ADDRESSING THE CYBER-SECURITY OF MARITIME SHIPPING

Elisa Heymann
Universitat Autònoma de Barcelona, Spain
Barton P. Miller
University of Wisconsin-Madison, US
Mohammed J. Alghazzawi
Universitat Autònoma de Barcelona, Spain
David Incertis
FEPORTS, Valencia, Spain

1. ABSTRACT

Attacks on software occur world-wide on a daily basis targeting individuals, corporations and governments alike. The computer systems that control maritime shipping are at risk from serious disruptions, and these disruptions can stem from vulnerabilities in the software and processes used in these computer systems. These vulnerabilities leave such information systems open to cyber-attack. Disruption of those systems could have disastrous consequences at worldwide level.

The assessment of the security of maritime shipping systems has had two significant limitations. First, existing studies have been directed at identifying risks, but have not taken the critical (and expensive) next step of actually identifying the vulnerabilities present in these systems. Second, these studies have focused on overall port operations. While such an overview is important, and has resulted on overall recommendations for changes in policy, they have not provided an evaluation of security issues in the computer systems that control these ports and their terminals.

We need a focused, detailed, in-depth vulnerability assessment of the software that manages freight systems. In this paper, we survey the state of the art in cyber-security for maritime shipping, identify the main problems and current initiatives, and then outline a new research direction for improving the security of our freight systems.

2. INTRODUCTION

The maritime sector is crucial to the world economy, and the computer technology that manages it is critical to its successful operation. Five years ago, 52% of the goods traffic inside of Europe was carried by maritime

shipping, and today that number is 60%. Maritime shipping uses millions of containers and employs millions of people to move billions of tons of freight annually. The European economy is therefore critically dependent upon the maritime movement of cargo and containers. As a consequence, it is dependent upon the software systems that control their operations.

Maritime freight transportation increasingly relies on Information Communication and Technology (ICT) to manage and optimize its operations and services. ICT makes the essential operations not only manageable but also cost effective; this technology is involved in many areas, from traffic control communications to container freight tracking to the actual movement of containers. As a consequence, there is an increased dependency on electronic communication and processes with little human interaction. The flip side of these operational benefits is that **freight ICT systems can be extremely vulnerable to cyber attack**.

Freight ICT systems are large and complex. This software system has many components used by different principals involved in the supply chain. Some of these components are used by the general public, for example the Port Community System (PCS) to book and track shipments and exchange documents and information between public and stakeholders. Other components are intended to be used by port operators, for example the Terminal Operating System (TOS) to control containers movement and storage in the maritime port. There is also a back-office management and integration system, which allows companies to manage, link and share internal processes with suppliers and customers. Attackers can take advantage of the complexity of this diverse collection of software. For example, in 2013 drug traffickers recruited hackers to breach the ICT systems that controlled the movement and location of containers in the Belgian port of Antwerp, managing to reroute containers carrying drugs (BBC 2013).

The software that manages and controls freight transportation systems must be hardened against cyber-attacks. Disruption or unavailability of these ICT systems could have disastrous consequences in cost and availability of goods. Attacks against vulnerabilities in the software can lead to a wide range of consequences. These consequences include service disruption, cargo being shipped to an unintended destination, threat to human lives (for example, remotely controlling the twistlocks of a container spreader to release it over a person), and unauthorized operation of a crane. Therefore, there is a critical need to ensure the robustness of the ICT, and to secure it against cyber attacks. Improving the security and protection of maritime freight

information system from crime and terrorism should be a priority for the European Community and broader world.

3. RELATED WORK

There has been an increasing awareness on port security. Nevertheless the assessment of the security of maritime freight systems (in both the E.U. and U.S.) has had two significant limitations. First, while existing studies have been directed at taking the important first step of identifying risks, they have not taken the critical and expensive next step of actually identifying the vulnerabilities present in these systems. Second, these studies have focused on overall port operations. While such an overview is important, and has resulted on overall recommendations for changes in policy, they have not provided a detailed evaluation of security issues in the ICT systems that control these ports.

In this section we review related work in the areas of risk assessment in container seaports, focusing on its relationship to in-depth software assessment of maritime freight ICT systems.

There have been several efforts that have addressed the risk assessment of seaports. Current efforts for risk assessment for maritime security are depicted in Figure 1.

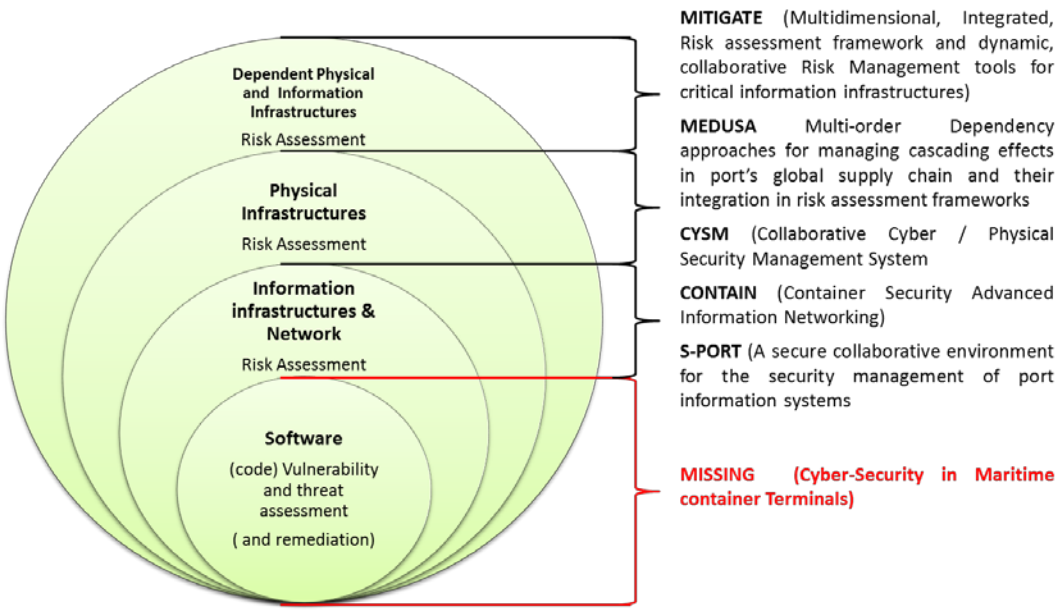

Figure 1 Cyber-Physical security efforts

European Projects like MEDUSA and MITIGATE (Medusa 2014 and Mitigate 2015), both ongoing at present, focus on the interdependencies and

cascading effects in the maritime supply chain and port/maritime systems. MEDUSA concentrates on port IT infrastructure at the supply chain level, while MITIGATE concentrates at the asset level. These approaches are at a high level and do not consider how a vulnerability in the code could affect the higher level spheres (physical assets, networks, information infrastructure) and specifically how these vulnerabilities could cause a cascading effect inside and outside a terminal or port area. This kind of cascading-effect assessment from the low level (code) to the high level (systems and infrastructures) has not been yet tackled in the maritime transport domain.

Assessing risk in these critical freight infrastructures requires a novel approach due to the high complexity, multiple interdependencies, inherent heterogeneity of the port environment (Theoharidou et al. 2011), and the critical and sensitive data managed by those systems. The large volume of information processed, complexity and distributed assets characterizes port ICT systems (Ntouskas et al. 2010).

Existing security standards, best practices, maritime regulation and risk assessment methodologies and tools fail to adequately address the specific needs of port authorities (Ministry of Shipping and Aegean 2013 and International Maritime Organization 2012). In the S-Port project (Polemi et al. 2013), they developed a prototype software platform consisting of a collaborative environment to host security management services and guide commercial ports to monitor and self-manage their port ICT security. Safety standards and regulations (specifically ISO 27001 and ISPS Code) were first identified and then actions were taken to address some specific security management needs of port ICT systems. The architecture of the S-Port platform incorporates various collaborative tools, which are focused on high-level risk assessment (Ntouskas et al. 2010).

Historically physical security has been the main emphasis when thinking about port security. Traditionally the various seaports standardization bodies did not refer in their memoranda to ICT/Cyber-security (Polemi et al. 2012-B). Most of the existing freight seaport security standards and methodologies concentrated only on the physical security of the ports (safety) (Polemi et al. 2012).

The International Maritime Organisation (IMO) is developing guidelines for maritime cyber-risks, as the basis for future regulation in the maritime and port sector. During the last IMO's Maritime Safety Committee (MSC) session held in May 2016, the Committee approved the new MSC.1/Circ.1526 (IMO 2016)

providing draft interim guidelines on maritime cyber risk management. The guidelines provide high-level recommendations to safeguard shipping from current and emerging cyber-threats and vulnerabilities.

Since port ICT systems face combined physical and cyber threats, a holistic risk assessment methodology for these infrastructures should combine the analysis of the physical and ICT aspects. For example using MSRAM (Downs et al. 2008), and CMA (Kang et al. 2009) for physical risk assessment; and CRAMM (Insight Consulting 2012), OCTAVE (Alberts et al. 2001)), or current standards (ISO27005 (ISO/IEC 2011), NIST-SP 800-30 (Stoneburner et al. 2011) for ICT risk assessment.

While awareness of cyber-risks is steadily increasing in the maritime sector, we need to go beyond risk assessment to the actual evaluation of the security of the software systems that operate in this environment.

4. IN-DEPTH VULNERABILITY ASSESSMENT

Current approaches are not sufficient to address the challenges ahead to prevent cyber-attackers from accessing and controlling the software systems that would grant the attackers control to the freight transportation chain.

As a result of this situation, we need a **focused, low level, in-depth vulnerability assessment** of the software that manages freight systems. This would include a deep analysis of the software including a low level code review that goes beyond the use of automated assessment tools. The ultimate goal is to find critical vulnerabilities so that the software providers could remediate them before the attackers are able to exploit them.

Until recently, there was no structured methodology for in-depth assessment of software systems at the code level. Simply trying to examine all the code in a complex system such as these would be an overwhelming task, beyond any reasonable cost or staffing. Based on our experience with analysing code for security flaws, we developed the First Principle Vulnerability Assessment (FPVA) methodology. FPVA (Kupsch et al. 2009) was developed primarily as an analyst-centric approach to assessment, the aim of which is to focus the analyst's attention on the parts of the software system and its resources that are mostly likely to contain vulnerabilities and provide access to high-value assets. FPVA (Kupsch et al. 2010) has been used to evaluate several well-known systems, such as Google Chrome, HTCondor (HTCondor 2016), and Wireshark (Wireshark 2016).

Rather working from known vulnerabilities, the starting point for FPVA is to identify **high value assets** in a system, i.e., those components (for example, processes or parts of processes that run with high privilege) and resources (for example, configuration files, databases, connections, devices) whose exploitation offer the greatest potential for damage by an intruder. From these components and resources, we work outward to discover execution paths through the code that might exploit them. This approach has a couple of advantages. First, it allows us to find new vulnerabilities, not just exploits based on those that were previously discovered. Second, when a vulnerability is discovered, it is likely to be a serious one whose remediation is of high priority.

FPVA starts with an architectural analysis of the code, identifying the key components in a distributed system. It then goes on to identify the resources associated with each component, the privilege level of each component, the value of each resource, how the components interact, and how trust is delegated. The results of these steps are documented in clear diagrams that provide a roadmap for the last stage of the analysis, the manual code inspection. In addition, the results of this step can also form the basis for a risk assessment of the system, identifying which parts of the system are most immediately in need of evaluation. After these steps, we then use code inspection techniques on the critical parts of the code. Our analysis strategy targets the high value assets in a system and focuses attention on the parts of the system that are vulnerable to not only unauthorized entry, but unauthorized entry that can be exploited.

After we know where to focus the search, that means after we understand which the high value assets are, we can apply a variety of tools and techniques to the actual analysis of the code. It is worth noticing that they complement the manual inspection of the code, but never replace it. We now describe some of those tools and techniques.

F*uzz testing* (Clarke et al. 2009 and Miller et al. 1991) is a software testing technique, often automated or semi-automated, that involves providing invalid, unexpected, or random data to the inputs of a computer program, then monitoring its responses for exceptions such as crashes, for finding potential memory leaks, unhandled exception, or anything could lead to a vulnerability. The fuzzing data can be providing to a system as parameters, input file, network packets, or any other way a system accepts user input. Fuzz testing enhances software security and software safety because it often finds weaknesses and defects that human testers would fail to find, and that even

careful human test designers would fail to create tests for this purpose. However the main problem with fuzzing is that it generally only finds very simple faults.

Runtime checking is a technique for preventing the exploitation of vulnerabilities, but when combines it with fuzz testing it can be used for vulnerability detection. This method based on inserting additional checks into a system to ensure that its behavior fits with a certain set of restrictions. For instance, Mudflap (Frank 2003) and ProPolice (Etoh et al. 2000) are pointer use checking extensions to the GCC compiler. They add instrumentation code to potentially unsafe pointer operations in programs and detect errors such as NULL pointer dereferencing, buffer overflows and memory leaks. Runtime checking can help uncover vulnerabilities during fuzz testing, even if they do not result in a system crash. The disadvantage of both fuzz testing and runtime checking is that some vulnerabilities can be discovered only under specific cases that might not appear during regular use of the system. For example, exploiting a system might require sending a long sequence of specifically crafted network packets to reach a vulnerable state.

Static source analysis tools or *automated vulnerability assessment* tools can help to increase the efficiency of source code auditing. Such tools are widely available now, with examples including commercial tools such as Coverity Analysis (Coverity 2016) and Parasoft JTest (Parasoft 2016). Those tools examine the source code of a program and report possible weaknesses. They are useful for guiding developers to potentially vulnerable code. A single tool will not be able to find any possible weaknesses in the code. Different tools find different kind of weaknesses, and tools will never find new kinds of vulnerabilities, i.e. vulnerabilities not reported before. A complete survey on automated tools and its utilization can be found in (Kupsch et al. 2016). The Swoftware Assurance Market Place (SWAMP) is a facility available to developers that provides access to different automated tools, both open source and commercial (SWAMP 2016).

In the next section we describe how to apply FPVA to the software systems involved in freight maritime transportation with the goal of making it less vulnerable o cyber-attackers.

5. RESEARCH AGENDA

In this research, we will apply for the first time in the seaports domain the First Principle Vulnerability Assessment (FPVA) methodology, developed by the University of Wisconsin-Madison and the Universitat Autònoma de Barcelona

(Kupsch et al. 2010). FPVA is aimed at finding critical vulnerabilities first. Critical vulnerabilities are the ones affecting high value assets. We will apply FPVA to both Terminal Operating Systems (TOS) and Port Community Systems (PCS).

FPVA is composed of the following steps:

1) *Architectural analysis:* We will identify the different software components (processes, threads) running on the different hosts; and will identify the communication amongst those components, and the points where the different users interact with the system (attack surface). Both TOS and PSC are complex, with many components allowing the interaction among the different entities, such as the port authority, the container terminal, the consignee, and the forwarder among others.

2) *Resource analysis:* We will identify the different resources (logical and physical) accessed by the components in step 1. For example relevant resources include the bill of lading, bay plan, the list of containers with dangerous goods, and the database containing information on the containers on the yard. And attacker gaining access to critical resources will result in a big damage.

3) *Privilege and trust delegation:* In this step we identify how resources are protected, the privilege level at which the different components run, and how trust is delegated. Authentication and authorization of access to resources are included in this step. We will analyze the trust relationship between the key entities such as vessels, port operators, and regulatory agencies.

4) *Component analysis:* In this step, we will dig into the TOS and PCS critical components from step 1, accessing the critical resources from step 2.

   We first look for classical vulnerabilities such as:

   - Lack of data validation.
   - Error handling.
   - Buffer overflows.
   - Numeric errors.

- Race conditions (on data and TOCTOU).
- Injection attacks: Format string attacks, command injection, SQL injection, and XML injection.
- Web attacks: Cross-site scripting (XSS), cross-site request forgery, session hijacking, and open redirect.
- Directory traversals.
- Serialization.
- Containment attacks: Insecure permissions, not dropping privileges, information leaks, and lack of authorization

We also look for new vulnerabilities resulting from the interaction of the components in step 1. Therefore we should be able to find both well-known vulnerabilities and new ones.

We will combine the previous manual assessment with using automated assessment tools. Ideally people would like to simply push a button and get a report on the vulnerabilities affecting their software; but current automated tools (even the best of their breed) fail to find relevant vulnerabilities (Kupsch 2009 et al.). Nevertheless these tools provide a good starting point for detecting certain kind of bugs.

6. CONCLUSIONS

In this position paper, we surveyed the state of the art in cyber-security for freight ports, identified the main problems and current initiatives, and then proposed a new research direction and approach for improving the security of our freight systems.

The line (in red in Figure 1) between physical and information infrastructures security and code level software systems security has not yet been crossed in the maritime transport sector, which constitutes a gap and vulnerability itself, and represents an increasing concern for port and terminal operators. Our research will tackle this issue for the first time in this domain.

The starting point for narrowing the gap of cyber security of maritime transportation systems is to apply FPVA with the goal of (1) discover critical vulnerabilities in software for freight terminals and suggest mitigations, and (2) generate the first set of recommendations and guidelines for the maritime

sector on low level cyber-security (code level) of their freight management systems.

7. ACKNOWLEDGMENT

This work was supported by MINECO Spain contract number TIN2014-53172-P.

BIBLIOGRAPHY

Alberts, C.J., and Dorofee, A.J. (2001) OCTAVE Method Implementation Guide Version 2.0, Pittsburgh, PA, United States: Software Engineering Institute, June.

BBC. (2013) http://www.bbc.com/news/world-europe-24539417

Clarke, T. (2009) Fuzzing for Software Vulnerability Discovery, Royal Holloway University of London, Egham, England, United Kingdom, 17, February.

Coverity (2016) http://www.coverity.com/products/coverity-save/

CYSM (2013) http://www.cysm.eu/

Downs, B. (2008) The maritime security risk analysis model, in Critical Infrastructure Protection Workshop, Washington, D.C., US, June.

Etoh, H., Yoda, K. (2000) Protecting from Stack Smashing Attacks, IBM Research Divison, Tokyo, June 19.

Frank, E. (2003) Mudflap: Pointer use checking for C/C++, in Proceedings of the GCC Developers Summit, Otawa, Ontario, Canada, 25 - 27, May.

HTCondor (2016) http://research.cs.wisc.edu/htcondor/

IMO (2016) MSC.1/Circ.1526, Interim guidelines on maritime cyber risk management. International Maritime Organisation.

Insight Consulting (2012) CRAMM User Guide, Global network of innovation, United Kingdom.

International Maritime Organization, (2012) International Ship and Port Facility Security Code and SOLAS Amendments, in Conference of Contracting Governments to the International Convention for the Safety of Life at Sea, Croydon, UK, 17, December.

ISO/IEC (2011) Information Technology - Security Techniques - Information Security Risk Management, ISO, British Standards, London, United Kingdom.

Kang, M., Li, M., Montrose, B., Khashnobish, A., Elliott, S., Bell, M., Pieper, S. (2009) Overview of the security architecture of the Comprehensive Maritime Awareness system, in IEEE Military Communications Conference, Boston, Massachusetts, USA, 18 -21, October.

Kupsch, J., Heymann, E., Miller, B., Basupalli, V. (2016) Bad and good news about using software assurance tools, Software: Practice and Experience (to appear). Available at http://onlinelibrary.wiley.com/doi/10.1002/spe.2401/epdf

Kupsch, J., Miller, B.P. (2009) Manual vs. Automated Vulnerability Assessment, in The First Inernational Workshop on Managing Insider Security Threats, West Lafayette, IN, United States, 15 - 19, June.

Kupsch, J., Miller, B.P., Heymann, E., Cesar, E. (2010) First Principles Vulnerability Assessment, in Proceedings of the 2010 ACM workshop on Cloud computing security workshop, Chicago, Illinois, United States, 8, October.

Medusa (2014) http://athina.cs.unipi.gr/medusa/

Miller, B.P., Fredriksen, L., So, B. (1991) An Empirical Study of the Reliability of UNIX Utilities, in Communications of the ACM33, December.

Ministry of Shipping and Aegean (2013), European Maritime in Aegean Sea Islands, in European Maritime Day, Valletta, Malta, 20 - 21, May..

Mitigate (2015) http://www.mitigateproject.eu/

Ntouskas, T., Polemi, N. (2010) A secure, collaborative environment for the security management of port information systems, IEEE computer society, vol. 62, no. 10, pp. 374 - 379.

Papastergiou, S., Polemi, D., Karantjias, A., (2015) CYSM: An innovative physical/cyber security management system for ports, in Special Session on Innovative Risk Management Methodologies and Tools for Critical Information Infrastructures (CII) within the 6th International Conference on Digital Human Modelling and Applications in Health, Safety, Ergonomics and Risk Management (HCI International), pp. 2-7.

Parasoft (2016) Parasoft Jtest https://www.parasoft.com/product/jtest/

Polemi, D., Ntouskas, T. (2012) Collaborative Security Management services for Port Information Systems, in Proceedings of International Conference on e-Business, Scitepress, Italy, 10, March.

Polemi, D., Ntouskas, T. (2012-B) Open issues and proposals in the IT security management of commercial ports: The S-Port national case, in The 27th IFIP International Information Security and Privacy Conference, Heraklion, Crete, Greece, 4-6, June.

Polemi, D., Ntouskas, T., Georgakakis, E., Douligeris, C. (2013) S-Port: Collaborative Security Management of Port Information Systems, in Information, Intelligence, Systems and Applications (IISA), 2013 Fourth International Conference on, Piraeus, 10-12 July.

Serrano, J., Cesar, E., Heymann, E., Miller, B.P. (2013) Increasing Automated Vulnerability Assessment Accuracy on Cloud and Grid Middleware, Springer-Verlag, vol. 7863, pp. 278 - 294,.

Stoneburner, G., Goguen, A., Feringa, A. (2011) Information Security, National Institute of Standards and Technology, Gaithersburg, MD, United States, March.

SWAMP (2016) Software Assurance Market Place. https://continuousassurance.org/

Theoharidou, M., Kandias, M., Gritzalis, D. (2011) Securing transportation critical infrastructures: Trends and perspectives, in The 7th IEEE International Conference in Global Security Safety and Sustainability, Thessaloniki, Greece, 24 - 26, August.

Wireshark (2016) https://www.wireshark.org/