the ICS field to begin following enterprise system security recommendations in so far as the real-time response requirements and limited computational power of field devices will allow.

Information is readily available on applying the usual panoply of enterprise threat remediation measures for taking a depth-in-defense posture for control networks (Fenrich, 2007; Scarfone & Hoffman, 2009; Stouffer et al., 2008). There is strong agreement on best practices as promulgated by NIST to configure networks by deploying multiple firewalls and DMZs for isolating control networks, most particularly from associated enterprise networks (IAONA, 2003; Stouffer et al.), while Byres et al. (2005) add particular advice on isolation practices in instances where only a single firewall sits between the process control and corporate networks.

The legacy of cyber-insecure ICS design has meant that the full range of concerns surrounding identity management and access control are often quite poorly managed even now (Graham & Maynor, 2006; Patel et al., 2009; Udassin, 2008a). Since field systems often have auxiliary access ports behind the firewall for diagnostic and maintenance purposes, this is an area of control systems where additional attention to issues of authorization and authentication is required (DOE, 2002; Holstein & Diaz, 2006).

NIST also provides detailed guidance on standard system hardening activities such as restricting unnecessary applications, blocking unused ports, and configuring or disabling services (Scarfone & Hoffman; see also IAONA; Stouffer et al.). Additional ICS-specific guidance is available for hardening against RPC, DCOM and other vulnerabilities that become prevalent with the use of OPC (Byres Research, 2007c; Dzung, et al., 2005).

Many other aspects of defense-in-depth have been thoroughly considered from an ICS perspective. Besides NIST, U.S. government agencies such as Homeland Security and the Department of Energy have developed guidance regarding security policies and enforcement (Fabro & Maio, 2007; Industrial Control Systems Cyber Emergency Response Team [ISC-CERT], 2010; NSTB, 2010).

Less clearly resolved are how to effectively manage newer software vulnerabilities such as buffer overflows, SQL injection, or cross-site scripting (NSTB, 2010; Stouffer et al., 2008) that are commonly recognized in enterprise computing but are less know though also present in ICS. Part of the problem in dealing with threat evolution in SCADA networks is that operational reliability requires system stability, which can be threatened by routine IT security maintenance activities. Patch management, for example, is risky and complicated for ICS as changes that enterprise systems find completely innocuous can derail process control (Dzung et al., 2005; Naedele, 2007; Stouffer et al., 2008).

The hardware devices at the edge of the control network closest to physical processes are very resource constrained compared to the typical clients found on an enterprise network (*Control systems cyber*, 2009; Dawson et al., 2006; Dzung et al., 2005; Naedele, 2007; Stouffer et al., 2008. Lack of memory and processor power severely curtails the use of typical enterprise host-based security solutions for IDS, anti-virus, and the like because of the computational

overhead required for these activities (Dzung et al.; Stouffer et al.). This same concerns apply to the use of encryption or network scanning to increase security. A number of expensive incidents have been reported where control networks have been disabled or systems driven to damage by well-intentioned network scans, including a simple ping scan (Duggan et al., 2005). If the added activity does not simply overwhelm these devices, they readily become so mired down that they are no longer able to meet real-time operating requirements (Duggan et al.; Naedele). Even data logging capabilities are limited or non-existent, giving rise to a very limited ability to capture events and perform forensic assessments (Fabro & Cornelius, 2008). As a result, more reliance is placed on firewall security, perhaps supplemented by the addition of in-line appliances that buffer PLCs and the like with IDS or cryptographic services (Holstein & Diaz, 2006).

Complexity is another critical issue for SCADA networks because complexity makes changes difficult and, more importantly, impacts reliability (Fernandez & Fernandez, 2005; Miller, 2005; Stouffer, 2008). Factors like having thousands of data inputs and a hodge-podge of heterogeneous subsystems make process control systems intrinsically more complex and unique than IT systems notes Weiss, adding that ICS designers "view 'the enemy of the ICS' not as an attacker, but rather system failure. Therefore the ICS design uses the 'KISS' principle (keep it simple stupid) intentionally making systems idiot-proof" (*Control systems cyber*, 2009, p. 3). In addition to design complexity, Stouffer points out that "regulatory compliance can add complexity to security and authentication management, registry and installation integrity management, and all functions that can augment an installation and operational qualification exercise" (p. 6-31).

4.2 Findings on SOA

The development of cloud computing has so far has been focused on meeting the needs of typical business users and satisfying the performance requirements of enterprise IT. Research identified little work having been done to date to vet cloud computing as a sound solution for addressing the needs of SCADA networks and automation control, particularly in regard to real-time process control. Web service components like XML and SOAP were adopted by the process control industry's standard bearer, the OPC Foundation, to create OPC-UA. However, this cannot be construed as an endorsement of cloud computing for control networks. Rather, these developments were undertaken in response to Microsoft's plans to abandon DCOM, the technology at the heart of the original OPC (Byres, 2007a).

The advent of cloud computing is the result of a number of computational system advances in software, hardware, and system design and architecture. Faster hardware and, most particularly, faster networking have been essential enablers of cloud computing due to the absolute dependence of SOA on a high volume of messaging activity. As SOA has evolved, typical Ethernet transfer rates have increased from Megabit to Gigabit ranges and an increasing volume of high bandwidth fiber optic cables have been brought on line. Meanwhile, WAN communication has been enhanced by the application of multi-protocol label switching (MPLS) to further accelerate high-speed protocols like SONET, ATM, and frame relay (Fischer, 2007). Some of these networking advances are finding their way into process control at "the higher and less time-critical levels in the industrial automation hierarchy" (Dzung et al., 2005).

Virtualization is central to utility computing, one of the most compelling features driving the adoption of cloud services as a means to increase agility in business (Baer, 2008; Frievald, 2007). Virtualization might be considered an option for SCADA data backup and/or system

recovery solutions with or without incorporating SOA. Virtualization techniques also may hold some promise for encapsulating the static elements of an ICS to provide an abstracted interface to more mutable parts of the network (Naedele, 2007). However, control networks use slow, non-enterprise protocols on field device LANs, have real-time response requirements, and harbor systems with uptime requirements that demand local administration and require extraordinary care to manage the slightest amount of change (Dzung et al., 2005; Naedele; Stouffer et al., 2008). Such systems offer generally poor prospects for taking advantage of the shared-resource aspect of virtualization, particularly if acquired on an outsourced basis.

Virtualization further offers certain security disadvantages. There are data latency concerns in a multi-tenant environment, as well as added risk stemming from the fact that all tenants will share the security level of the least secure tenant. Collecting data into a centralized cloud services database is, in principle, more secure than having it distributed among numerous varied endpoints but also increases the seriousness of a single-point breach. According to the Cloud Security Alliance (2009), another important risk is that virtualization technology adds new attack surfaces in the hypervisor and other management services "but more important is the severe impact virtualization has on network security. Virtual machines now communicate over a hardware backplane, rather than a network. As a result, standard network security controls are blind to this traffic and cannot perform monitoring or in-line blocking" (p. 68), which also greatly complicates compliance assurance.

XML is a cornerstone technology for cloud computing. "Web services implement service oriented architectures (SOA) using open standards based on XML messages and widespread Internet transport protocols such as HTTP" (Kearney, 2005). More specifically, "The technologies that form the foundations of Web services are SOAP, WSDL, and UDDI"

(Sinha et al., 2010, p. 135), all of which are derived from XML. A key virtue of utilizing XML is that "All of these protocols are independent from the machine architecture, the underlying operating system and the programming language" (Mehta, 2009 p. 1).

While many of the advantages of cloud computing derive from the use of XML, numerous issues do as well. XML presents intruders with an increased attack surface (Byres Research, 2007a.) and is vulnerable to a variety of assaults such as XML schema poisoning (Mehta, 2009). None of the three foundational technologies of Web services are free of XML vulnerabilities. SOAP messages may be compromised through XML rewriting attacks (Rahaman et al., 2006) while WSDL scanning and enumeration can locate vulnerabilities by probing the UDDI registry (Breytenbach, 2005; Mehta).

A large set of standards to address Web services security concerns has been evolving for some time (Martino & Bertino, 2006; Sinha et al., 2010). Since XML is text based, encryption standards have been established to preserve the confidentiality of stolen or intercepted material. Message level security measures have also been created since transport level (SSL/TLS) point-to-point security is only partially effective for multi-hop SOAP message transport (Martino & Bertino; Rahaman et al., 2006; Sinha et al.). However, despite the progress that has been made, "there isn't a complete architecture for the Web service security" (Hongzhao, 2010, p. 4). According to Martino and Bertino, the principle problem is that standards for the multiple security protocol levels are being developed by different standards bodies so that compatibility between them is not always assured for every combination of implementation options specified at each level (see also Carlson & Himler, 2005). They further note that resolving such issues into a unified standard can be expected to take years. Accordingly, "The security future thus remains in considerable flux" (Jei et al., 2011, p. 12:24).

Even assuming that Web services cannot be hijacked or corrupted, other security concerns arise simply as the result of the distributed nature of SOA. Since Web services that reside in a variety of security domains may be called into use by a single requestor, significant access control concerns come into play (Balasubramaniam et al., 2009). A complex web of interrelated issues regarding federated identity management, SSO, authorization and authentication must be considered in the effort to grant secure access in a manner that is not unduly cumbersome in terms of requiring multiple sign-ins by users (Carr, 2008; Jei et al., 2011; Martino & Bertino, 2006). According to Delessey and Fernandez (2008), "A methodology to design and build secure SOA-based applications is still lacking" (p. 1).

Execution overhead is a concern with XML as "XML messages processing can require a very large amount of bandwidth with respect to traditional binary messaging protocols" (Martino & Bertino p 22). Along with the issue of expanded message length resulting from XML's verbose text format, "overhead can occur due to the extra CPU time for processing information related to WS-Security, and more time to carry SOAP messages on the network is increased because of additional content to the WS-Security" (Rodrigues et al., 2011, p. 18). Studies indicate a three- to 10-fold penalty for SOA message security processing alone, without considering Web services discovery or binding activities (Rodrigues et al.; Tan et al., 2009). Tan reports result that show SOAP extensions allowing binary rather than text data reduce transmission times by a factor of six, but no performance comparison was found of binary SOAP messaging to the OPC DCOM messaging that is in prevalent use today.

Interoperability is another concern with SOA as exemplified by the incompatibilities between security protocols. At best, messaging overhead is increased by the need for Web services to negotiate over issues like what encryption standard they are jointly able to apply in

order to interoperate (Nagarajan et al., 2006). Far more numerous opportunities to have problems with interoperability due to semantic discrepancies, that is, format incompatibilities at the data or message level. This leads to yet another layer of complexity being added to deliver interoperability by providing data mediation through OWL schemas and Semantic Web services such as WSDL-S that are under development (Nagarajan).

One of the primary means employed in cloud computing to assist with simplifying and managing complexity is the use of an ESB (Flurry & Reinitz, 2007; Tibbling, 2007), always an objective for maximizing system reliability (Stouffer, et al., 2008). As with SOA, the degree of complexity intrinsic to process control networks is greater than that of conventional networks (*Control systems cyber*, 2009). This research did not identify if cloud complexity could be substituted for ICS complexity or would be in addition to it. However, one of the problems in dealing with "an admittedly immature cloud ecosystem" (CSA, 2009, p. 6) is that complexity is increased by the need to continuously fine-tune implementations until best practices have evolved to the point of having a truly stable set of SOA standards.

The adoption of SOA has a strong impact on achieving regulatory compliance for systems. "Assurance accreditation of agile, interconnected IT landscapes is a great challenge, and is currently often cited as one of the showstoppers for the adoption of modern IT architectures (e.g. agile, model-driven, process-led SOA and Cloud) in mission critical domains" (Lang & Schreiner, 2009, p. 13). SOA features such as dynamic binding and unpredictable Web service performance under variable system loads present significant challenges for audit and assurance activities.

4.3 Cloud Computing and the Process Control Environment

The resource limitations of field devices are at odds with the processing requirements of modern networking, particularly when SOA technologies are being utilized. One option to address this dilemma is to employ some type of middleware – hardware, software, or both – to isolate edge clients from high volumes of network activity and complex processing overhead. OPC is the predominant example of such a middleware solution (Dzung et al., 2005) but there are others like the Actor Role Coordinator (ARC) middleware model created by Xiao et al. (2008). Fairly early on, Web services achieved recognition as a form of middleware, albeit for business systems rather than control networks (Kearney, Chapman, Edwards, Gifford, & He, 2004). However, others have since recognized that new architectures incorporating OPC-UA will bring SOA technologies into the realm of middleware for SCADA systems (Tan & Yoo, et al., 2009; Tu et al., 2010).

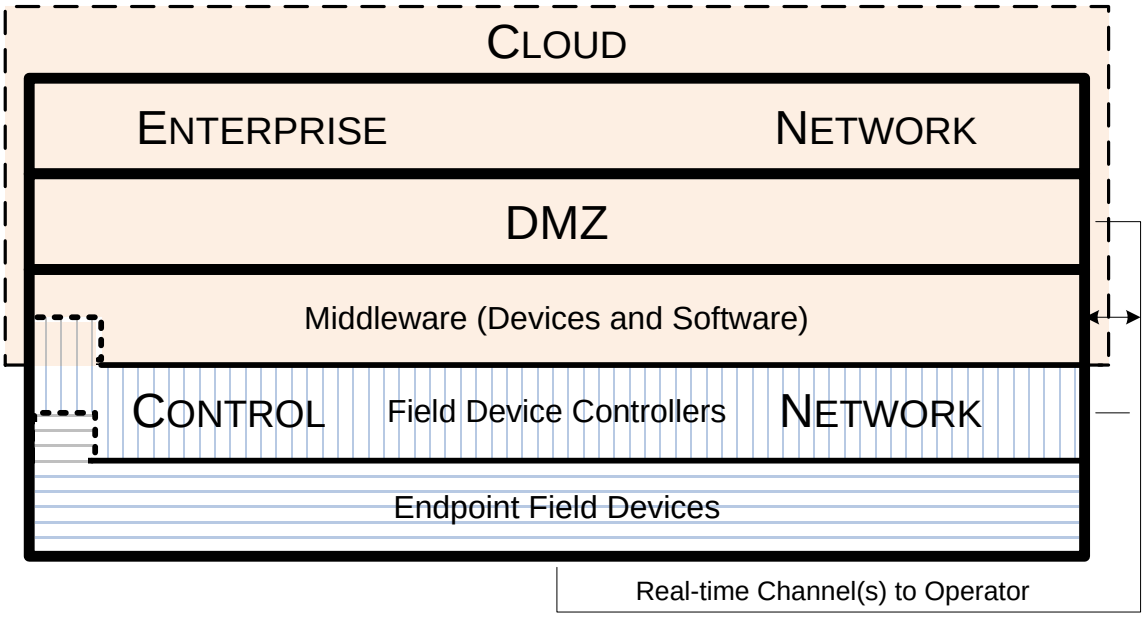


Figure 6: Combining Cloud Computing with a Control Network

Figure 6 illustrates the concept of segregating field hardware from SOA networking and shows the limits of potential cloud penetration into a control network. The cloud could be entirely private but is generalized for illustration as a public or public-private hybrid. The enterprise network is represented as having a contained perimeter within the cloud without revealing any internal complexity or close linkages that might exist with vendors or other outside business partners. In keeping with NIST architectural recommendations for industrial control networks (Stouffer et al., 2008), the DMZ tier is primarily viewed as a location for the Data Historian so that system status and historical operational information may be shared with the enterprise network while minimizing direct exposure of the control network to intrusion from the Internet via the enterprise. The control network is stratified into functional three tiers with some limited overlap between tiers as discussed below.

The corporate network and DMZ may or may not embrace cloud computing. If they do, SOA can be considered as an option for providing a channel to the control network. This channel would extend no deeper than the middleware tier, principally in consideration that Web services could be selected as a means to update the Data Historian in the DMZ. The choice to use a cloud at this level will depend on a number of factors including security policies and response time or latency considerations with respect to the control network.

Some occupants of the middleware tier include control network members that are high up on the process control chain: SCADA computers, HMI or engineering workstations, and any other servers required for configuration, applications, backup, etc. Other occupants with less overall capability would include dedicated inline hardware appliances used to provide additional firewall, encryption, or other security services for systems that lack the resources to incorporate host-based solutions to strengthen their communication channels (CSSP, 2009; Holstein & Diaz,

2006). To the extent that cloud computing extends into a control network, XML appliances designed to relieve the burden of XML overhead and expedite SOAP transactions should be considered middleware (Martino & Bertino, 2006).

The demarcation between tiers is conceptual and not always absolute. In some instances a unit may be deployed that directly controls devices and is powerful enough to straddle across the upper and middle tiers of the control network as indicated by the overlapping corner with the dotted edge. Examples of these systems are OPC servers, MTUs, or SCADA controllers that have the capability to direct devices while also performing middleware duties – including, perhaps, using Web services. However, the great majority of device controllers – PLCs, RTUs and the like – will fit squarely in the middle tier of the control network, primarily engaged in controlling devices while maintaining a modest level of communications with the middleware tier as a conduit for outbound hardware status data and occasional operator input. Field devices at the furthest edge of the control network are represented in the lowest tier of the diagram. Here again, a dotted corner indicates that the idea of a homogeneous tier involves some simplification as IEDs incorporate onboard controllers directly into their devices and functionally overlap both lower tiers.

Besides separating hardware control from an overwhelming amount of network activity, this architectural pattern also ensures that the most deterministic real-time control activities remain with or near the endpoint hardware. System requirements for deterministic performance will vary considerably depending on the processes under management. Extremely tight regulation of temperature and humidity profiles may be critical in a pharmaceutical manufacturing process whereas the tolerable variance for a switch change in a railroad track is likely to be measured in seconds.

Beyond real-time device control there is a general system requirement to transfer alarms and other rapid response signals to and from the operator with a minimum of delay (Dzung et al., 2005; Tu et al., 2010). Figure 6 represents the need to address this requirement through inclusion of the bidirectional Real-time Channel(s) to Operator in a manner similar to the unidirectional path proposal by Tan, Yoo, and Yi (2009). Separate back channel provisioning for the most urgent device communications promotes consideration of SOA networking for the most modern systems in the middleware tier, such as the SCADA controller, HMI, and other servers and workstations.

Chapter 5 – Conclusions

The objective of this project is to determine the present viability of a cloud-based architecture as an alternative to conventional networking approaches for implementing SCADA systems. To this end, the prospects for a cloud solution were evaluated against conventional approaches.

There are *a priori* limits to the degree of improvement cloud technologies might bring to control systems. Simply modernizing distributed processing technologies using SOA will not address many of the vulnerabilities in that presently exist in control networks due to such issues as weak security policies, implementations, and enforcement; under-resourced field devices; or the need for staff with combined expertise in both IT and process control systems.

The impact of alternative technologies on SCADA must be considered as well with respect to implementations that are presently in place, OPC solutions in particular, which have been developed for most control systems of significant complexity. Simply replacing an existing functionality may be considered worthwhile if the replacement provides some form of maintenance advantage such as technological forward-compatibility. This would, however, be a minimum useful gain and some operational performance benefit would almost certainly be required to financially justify a platform replacement.

Superficially, SCADA and cloud computing share a certain degree of alignment by virtue of being intrinsically distributed systems. Both have common security needs with regard to accessing geographically dispersed resources. Further, SOA communication standards to implement sharing of on-demand Web services are evolving rich protocols for managing the authentication and related security issues of every transaction undertaken within a cloud, potentially addressing a key weakness of many control network implementations.

Despite their obvious points of alignment, SCADA and cloud computing are not fundamentally compatible technologies. This research has identified a number of factors that make the present state of cloud computing generally unacceptable for use in SCADA systems. These factors can be grouped into four interrelated areas: reliability and availability, compliance, security, and complexity.

Availability is closely related to reliability as the process output must be continuously available (e.g., electricity) while monitoring and control systems must always be available as well to avoid unregulated activity or automated safety shutdowns due to control access failures. Control system reliability further imposes a strong requirement for deterministic system performance to assure the correct uninterrupted execution of time-sensitive processes.

Cloud computing has been developed so far only to meet enterprise networks needs where a best effort response is sufficient as opposed to one in real-time. Even with conventional networking, limited-capacity process control field devices must generally be buffered from high bandwidth activities like TCP/IP networking to avoid being overwhelmed, causing a disruption of safe unimpeded operation. The increased message length and additional security overhead that comes with SOA only serve to exaggerate this isolation requirement.

Control networks are often responsible for critical or potentially hazardous processes and so must meet reliability criteria that are not considerations for enterprise systems. As a consequence, SCADA systems also comply with a broader set of regulatory mandates than is typical for business systems. Cloud computing properties such as the composability of Web services, late system binding, and unpredictable execution under different system loads confound the process of compliance assurance (Balasubramaniam, 2009). This presents regulatory risks, if not operational ones.

Not only is security closely intertwined with compliance, it is accorded an extra measure of gravitas when dealing with process control where a breach may result in environmental and/or human health and safety may being seriously affected. There is strong evidence that SOA is harder to secure overall than more conventional network architectures due to the increased attack surfaces presented by XML-based Web services compounded by its complex, multi-component structure. This security risk is not acceptable for SCADA systems because of the dangers concomitant with disabled or maliciously directed hardware operations. While SOA-driven developments in access and authentication solutions would remediate one of the most frequent and severe security weaknesses of control networks, SOA is not necessarily a prerequisite to their adoption. The increasing wealth of transaction security measures currently being nurtured within the cloud community are undermined by the lack of a set of standards to fully assure secure interoperability.

Finally, complexity is a concern as it not only adds a set of issues in its own right but it makes correct implementation more difficult, impedes compliance verification of consistent and reliable operation, and increases the number of potential vulnerabilities that must be secured against. Control networks are intrinsically more complex than enterprise networks so their designs are simplified wherever possible in an effort to remove risk of the unforeseen and enhance reliable operations. SOA is more complex than conventional network architecture making its use contrary to control network design practices, particularly since the enterprise computing benefits of SOA complexity do not accrue to control networks. Complexity is one reason that cloud computing is still undergoing significant evolution, which is further a negative for control networks where patches and other updates are difficult to administer without risking the underlying stability of process operations.

It is functionally possible for SOA to be utilized for some aspects of SCADA system as demonstrated by the tiered network architecture presented herein. However, the degree of such inclusion does not at this point constitute a fundamental improvement in implementing distributed control networks. Nor does the mere fact that an SOA implementation can be made to function necessarily recommend for its adoption when risk considerations are factored in. At this time it would take a significant effort to avoid cloud implementations that were both more complex and poorer performing than the solutions they replaced. The security and regulatory compliance of XML-based Web service technologies need to be assured. At the same time, field devices must be upgraded with the capacity to accommodate SOA processing overhead without diminishing the performance reliability of endpoint hardware. Given the slow replacement cycle of control devices, edge clients will remain incapable of employing Web services for some time, thus preventing their direct participation in cloud computing for the foreseeable future.

Much of the research required to advance SOA as an architectural choice for control networks aligns with work already underway in the enterprise networking realm. The advent of cloud computing has turned the IT spotlight on longstanding ICS issues regarding identity management and access control for distributed processing systems. Even without migrating to the cloud, SCADA systems ultimately stand to benefit from accelerated efforts in this area. Similarly, improved provisions in SOA to enable compliance assurance will not only be a benefit to ICS but will be required before highly regulated process operations can embrace the cloud.

The most immediate need for research on cloud computing in process management concerns the ability of SOA technologies to meet real-time control performance requirements. Knowledge in this area would be furthered by research designed to compare the throughput of binary SOAP messaging to that of DCOM communications in current use.

The prospective instability of cloud implementations due to the continued necessary evolution of interoperability features presents at least a near-term negative for process control. This is due to the fact that any need to patch or upgrade control systems represents a threat to reliable continuous operations. Here the likely opportunity is to identify and confirm a streamlined subset of SOA standards that may already be sufficient for adoption by SCADA systems without the expectation of requiring substantial future refinements – or to clearly identify what the remaining unmet requirements are. Focus areas for such research include simplifying SOA implementations and achieving performance optimization through the use of such measures as binary SOAP messages, restricted preset security options to avoid negotiation overhead, and a indentifying a Web services deployment model that precludes multi-hop transmissions to avoid onerous additional message level security processing.

Process control systems are already complex and will become even more so as legacy design deficiencies in security are addressed. The ultimate question is whether or not SOA complexity may be exchanged for, rather than added to, existing SCADA complexities in order to deliver an improved solution overall.

References

Adomavicius, G., Bockstedt, J. C., Gupta, A., & Kaufman, R. J. (2008). Making sense of technology trends in the information technology landscape: A design science approach. *MIS Quarterly, 32*(4), 779-809. Retrieved from http://web.ebscohost.com.dml.regis.edu/ehost/pdfviewer/pdfviewer?hid=17&sid=708d9634-df1e-4c10-a40c-72c5d964444e%40sessionmgr10&vid=3

Alam, M., Hafner, M., & Breau, R. (2008). Constraint based role based access control in the SECTET-framework. *Journal of Computer Security, 16*(2), 223-260. Retrieved from http://web.ebscohost.com.dml.regis.edu/ehost/pdfviewer/pdfviewer?hid=104&sid=fcd2202f-bc91-4b7d-a49b-cf0212d08b04%40sessionmgr114&vid=1

Badger, L., & Grance, T. (2010). *Standards acceleration to jumpstart adoption of cloud computing (SAJACC)* [PowerPoint slides]. Retrieved November 5, 2010, from NIST ITL - Computer Security Division website: http://csrc.nist.gov/groups/SNS/cloud-computing/documents/forumworkshop-may2010/nist_cloud_computing_forum-badger_grance.pdf

Baer, T. (2008, March 8). *SOA benefits: too much reuse of reuse?* Retrieved December 2, 2010, from http://www.theregister.co.uk/2008/03/08/soa_reuse_repetition/

Balasubramaniam, S., Lewis, G. A., Morris, E., Simanta, S., & Smith, D. B. (2009). Challenges for assuring quality of service in a service-oriented environment. In (Ed.), *PESOS '09 Proceedings of the 2009 ICSE Workshop on Principles of Engineering Service Oriented Systems* (pp. 103-106). Vancouver, Canada: IEEE. Retrieved November 17, 2010, from http://delivery.acm.org.dml.regis.edu/10.1145/1570000/1564736/05068829.pdf?key1=1564736&key2=3099362921&coll=DL&dl=ACM&CFID=2581268&CFTOKEN=77746308

Blackwell, C. (2008). A multi-layered security architecture for modelling complex systems. In F. Sheldon, A. Krings, R. Ambercromberie & A. Mili (Eds.), *Proceedings of the 4th Annual Workshop on Cyber Security and Information Intelligence Research: Developing Strategies to Meet the Cyber Security and Information Intelligence Challenges Ahead: Vol.288* (p. Article 35). New York, NY: ACM. doi:http://doi.acm.org/10.1145/1413140.1413180

Breytenbach, C. (2005). Introduction to assessing and securing web services. In H.S .Venter, J.H.P. Eloff, L. Labuschagne, & M.M. Eloff (Eds.), *ISSA 2005 New Knowledge Today Conference* (p.). Pretoria, South Africa: University of Pretoria. Retrieved March 10, 2011, from University of Pretoria website: http://icsa.cs.up.ac.za/issa/2005/Proceedings/Full/058_Article.pdf

Byres, E., & Hoffman, D. (2004). The myths and facts behind cyber security risks for industrial control systems. In, *VDE Kongress, 2004* (pp. 1-7). Berlin, Germany: VDE Verlag. Retrieved October 3, 2010, from http://www.isa.org/CustomSource/ISA/Div_PDFs/PDF_News/Glss_2.pdf

Byres, E., Karsch, J., Carter, J., & Savage, K. (2005). *NISCC good practice guide on firewall deployment for SCADA and process control networks* (pp 1-42). Retrieved June 7, 2010, from National Infrastructure Security Co-ordination Centre (NISCC) website: http://www.cpni.gov.uk/docs/re-20050223-00157.pdf

Byres Research (2007c). *OPC security whitepaper #3: Hardening guidelines for OPC hosts* (Version 1-3c). Retrieved June 17, 2010, from http://www.pacontrol.com/download/OPC-Security-WP3.pdf

Byres Research (2007a). *OPC security white paper #1: understanding OPC and how it is deployed* (Version 1-3b). Retrieved June 17, 2010, from http://www.pacontrol.com/download/OPC-Security-WP1.pdf

Byres Research (2007b). *OPC security white paper #2: OPC exposed* (Version 1-3c). Retrieved June 17, 2010, from http://www.pacontrol.com/download/OPC-Security-WP2.pdf

Cagalaban, G., Kim, T., & Kim, S. (2008). Improving SCADA control systems security with software vulnerability analysis. In Niola, V. Quartieri, J.; Neri, F.; Caballero, A.; F. Rivas-Echevererria & N. Mastorakis (Eds.), *12th WSEAS International Conference on Automatic Control, Modelling & Simulation: Vol.12th* (pp. 409-414). Stevens Point, WI: WSEAS Press. Retrieved September 8, 2010, from http://www.wseas.us/e-library/conferences/2010/Catania/ACMOS/ACMOS-69.pdf

Carlson, B., & Himler, A. (2005). Turning application security inside out: Security for service-oriented architectures (SOAs). *Information Systems Security*, *14*(4), 27-35. Retrieved from http://web.ebscohost.com.dml.regis.edu/ehost/pdfviewer/pdfviewer?hid=104&sid=71a02598-a73c-4627-a899-a733a80cd2bc%40sessionmgr113&vid=1

Carr, J. (2008, July). The Fword. *SC Magazine: for IT Security Professionals*, (July 2008), 34-37. Retrieved November 17, 2010, from http://web.ebscohost.com.dml.regis.edu/ehost/pdfviewer/pdfviewer?hid=104&sid=501f5808-ffb2-4882-987d-9878f2d052f6%40sessionmgr104&vid=6

Chatarji, J. (2004, October 13). *Introduction to service oriented architecture (SOA) - Benefits of SOA*. Retrieved December 2, 2010, from Introduction to Service Oriented Architecture (SOA) - Benefits of SOA

Control systems cyber security—the current status of cyber security of critical infrastructures: Hearings before the Senate Committee on Commerce, Science, and Transportation U.S. Senate, 111th Cong. (2009) (testimony of Joseph M. Weiss PE, CISM). Retrieved March 10, 2011, from: http://cimic.rutgers.edu/WS-CFP/Joe-Weiss.pdf

Control Systems Security Program (2009). *Recommended practice: Improving industrial control systems cybersecurity with defense-in-depth strategies*. Retrieved August 11, 2010, from Homeland Security website: http://csrp.inl.gov/Documents/Defense_in_Depth_Oct09.pdf

Centre for the Protection of National Infrastructure (2008a). *Good practice guide – process control and SCADA security*. Retrieved June 7, 2010, from Centre for the Protection of National Infrastructure website: http://www.cpni.gov.uk/Docs/Overview_of_Process_Control_and_SCADA_Security.pdf

Cloud Security Alliance (2009). *Security guidance for critical areas of focus in cloud computing v2.1*. Retrieved November 2, 2010, from Cloud Security Alliance website: http://www.cloudsecurityalliance.org/csaguide.pdf

Cloud Security Alliance (2010). *Domain 12: Guidance for identity and access management v2.1*. Retrieved November 2, 2010, from Cloud Security Alliance website: http://www.cloudsecurityalliance.org/guidance/csaguide-dom12-v2.10.pdf

Dawson, R., Boyd, C., Dawson, E., & Nieto, J. (2006). SKMA: a key management architecture for SCADA systems. In R. Buyya, T. Ma, R. Safavi-Naini, C. Steketee & W. Susilo (Eds.), *Fourth Australasian Information Security: Vol.54* (pp. 183-192). Darlinghurst, Australia: Australian Computer Society, Inc. Retrieved August 11, 2010, from http://delivery.acm.org.dml.regis.edu/10.1145/1160000/1151850/p183-dawson.pdf?key1=1151850&key2=8905034821&coll=GUIDE&dl=ACM&CFID=104214748&CFTOKEN=38883115

Delessey, N. A., & Fernandez, E. B. (2008). A pattern-driven security process for SOA applications. In R. L. Wainright & H. Haddad (Eds.), *The 2008 ACM Symposium on Applied Computing (SAC)* (pp. 2226-2227). Fortalezza, Brazil: ACM. Retrieved November 17, 2010, from http://delivery.acm.org.dml.regis.edu/10.1145/1370000/1364217/p2226-delessy.pdf?key1=1364217&key2=9725262921&coll=DL&dl=ACM&CFID=2581268&CFTOKEN=77746308

Duggan, D., Berg, M., Dillinger, J., & Stamp, J. (2005). *Penetration testing of industrial control systems* (SAND2005-2846P). Retrieved March 10, 2011, from Sandia National Laboratories website: http://www.sandia.gov/ccss/documents/sand_2005_2846p.pdf

Dzung, D., Naedele, M., Von Hoff, T. P., & Crevatin, M. (2005). Security for industrial communication systems. *Proceedings of the IEEE, 93*, 1152-1177. Retrieved March 10, 2011, from http://www.tik.ee.ethz.ch/~naedele/ProcIEEE05.pdf

Fabro, M., & Cornelius, Eric (2008). *Recommended practice: Creating cyber forensics plans for control systems* (). Retrieved June 17, 2010, from Department of Homeland Security website: http://csrp.inl.gov/Documents/Forensics_RP.pdf

Fabro, M., & Maio, V. (2007). *Using operational security (OPSEC) to support a cyber security culture in control systems environments* (Version 1.0 Draft). Retrieved September 26, 2010, from Idaho National Laboratory website: http://www.us-cert.gov/control_systems/practices/documents/OpSec%20Rec%20Practice.pdf

Falliere, N. (2010, August 6). *Stuxnet introduces the first known rootkit for industrial control systems*. Retrieved September 2, 2010, from http://www.symantec.com/connect/blogs/stuxnet-introduces-first-known-rootkit-scada-devices

Farkas, C., & Huhns, M. N. (2008). Securing enterprise applications: Service-oriented security (SOS). In *2008 10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services* (pp. 428-431). Washington, DC: IEEE Computer Society. doi:http://doi.ieeecomputersociety.org/10.1109/CECandEEE.2008.151

Fenrich, K. (2007, June). *Securing your control system*. Paper presented at the 50th Annual ISA. POWID Symposium/17th ISA POWID/EPRI Controls & Instrumentation Conference. Retrieved from http://www.controlglobal.com/wp_downloads/pdf/abb_secure_control_sys.pdf

Fernandez, E., Wu, J., Larrondo-Petrie, M., & Shao, Y. (2009). On building secure SCADA systems using security patterns. In F. Sheldon, G. Peterson, A. Krings, R. Ambercrombie & A. Mili (Eds.), *5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies* (Article 17). New York, NY: ACM. Retrieved September 12, 2010, from http://delivery.acm.org.dml.regis.edu/10.1145/1560000/1558627/a17-fernandez-slides.pdf?key1=1558627&key2=5474034821&coll=GUIDE&dl=ACM&CFID=104214748&CFTOKEN=38883115

Fernandez, J., & Fernandez, A. (2005). SCADA systems: Vulnerabilities and remediation. *Journal of Computing Sciences in Colleges, 20*(4), 160-168. Retrieved from http://delivery.acm.org.dml.regis.edu/10.1145/1050000/1047872/p160-fernandez.pdf?key1=1047872&key2=4005034821&coll=GUIDE&dl=ACM&CFID=104214748&CFTOKEN=38883115

Fischer, T. (2007). *MPLS security overview* (). Retrieved March 10, 2011, from Information Risk Management Plc. website: http://www.tmplab.org/wiki/images/e/ef/MPLS_Security_Overview.pdf

Flurry, G. (2007). *Exploring the Enterprise Service Bus, Part 1: Discover how an ESB can help you meet the requirements for your SOA solution*. Retrieved October 10, 2010, from http://www.ibm.com/developerworks/webservices/library/ar-esbpat1/index.html

Flurry, G., & Reinitz, R. (2007). *Exploring the enterprise service bus, Part 2: Why the ESB is a fundamental part of SOA* (). Retrieved October 10, 2010, from IBM website: http://www.ibm.com/developerworks/architecture/library/ar-esbpat2/

Freivald, J. (2007). The business benefits of SOA implementation. *InfoManagement Direct*. Retrieved October 10, 2010, from http://www.information-management.com/infodirect/2008_57/10000665-1.html

Lawton, G. (2010, July 6). Three tips for choosing an ESB [Web log post]. Retrieved from http://searchsoa.techtarget.com/tip/0,289483,sid26_gci1361026_mem1,00.html

Giani, A., Karsai, G., Roosta, T., Shah, A., Sinopoli, B., & Wiley, J. (2008). A testbed for secure and robust SCADA systems. In, *14th IEEE Real-time and Embedded Technology and Applications Symposium (RTAS'08) WIP Session* (Article 4). New York, NY: ACM. Retrieved September 12, 2010, from http://delivery.acm.org.dml.regis.edu/10.1145/1400000/1399587/p4-giani.pdf?key1=1399587&key2=8494034821&coll=GUIDE&dl=ACM&CFID=104214748&CFTOKEN=38883115

Graham, R. D. (2007, March 22). New SCADA vulns [Web log post]. Retrieved from http://erratasec.blogspot.com/2007/03/new-scada-vulns.html

Graham, R., & Maynor, D. (2006). *SCADA security and terrorism: We're not crying wolf!* [PowerPoint slides]. Retrieved September 8, 2010, from Black Hat website: http://www.blackhat.com/presentations/bh-federal-06/BH-Fed-06-Maynor-Graham-up.pdf

Hahn, A., Kregel, B., Govindarasu, M., Fitzpatrick, J., Adnan, Rafi, Sridhar, S., Higdon, M. (2010). Development of the PowerCyber SCADA security testbed. In F. Sheldon, S. Prowel, A. Abercrombie & A. Krings (Eds.), *Sixth Annual Workshop on Cyber Security and Information Intelligence Research* (p. Article 21). New York, NY: ACM. Retrieved September 27, 2010, from http://portal.acm.org.dml.regis.edu/toc.cfm?id=1852666&type=proceeding&coll=Portal&dl=GUIDE&CFID=106348341&CFTOKEN=63643112

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in information systems research. *MIS Quarterly*, *28*(1), 75-105. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.126.5174&rep=rep1&type=pdf

Hevnor, A. R. (2007). A three cycle view of design science research. *Scandinavian Journal of Information Systems*, *19*(2), 87-92. Retrieved from http://community.mis.temple.edu/seminars/files/2009/10/Hevner-SJIS.pdf

Hinton, H., Hondo, M., & Hutchison, B. (2005). *Security patterns within a service-oriented architecture* (). Retrieved October 10, 2010, from http://s3.amazonaws.com/ppt-download/security-patterns-within-a-serviceoriented-architecture3304.pdf?Signature=6m%2F5xAB6HfS7rsRod%2BE6vlGuPJM%3D&Expires=1295990214&AWSAccessKeyId=AKIAJLJT267DEGKZDHEQ

Holstein, D., & Diaz, J. (2006). Cyber security management for utility operations. In, *Proceedings of the 39th Annual Hawaii International Conference on System Sciences* (p. 241c). Washington, DC: IEEE. Retrieved September 12, 2010, from http://csdl2.computer.org.dml.regis.edu/comp/proceedings/hicss/2006/2507/10/25071002 41c.pdf

Hongzhao, K. (2010). *A study on the security mechanism for web services*. Paper presented at the World Congress on Engineering and Computer Science 2010. Retrieved from http://www.iaeng.org/publication/WCECS2010/WCECS2010_pp93-96.pdf

IAONA (2003). *The IAONA handbook for network security* (DRAFT Version v0.4). Retrieved September 30, 2010, from IAONA website: http://www.innominate.com/images/stories/documents/publikationen/2003/public_iaona-security.pdf

IBM Internet Security Systems. (2007). *A strategic approach to protecting SCADA and process control systems* (pp. 1-10). Retrieved September 8, 2010, from http://documents.iss.net/whitepapers/SCADA.pdf

Industrial Control Systems Cyber Emergency Response Team (2010). *Control systems analysis report - SSH brute-force scanning and attacks* (pp. 1-4). Retrieved September 12, 2010, from United States Computer Emergency Readiness Team (US-CERT) website: Department of Homeland Security website: http://www.us-cert.gov/control_systems/pdf/ICS-CERT%20CSAR-SSH%20SCANNING.pdf

Jie, W., Arshod, J., Sinnot, R., Townend, P., & Lei, Z. (2011). A review of grid authentication and authorization technologies and support for federated access control. *ACM Computing Surveys, 43*(2), 12:1-12:26. Retrieved from http://delivery.acm.org.dml.regis.edu/10.1145/1890000/1883619/a12-jie.pdf?key1=1883619&key2=1360806031&coll=DL&dl=ACM&ip=207.93.211.102&CFID=24563287&CFTOKEN=53164629

Kearney, P. (2005). Message level security for web services. *Information Security Technical Report, 10*(1), 41-50. doi:doi:10.1016/j.istr.2004.11.003

Kearney, P., Chapman, J., Edwards, N., Gifford, M., & He, L. (2004). An overview of web services security. *BT Technology Journal, 22*(1), 27-42. Retrieved from http://mmlab.ceid.upatras.gr/courses/ais_site/files/3%5CAn%20Overview%20of%20Web%20Services%20Security.pdf

Landau, S. (2008). Security and privacy landscape in emerging technologies. *IEEE Security and Privacy, 6*(4), 74-77. doi: 10.1109/MSP.2008.95

Lang, U., & Schreiner, R. (2009). Model driven security accreditation (MDSA) for agile, interconnected IT landscapes. In S. Jajodia & M. Kudo (Eds.), *The 1st ACM Workshop on Information Security Governance: Vol.1* (pp. 13-22). New York, New York: ACM. Retrieved November 17, 2010, from http://delivery.acm.org.dml.regis.edu/10.1145/1660000/1655173/p13-lang.pdf?key1=1655173&key2=6566262921&coll=DL&dl=ACM&CFID=2581268&CFTOKEN=77746308

Leischner, G., & Tews, C. (2007). *Security through VLAN segmentation: Isolating and securing critical assets without loss of usability*. Paper presented at the 9th Annual Western Power Delivery Automation Conference. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.125.5869&rep=rep1&type=pdf

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems, 15*, 251-266. Retrieved June 14, 2010, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.111.5734&rep=rep1&type=pdf

Martino, L. D., & Bertino, E. (2006). *Security for web services - standards and research issues* (CERIAS Tech Report 2006-34). Retrieved March 12, 2011, from Center for Education and Research in Information Assurance and Security website: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.159.264&rep=rep1&type=pdf

McQueen, M. A., Boyer, W. F., Flynn, M. S., & Beitel, G. A. (2006). Quantitative cyber risk reduction estimation methodology for a small SCADA control system. In, *39th Annual Hawaii International Conference on System Sciences: Vol.9* (pp. 1-11). Washington, DC: IEEE Computer Society. Retrieved September 8, 2010, from http://csdl2.computer.org.dml.regis.edu/comp/proceedings/hicss/2006/2507/09/250790226.pdf

Mehta, D. M. (2009). *Jeopardy in Web 2.0 - the next generation web* (). Retrieved March 10, 2011, from OWASP website: https://www.owasp.org/images/b/b6/Jeopardy_in_Web_2.0_-_The_Next_Generation_Web_Applications.pdf

Mell, P., & Grance, T. (2009). *Effectively and securely using the cloud computing paradigm* [PowerPoint slides]. Retrieved November 2, 2010, from FISSEA website: http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-computing-v26.ppt

Miller, A. (2005). Trends in process control systems security. *IEEE Security and Privacy, 3*(5), 57-60. doi:10.1109/MSP.2005.136

Naedele, M. (2007). Addressing IT security for critical control systems. In, *40th Annual Hawaii International Conference on System Sciences* (pp. 1-9). Washington, DC: IEEE Computer Society. Retrieved September 12, 2010, from http://csdl2.computer.org.dml.regis.edu/comp/proceedings/hicss/2007/2755/00/27550115a.pdf

Naedele, M., & Dzung, D. (2005). *Industrial information system security - IT security in industrial plants - an introduction* (9AKK100580A1770). Retrieved October 3, 2010, from http://www05.abb.com/global/scot/scot296.nsf/veritydisplay/1c3f01fb6db52b9fc12571350078dd17/$File/3BUS094319_en_ABB_Review_Security_2005_1.pdf

Nagarajan, M., Verma, K., Sheth, A. P., Miller, J., & Lathem, J. (2006). Semantic interoperability of web services – challenges and experiences. In (Ed.), *ICWS '06 Proceedings of the IEEE International Conference on Web Services* (pp. 373-382). Washington, DC: IEEE Computer Society. Retrieved March 10, 2011, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.5339&rep=rep1&type=pdf

National SCADA Test Bed (2010). *NSTB assessments summary report - common industrial control system cyber security weaknesses* (INL/EXT-10-18381). Retrieved September 12, 2010, from National SCADA Test Bed, Idaho National Laboratory website: U.S. Department of Energy website: http://www.fas.org/sgp/eprint/nstb.pdf

Offermann, P., Levina, O., Schönherr, M., & Bub, U. (2009). Outline of a Design Science Research Process. In (Ed.), *DESRIST '09 4th International Conference on Design Science Research in Information Systems and Technology* (p. Article 7). New York, NY: ACM. doi:10.1145/1555619.1555629

Patel, S. C., Bhatt, G. D., & Graham, H. J. (2009). Improving the cyber security of SCADA communication networks. *Communications of the ACM, 52*(7), 139-142. Retrieved from http://delivery.acm.org.dml.regis.edu/10.1145/1540000/1538820/p139-patel.pdf?key1=1538820&key2=7489034821&coll=Portal&dl=ACM&CFID=104215562&CFTOKEN=78224433

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chaterjee, S. (2008). A design science research methodology for information systems research. *Journal of Management Information Systems, 24*(3), 45-77. doi:10.2753/MIS0742-1222240302

Permann, M., Hammer, J., Lee, K., & Rohde, K. (2006). Mitigations for security vulnerabilities found in control system networks. In, *16th Annual Joint ISA POWID/EPRI Controls and Instrumentation Conference;* (pp. 1-12). Research Triangle, NC: ISA - The Instrumentation, Systems and Automation Society. Retrieved September 29, 2010, from http://csrp.inl.gov/Documents/MitigationsForVulnerabilitiesCSNetsISA.pdf

Pries-Heje, J., Baskerville, R., & Venable, J. (2008). Strategies for design science research evaluation. In V. Viashnavi & R. Baskerville (Eds.), *Proceedings of the 16th European Conference on Information Systems: Vol.16* (pp. 255-266). Galway, Ireland: National University of Ireland. Retrieved June 14, 2010, from http://is2.lse.ac.uk/asp/aspecis/20080023.pdf

Purao, S., Baldwin, C., Hevner, A., Storey, V. C., Pries-Heje, J., Smith, B., Zhu, Y. (2008). The sciences of design: Observations on an emerging field. *Communications of the Association for Information Systems, 23,* 523-546. Retrieved from http://www.hbs.edu/research/pdf/09-056.pdf

Rahaman, M. A., Schaad, A., & Rits, M. (2006). Towards secure SOAP message exchange in a SOA. In (Ed.), *SWS '06 Proceedings of the 3rd ACM workshop on Secure web services: Vol.3rd* (pp. 77-84). New York, NY: ACM. Retrieved March 10, 2011, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.100.6358&rep=rep1&type=pdf

Roch, E. (2006, March 7). *SOA benefits, challenges and risk mitigation*. Retrieved October 10, 2010, from http://it.toolbox.com/blogs/the-soa-blog/soa-benefits-challenges-and-risk-mitigation-8075

Rodrigues, D., Estrella, J. C., & Branco, K. R. (2011). Analysis of security and performance aspects in service-oriented architectures. *International Journal of Security and Its Applications, 5*(1), 13-30. Retrieved from http://www.sersc.org/journals/IJSIA/vol5_no1_2011/2.pdf

Rosenberg, J., & Remy, D. (2004). Basic concepts of web services security. In *Securing Web Services with WS-Security: Demystifying WS-Security, WS-Policy, SAML, XML Signature, and XML Encryption* (1st ed., pp. 1-9). Indianapolis, IN: SAMS. Retrieved March 12, 2011, from http://www.devshed.com/c/a/Security/Basic-Concepts-of-Web-Services-Security/

Ross, R., Katzke, S., Johnson, A., Swanson, M., Stoneburner, G., & Rogers, G. (2007). *Recommend security controls for federal information systems: Special publication 800-53* (Rev. 2). Gaithersburg, MD: National Institute of Standards and Technology. Retrieved July 1, 2010, from http://csrc.nist.gov/publications/nistpubs/800-53-Rev2/sp800-53-rev2-final.pdf

Scarfone, K., & Hoffman, P. (2009). *Guidelines on firewalls and firewall policy: Special publication 800-41*(Rev. 1). Gaithersburg, MD: National Institute of Standards and Technology. Retrieved July 1, 2010, from http://csrc.nist.gov/publications/nistpubs/800-41-Rev1/sp800-41-rev1.pdf

Selvage, M., Flurry, G., Sauter, G., & Lane, E. (2008). *Exploring the enterprise service bus, part 3: Four approaches to implementing a canonical message model in an ESB* (). Retrieved October 10, 2010, from IBM website: http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ar-esbpat3/ar-esbpat3-pdf.pdf

Sinha, S., Sinha, S. K., & Purkayastha, B. S. (2010). Security issues in web services : A review and development approach of research agenda. *Assam University Journal of Science & Technology : Physical Sciences and Technology, 5*(2), 134-140. Retrieved from http://www.inflibnet.ac.in/ojs/index.php/AUJSAT/article/viewFile/101/99

SOATutorial.net. (2010, June 6). *Common tangible benefits of SOA*. Retrieved October 10, 2010, from http://www.soatutorial.net/common-tangible-benefits-of-soa/

Stouffer, K., Falco, J., & Scarfone, K. (2008). *Guide to industrial control systems (ICS) security: Special publication 800-82* (Final Public Draft). Gaithersburg, MD: National Institute of Standards and Technology. Retrieved July 26, 2010, from http://csrc.nist.gov/publications/drafts/800-82/draft_sp800-82-fpd.pdf

Tan, V. V., Yoo, D. S., & Yi, M. J. (2009). A SOA-based framework for building monitoring and control software systems. In D. Huang, K. Jo, H. Lee, H. Kang & V. Bevilacqua (Eds.), *Proceedings of the 5th international conference on Emerging intelligent computing technology and applications* (pp. 1013-1027). Berlin, Germany: Springer-Verlag. Retrieved August 11, 2010, from http://delivery.acm.org.dml.regis.edu/10.1145/1790000/1788286/p1013-vantan.pdf?key1=1788286&key2=2130134821&coll=Portal&dl=ACM&CFID=104215562&CFTOKEN=78224433

Taylor, C., Krings, A., & Alves-Foss, J. (2002). Risk analysis and probabilistic survivability assessment (RAPSA): An assessment approach for power substation hardening. In, *ACM Workshop on Scientific Aspects of Cyber Terrorism (SACT)* (p. 9). New York, NY: ACM. doi:10.1.1.80.8323.pdf

Tibbling, M. (2010). *Using an ESB to simplify the complexity of SOA*. Retrieved December 3, 2010, from http://searchsoa.techtarget.com/tip/Using-an-ESB-to-simplify-the-complexity-of-SOA

U.S Department of Energy (2002). *21 steps to improve cyber security of SCADA networks*. Retrieved August 11, 2010, fromhttp://www.oe.netl.doe.gov/docs/prepare/21stepsbooklet.pdf

Udassin, E. (2008a). Control system attack vectors and examples - Field site and corporate network. In D. Peterson (Ed.), *SCADA Security Scientific Symposium* (pp. 1-11). Sunrise, FL: Digital Bond Press. Retrieved October 2, 2010, from http://www.c4-security.com/SCADA%20Security%20-%20Attack%20Vectors.pdf

Udassin, E. (2008b). *General Electric grid malware design* [PowerPoint slides]. Retrieved September 12, 2010, from http://www.c4-security.com/SCADA%20Security%20-%20Generic%20Electric%20Grid%20Malware%20Design%20-%20SyScan08.pps

Vaas, L. (23). *Hole found in protocol handling vital national infrastructure*. Retrieved September 25, 2010, from http://www.eweek.com/c/a/Security/Hole-Found-in-Protocol-Handling-Vital-National-Infrastructure/

Verendel, V. (2009). Quantified security is a weak hypothesis. In R. Ford, M. H. Heydari & A. Somayaji (Eds.), *New Security Paradigms Workshop 2009* (pp. 37-49). New York, NY: Association for Computing Machinery. Retrieved September 12, 2010, from http://delivery.acm.org.dml.regis.edu/10.1145/1720000/1719036/p37-verendel.pdf?key1=1719036&key2=8625034821&coll=Portal&dl=ACM&CFID=104215562&CFTOKEN=78224433

Xiao, K., Ren, S., & Kwiat, K. (2008). Retrofitting cyber physical systems for survivability through external coordination. In, *41st Annual Hawaii International Conference on System Sciences* (pp. 1-9). Washington, DC: IEEE Computer Society. Retrieved September 12, 2010, from http://csdl2.computer.org.dml.regis.edu/comp/proceedings/hicss/2008/3075/00/30750465.pdf

Appendix A – Time Line of Well-known Industrial Control Security Incidents

- *June 2010:* Belorussian security company VirusBlokAda discovered the Stuxnet worm, the first-ever rootkit for exploiting a SCADA system. Indications are that the earliest versions of this worm were deployed up to a year before it was discovered. Stuxnet required a sophisticated understanding of both Windows and industrial control software to create. The malware is not only capable of stealing code and design projects stored on Windows systems, it can locate and exploit Siemens SCADA software on these machines to download and hide code on a variety of PLCs that directly control industrial processes.
- *January 2008:* Israeli SCADA security firm C4 released documentation that revealed vulnerabilities present in General Electric SCADA systems commonly used at nuclear facilities around the world. This is the first time an exploit was demonstrated to definitely enable the remote takeover of a SCADA system.
- *March 2007:* Research firm Neutralbit documented the discovery of a significant security hole in the popular NETxAutomation OPC server that could crash or potentially allow the takeover of systems controlling oil refineries, dams, railroads, and nuclear power plants.
- *February 2005:* Christopher Maxwell and two juvenile helpers were paid to spread adware by engineering a botnet attack that flooded the network at Seattle's Northwest Hospital, jamming keycards and doctors' pagers while also shutting down intensive-care unit computers. In 2006, Maxwell was sentenced to three years each of prison and supervised release for damaging the hospital's computers, over 1,000 California school computers, and hundreds of other computers worldwide according to the U.S. Department of Defense.
- Naedele & Dzung:

- *May 2004:* Some 300,000 commuters were stranded for a day when the "Sasser" worm infected the signaling and control system of Australian railway company, RailCorp.
- *August 2003:* The U.S. railway company, CSX Transportation, suffered a worm infection in the communication network used for signaling, halting all trains for half a day.
- *January 2003:* The safety monitoring system of the Davis-Besse nuclear power plant in the U.S. was infected with the "Slammer" worm. The worm bypassed the plant's firewalls via a contractor's laptop which was connected both to the power plant network and to the contractor's infected company network. Three years later, this worm impacted at least two power utilities, a nuclear reactor safety monitor, and an emergency services phone system.
- *December 2000:* Attackers compromised the computer network of an unnamed U.S. power utility by exploiting an unsecured data exchange protocol. The compromised hosts were used to play networked computer games and co-opted so much of network's computing resources and bandwidth that it severely impeded the utility's electricity trading.
- *March 2000:* The control system of a sewage treatment plant in Queensland Australia was accessed by a disgruntled former contractor who flooded the surrounding area with millions of gallons of untreated sewage.
- *January 1998:* External attackers took over the central control center for the pipeline system of Gazprom, the primary natural gas distributer in Russia. For an unknown period of time they were able to control flows throughout the complete Gazprom pipeline network.

Appendix B – Table of 10 Most Critical Industrial Control System Vulnerabilities

In May 2010 the U.S. Department of Energy's National SCADA Test Bed program released the Idaho National Labs 10 most significant ICS vulnerabilities shown below as established using Common Weakness Enumeration (CWE) and Common Vulnerability Scoring System (CVSS v2) metrics applied generically to the vulnerabilities identified during NSTB assessments.

	Vulnerability Source	Risk Level (0.0-10.0)	Ease of Attacker Detection	Attacker Awareness	ICS Prevalence
1	Unpatched Published Vulnerabilities	9.8	Easy	High	High
2	Use of Vulnerable Remote Display Protocols	9.8	Easy	High	High
3	Web HMI Vulnerabilities	9.8	Med-High	High	High
4	Buffer Overflows in ICS Services	9.3	Easy	High	Widespread
5	Improper Authentication	9.3	Moderate	High	High
6	Improper Access Control (Authorization)	9.1	Moderate	High	Widespread
7	Use of Standard IT Protocols with Clear-text Authentication	9.0	Easy	High	High
8	Unprotected Transport of ICS Application Credentials	9.0	Easy	High	Common
9	ICS Data and Command Message Manipulation & Injection	8.8	Med-High	High	Widespread
10	Data Historian Access / SQL Injection	8.6	Easy	High	Common