In-Depth Software Vulnerability Assessment of Container Terminal Systems

Joseph O. Eichenhofer Elisa Heymann Barton P. Miller

Computer Sciences Department
University of Wisconsin – Madison, USA
{eichenhofer, elisa, bart}@cs.wisc.edu

Abstract

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

The assessment of the security of maritime shipping systems has faced 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 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.

In response, we performed a detailed analysis of the information flow involved in the maritime shipping process. Though most of the communication is electronic, there are still some paper documents at some steps of the process.

After understanding the relevance of the cyber components involved, we executed a detailed, in-depth vulnerability assessment of the software that manages freight systems. In this paper, we show in detail the flow of information involved in the freight shipping process and explain how we performed the in-depth assessment, summarizing our findings. Like every large software system, maritime shipping systems have vulnerabilities. The question is who will find them first: Software analysts or real attackers.

1. Introduction

The maritime sector is crucial to the world economy, and the computer technology that manages it is critical to its successful operation. Maritime ports in the EU handled 3.8 billion metric tons of seaborne goods in 2015, which marked a slight increase of 1.4% when compared with 2014, but an increase of 10.8% when compared with 2009 [1]. In the US, maritime ports handle collectively 75% of America's international trade by volume [2]. Maritime shipping uses millions of containers and employs millions of people to move billions of tons of freight annually. The world economy is therefore critically dependent upon the maritime movement of cargo and containers. As a consequence, the economy is also dependent upon the software systems that control the maritime operations.

Maritime freight transportation increasingly relies on Information and Communications 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. In addition to these benefits, the freight ICT systems also introduce the risks of being extremely vulnerable to cyber-attack.

Freight ICT systems are large and complex, having 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 container 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, guns, and cash [3].

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 disruption of service, shipment of cargo to unintended destinations, threat to human lives (for example, by remotely controlling the twistlocks of a container spreader to release it over a person), and operation of seaport machinery by unauthorized users. Therefore, there is a critical need to ensure the robustness of the ICT and to secure it against cyber-attacks.

2. Related Work

There has been an increasing awareness of port security in the past decade. Nevertheless, assessment of the security of maritime freight systems (in both the E.U. and U.S.) has faced 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 the ICT systems. Second, these studies have focused on overall port operations. While such overviews are important and have resulted in overall recommendations for policy change, 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 summarized in Figure 1.

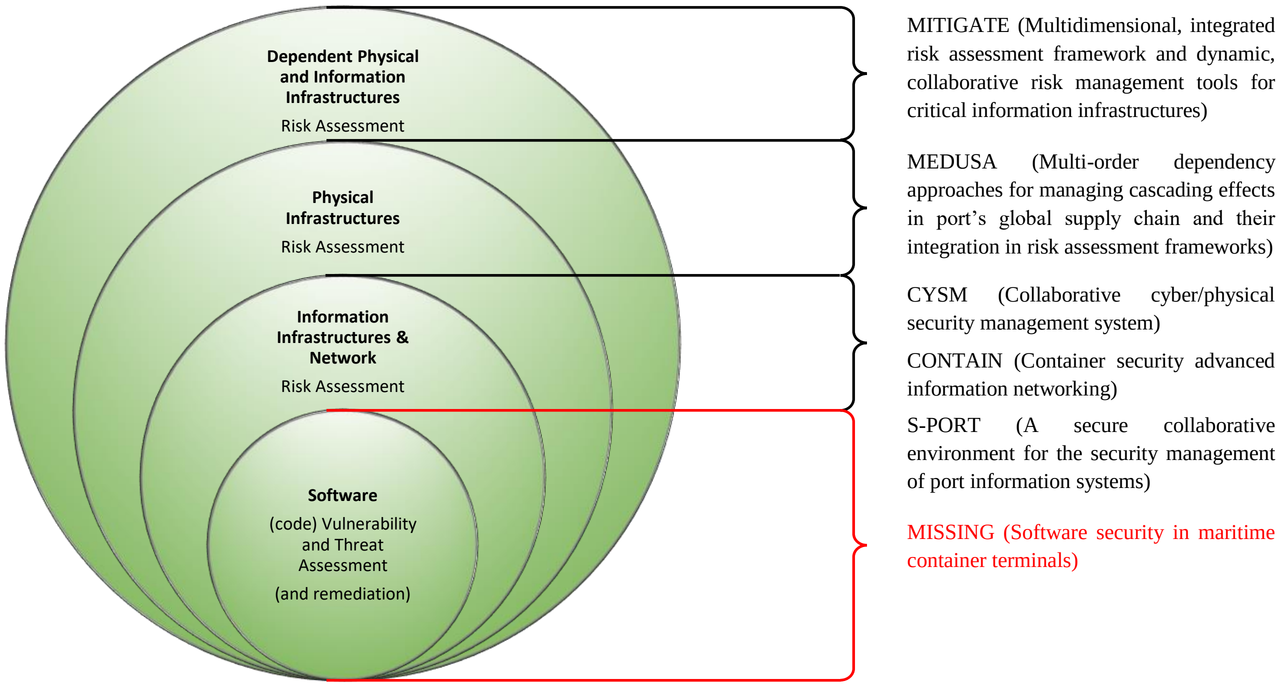

Figure 1. Cyber-Physical security efforts

Ongoing European Projects like MEDUSA and MITIGATE [4, 5], 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). Specifically, these approaches do not address 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 yet been tackled in the maritime transport domain.

Existing security standards, best practices, maritime regulation, and risk assessment methodologies and tools fail to adequately address the specific needs of port authorities [6, 7]. Researchers in the S-Port project 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 [8]. Safety standards and regulations were identified (specifically in ISO 27001 and ISPS Code), 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 [9].

Historically, physical security has been the main emphasis when thinking about port security; the various seaports standardization bodies did not specifically reference ICT/Cyber-security in their memoranda [10]. Most of the existing freight seaport security standards and methodologies concentrated only on the physical security of the ports (i.e., safety concerns) [11].

The International Maritime Organization (IMO) is developing guidelines for maritime cyber-risks as the basis for future regulation in the maritime and seaport sector. During the last IMO's Maritime Safety Committee (MSC) session held in June 2017, the Committee approved the new MSC.428(98) Maritime Cyber Risk Management in Safety Management Systems [12]. Following MSC.1/Circ.1526, the resolution affirms that approved safety management systems should take cyber risk management into account [13]. The guidelines in 2016 provided high-level recommendations to safeguard shipping from current and emerging cyber-threats and vulnerabilities. The 2017 guidelines are not available at the time of this writing.

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 [14] and CMA [15] for physical risk assessment, and using CRAMM [16], OCTAVE [17], or current standards such as ISO27005 [18] and NIST-SP 800-30 [19] 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 software systems that operate in this environment. The first step to an in-depth assessment of the software that controls maritime freight shipping consists of understanding the software involved. There cannot be a serious cyber security analysis without taking into account the software. For that purpose, we investigated the maritime shipping process and documented all the documents (both electronic and in paper) involved. This documentation is detailed in the next section.

3. Understanding Shipping Logistics

The process by which a shipping container carries goods from an exporter in one country to an importer in another can be viewed as a series of document and communication transactions.

Figure 2 shows the communications/transactions involved in shipping logistics. Due to the large and complex nature of freight logistics, it is beneficial to approach the process in stages. For the purposes of this paper, there are six such stages: booking, forwarding, outbound customs, outbound shipping, inbound shipping, and delivery. To better visualize these stages, the transactions involved in each stage are shown in Figure 3 through Figure 8. Each arrow represents a transaction of paper document (green), digital document (red), container movement (blue), or unspecified communication (black). Transactions are chronologically numbered. Simultaneous transactions in the same figure share the same number and are identified by letter.

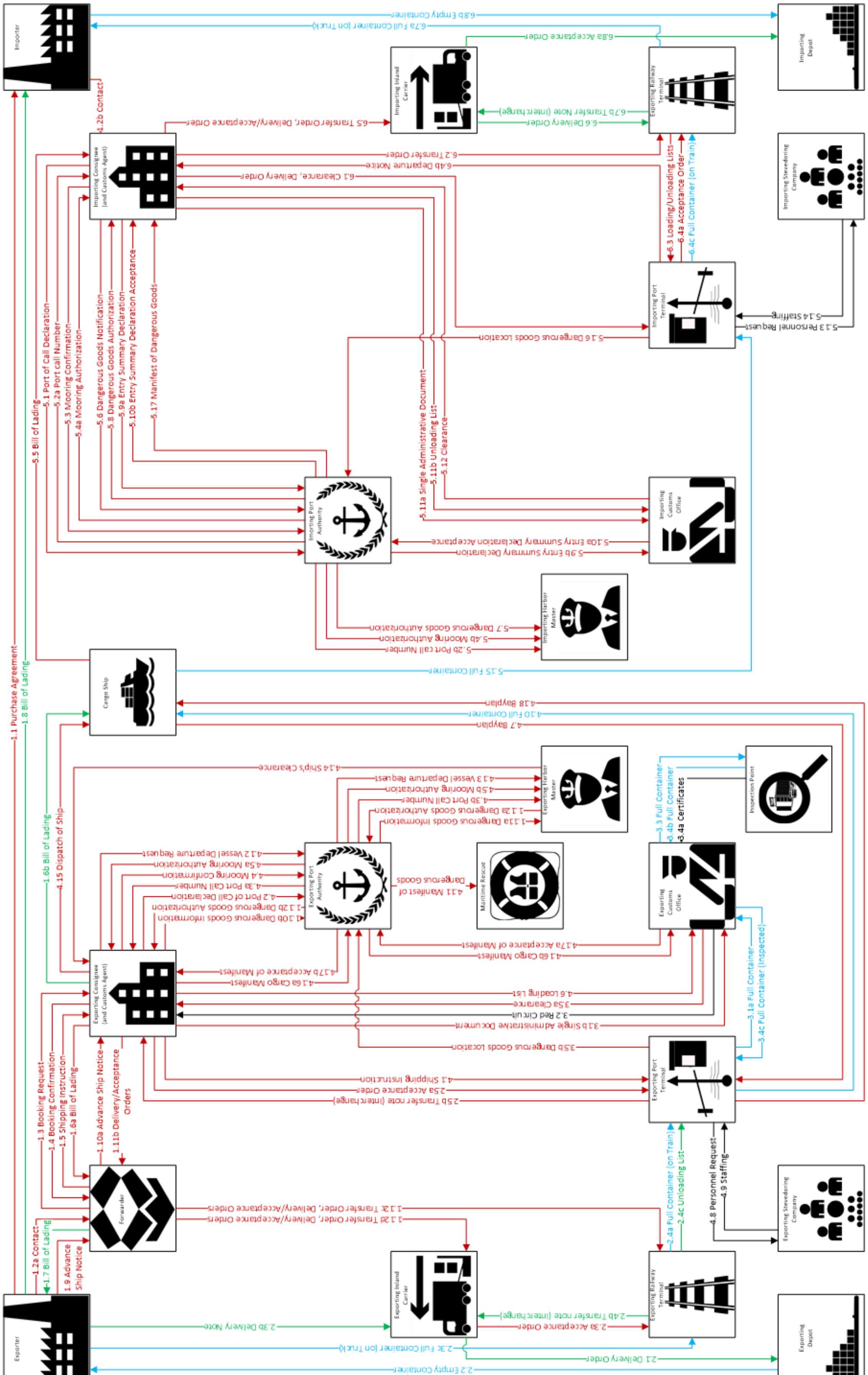

Figure 2. Shipping logistic data flows

3.1. Booking

Several booking-related documents must be created and exchanged before the container can be moved. In this section, parenthesized numbers refer to edges in Figure 3. The importer and exporter first agree on the goods to be purchased and shipped (1.1). For the sake of simplicity, we do not show the importer in this figure. The exporter contacts the freight forwarder (1.2a) who will negotiate shipment with the consignee that operates in the desired seaport (1.3, 1.4, and 1.5). A Bill of Lading is created by the consignee and given to the cargo ship, forwarder, exporter, and importer (1.6a, 1.6b, 1.7, and 1.8). When the exporter is ready to ship, it sends an advance ship notice to the forwarder who sends it to the consignee (1.9 and 1.10a). The consignee sends delivery and acceptance orders to the forwarder (1.11b) who sends them to the inland carrier and railway (1.12a and 1.12b). If the shipment is to contain any dangerous goods, the consignee reports them to the port authority (1.10b). When the port authority and harbor master approve the goods, authorization is recorded and given to the consignee (1.11a, 1.12a, and1.12b).

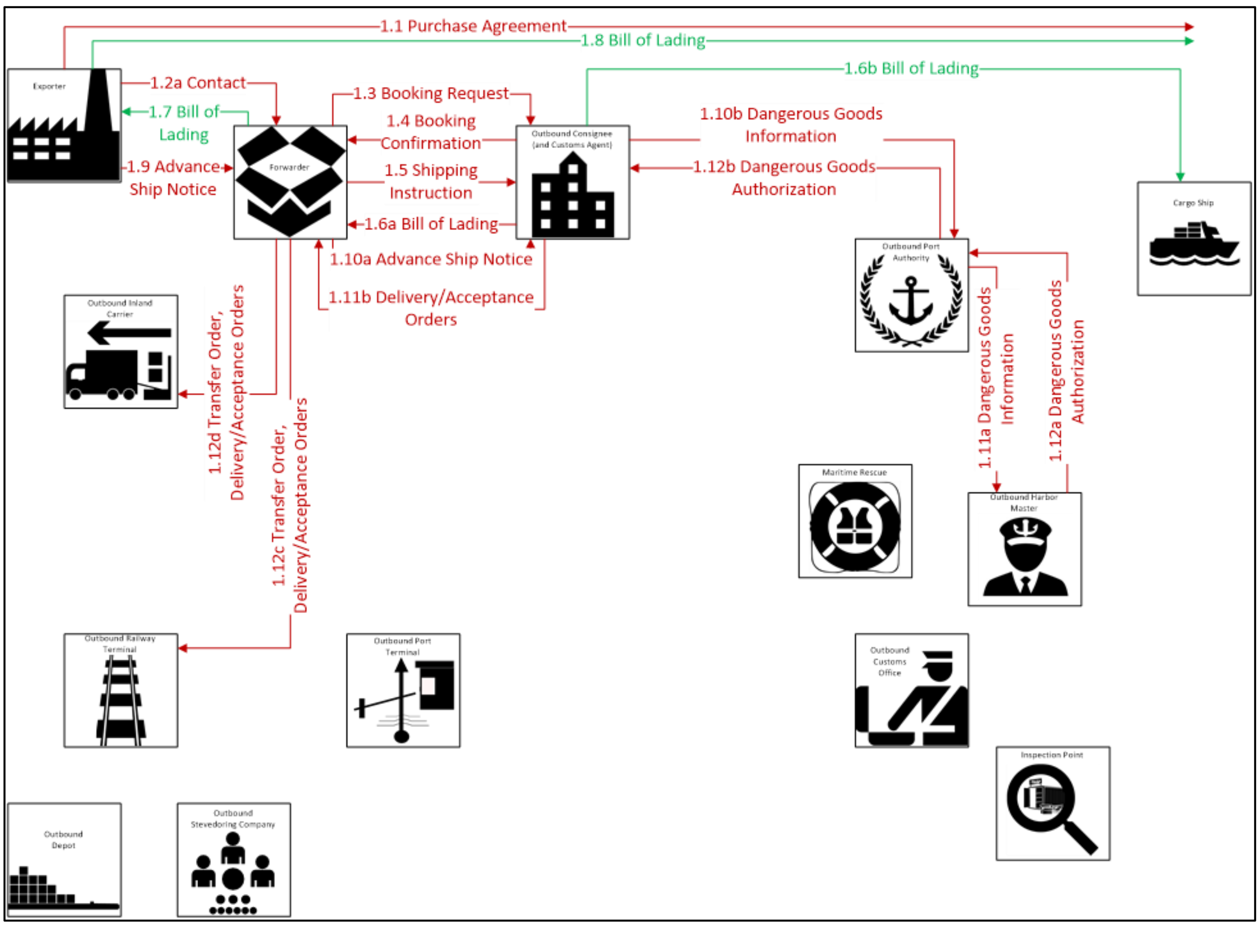

Figure 3. Booking logistic data flow

3.2. Forwarding

Once booking documents are in place, the goods will be forwarded to the seaport. Parenthesized numbers in this section refer to edges in Figure 4. The inland carrier first takes the delivery order to the depot at the seaport to receive the consignee’s container and takes the empty container to the exporter (2.1 - 2.2). The container is packed and sealed in the presence of a representative of the exporter who signs a delivery note and gives it to the carrier (2.3b). The carrier takes the full container and an acceptance order to the railway terminal (2.3a and 2.3c). The carrier is given a transfer note to document the exchange (2.4b). The railway operator loads and sends the container to the port terminal along with an unloading list that documents the goods (2.4a and 2.4c). The consignee sends to the terminal an acceptance order, and the terminal sends the consignee a transfer note to document the interchange (2.5a and 2.5b).

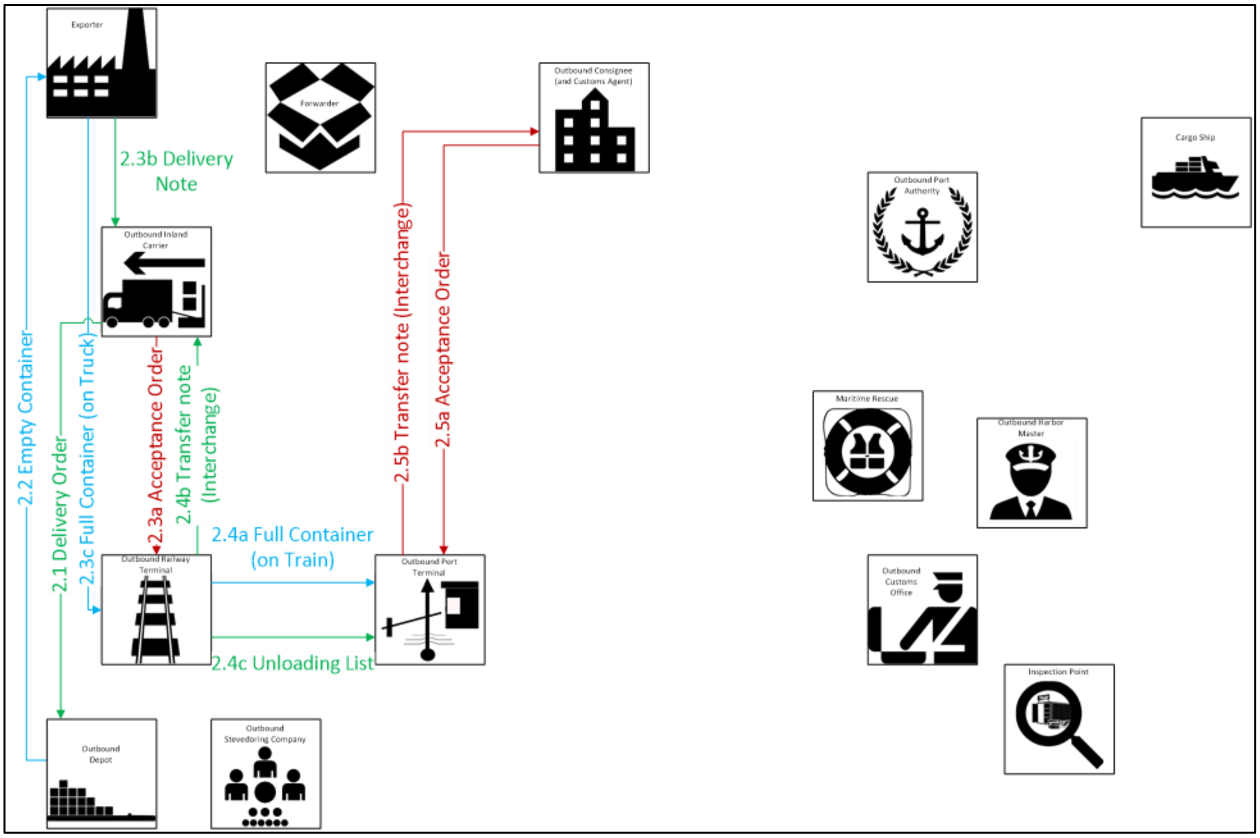

Figure 4. Outbound forwarding logistic data flow

3.3. Outbound Customs

Many containers are subject to customs clearance and/or inspection once they arrive at the seaport. Edges in Figure 5 are referenced by parenthetical numbers in this section. The container is taken to a checkpoint run by the customs office (3.1a). Customs declarations are sent by the consignee in the form of a “Single Administrative Document” to the customs office at the port (3.1b). If the container is to be inspected, a “red circuit” is initiated (3.2). The container is moved to the inspection site (3.3), certified, and returned to the customs office and port terminal (3.4a, 3.4b, and 3.4c). Clearance documentation is sent to the consignee (3.5a). If the container contains any dangerous goods, they are reported to and tracked by the port authority (3.5b).

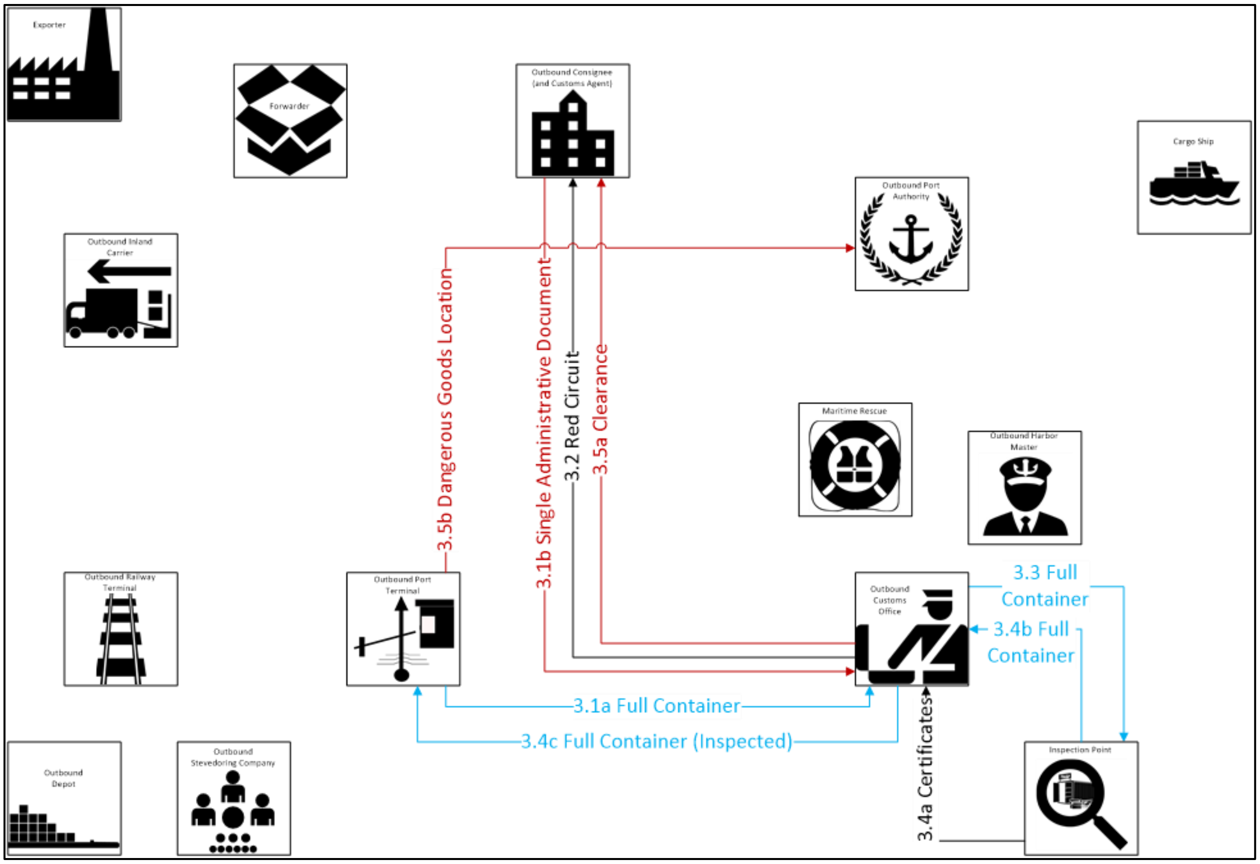

Figure 5. Outbound customs logistic data flow

3.4. Outbound Shipping

With the container certified and available at the terminal, arrangements must be made for its loading and shipment out of the port. In this section, parenthetical numbers are references to single edges in Figure 6. After sending shipment instructions to the terminal (4.1), the consignee sends and receives authorizing documents to the port authority for the cargo ship to dock (4.2, 4.3a, 4.4, and 4.5a), some of which are sent to the harbor master for record and reference (4.3b and 4.5b). The consignee must also report to the customs office a loading list for record of the goods (4.6). The docked ship then sends its bayplan to the terminal (4.7), where arrangements are made for the ship to be unloaded and loaded by stevedores (4.8, 4.9, and 4.10). If any dangerous goods are loaded, they are reported to maritime rescue authorities for tracking (4.11). Once loading is complete, the consignee makes a request to the port authority to embark (4.12), and notifies the ship after it is authorized by the port authority and harbor master (4.13, 4.14, and 4.15). The consignee sends a cargo manifest to the port authority (4.16a), which reviews it with the customs office before documenting its acceptance (4.16b, 4.17a, and 4.17b). An updated bayplan is sent back to the ship as it departs (4.18).

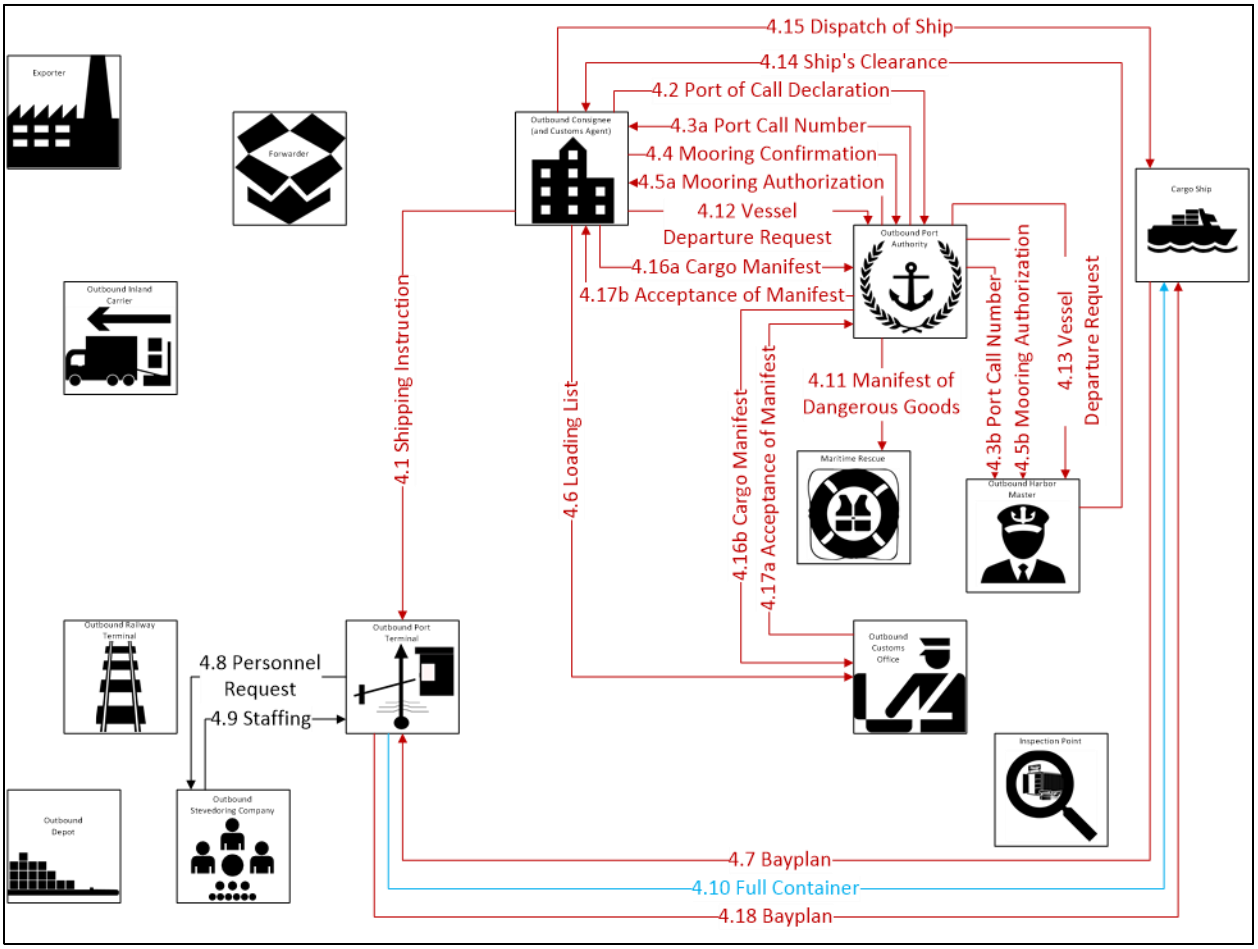

Figure 6. Outbound shipping logistic data flow

3.5. Inbound Shipping

The process of shipment into the receiving port begins as the cargo ship nears it. Parenthetical numbers in this section refer to edges in Figure 7. When the cargo ship approaches the receiving port, the consignee arranges for authorization from the port authority to dock (5.1, 5.2a, 5.3, and 5.4a). The port call number and mooring authorization are sent to the harbor master for record (5.2b and 5.4b). Dangerous goods must be reported to and authorized by the port authority and recorded by the harbor master (5.6, 5.7, and 5.8). The consignee sends an entry summary declaration to the port authority which forwards it to the customs office (5.9a and 5.9b). The customs office accepts the declaration (5.10a), and the consignee is notified (5.10b). A Single Administrative Document is sent to the customs office along with an unloading list (5.11a and 5.11b). Once customs clearance is granted (5.12), the port terminal arranges for stevedores to unload and load the ship (5.13, 5.14, and 5.15). Locations of dangerous goods are reported to the port authority (5.16), and a manifest of them are sent to the consignee (5.17).

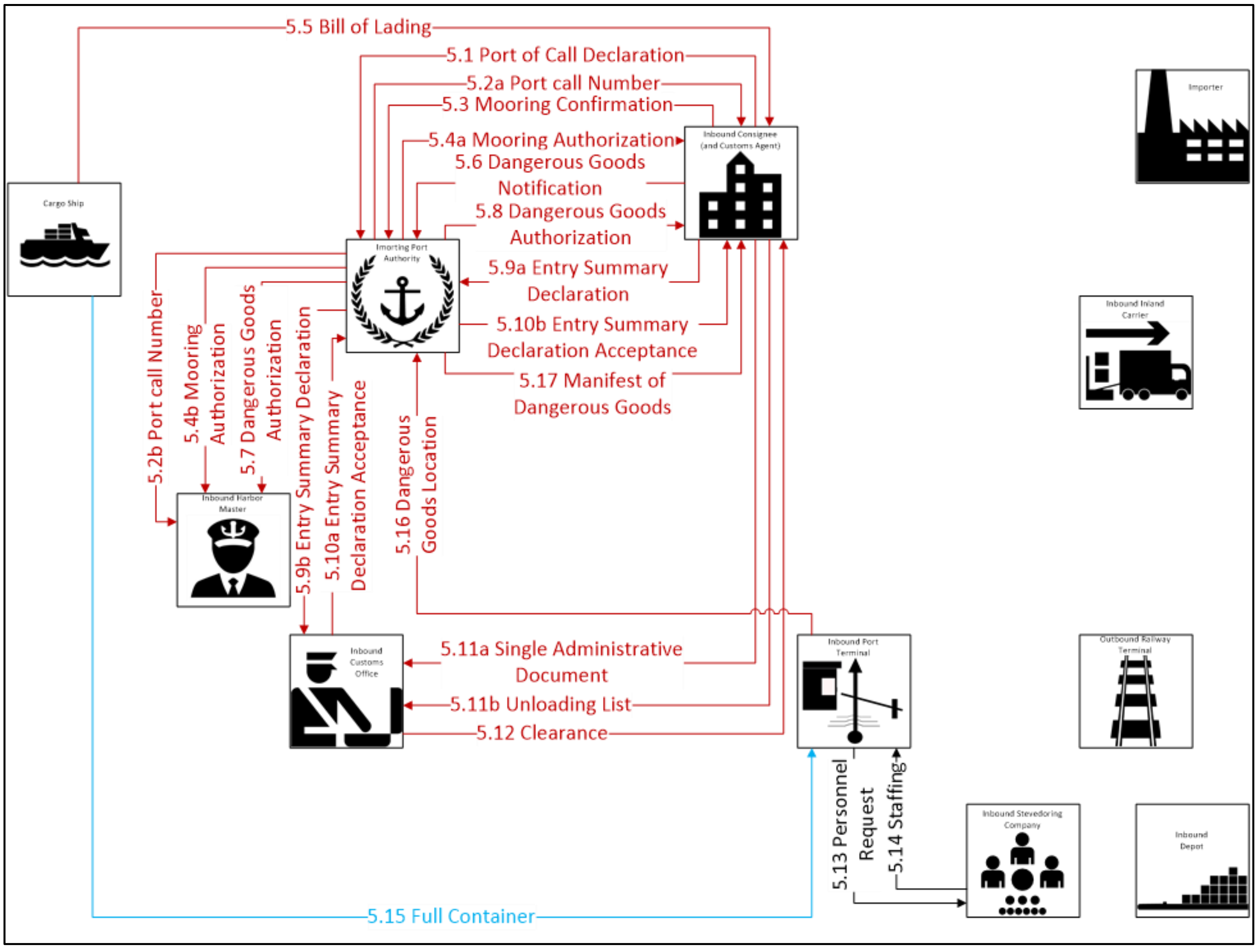

Figure 7. Inbound shipping logistic data flow

3.6. Delivery

The final stage of the process is to move the full container from the port, deliver the goods to the importer, and return the empty container to the depot. Parenthetical numbers in this stage are references to edges in Figure 8. The consignee sends its customs clearance and delivery order to the terminal (6.1) and a transfer order to the railway terminal that will take the container (6.2). The railway terminal sends a loading/unloading list to the port terminal (6.3), where internal transportation unloads and loads the appropriate containers. The container and an acceptance document are sent to the railway terminal (6.4a and 6.4c), and a departure notice is sent back to the consignee (6.4b). The consignee sends the required carriage documents to the inland carrier (6.5) which brings the consignee's delivery order to the railway terminal in order to take the container (6.6). The railway terminal gives the carrier a transfer note documenting the interchange (6.7b). The carrier delivers the container to the importer, where it is unloaded (6.7a). The empty container is then brought with the consignee's acceptance order to the depot where it is stored until a new shipment is ready (6.8a and 6.8b).

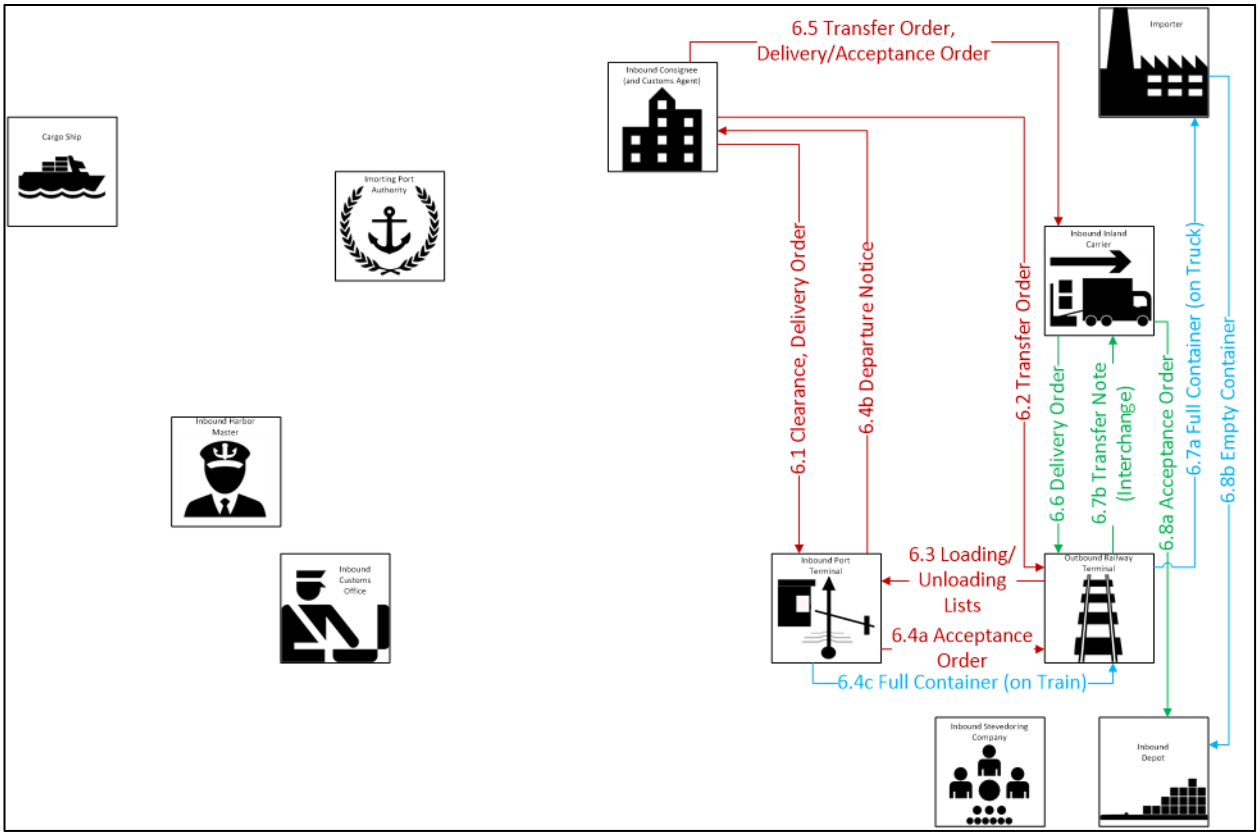

Figure 8. Delivery logistic data flow

4. In-Depth Vulnerability Assessment

In the previous section, we showed a transactional view of shipping logistics. In this section, we describe the methodology for performing an in-depth vulnerability assessment of the software that facilitates the transactions previously described. This assessment includes 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, a task beyond any reasonable cost or staffing. Based on our previous experience with analyzing code for security flaws, we developed the First Principle Vulnerability Assessment (FPVA) methodology [20]. FPVA 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 related to high-value assets. FPVA has been used to evaluate many well-known systems, including Google Chrome [21], HTCondor [22], and Wireshark [21].

Rather than working from known vulnerabilities, the starting point for FPVA is to identify high value assets in a system: 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 two immediate 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, the interaction between components, and the delegation of trust. The results of these steps are documented in clear diagrams that provide a roadmap for the last stage of the analysis, which is the manual code inspection. Additionally, 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 just unauthorized entry but specifically unauthorized entry *that can be exploited*.

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

In the FPVA of freight ICT systems, we followed the following steps:

1. **Architectural analysis:** Identify the different software components (processes and threads) running on the different hosts, the communication amongst those components, and the points where the different users interact with the system. Both TOS and PSC are complex, with many components facilitating the interaction among the seaport stakeholders including the port authority, the container terminal, the consignee, and the forwarder.
2. **Resource analysis:** Identify the different resources (logical and physical) accessed by the components in step 1. For example, relevant resources include the bill of lading, bayplan, the list of containers with dangerous goods, and the database containing information on the containers on the yard. An attacker gaining access to these critical resources would result in severe damage.
3. **Privilege analysis and trust delegation:** Identify the resource protections, the privilege levels at which each component runs, and the delegation of trust. Authentication and authorization of access to resources are also identified in this step. We analyze the trust relationships between key entities such as terminal stations, port operators, forwarders, and shipping companies.
4. **Component analysis:** Perform a fine-grain evaluation of the critical components and resources identified in step one and two. This step is the most time consuming and

involves the identification of vulnerabilities as well as the construction of proof-of-concept exploits. The process of this step is described below.

When performing the component analysis, we first look for classical vulnerabilities such as:

- Improper or insufficient data validation
- Improper error handling
- Buffer overflows
- Numeric errors
- 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 traversal
- Unsafe serialization
- Containment attacks: insecure permissions, not dropping privileges, information leaks, and improper authorization

We also look for new vulnerabilities resulting from the interaction of the components in step one. Through this approach, we identified both common vulnerabilities and vulnerabilities specific to the system we analyzed.

In this research, we applied the FPVA methodology for the first time in the maritime domain with the goal of making its software less vulnerable to cyber-attackers. We applied FPVA to some modules of the TOS and PCS provided by a well-known software provider in maritime freight shipping. The next section summarizes our findings.

5. FPVA Vulnerability Assessment Results

In this section, we summarize the results of performing an in-depth vulnerability assessment on some modules of a TOS and PCS from a well-known software provider in the domain of maritime freight shipping. A thorough report and discussion of vulnerability results is not within the scope of this paper. It is worth noting that our results were reported to the software developers in full, including close collaboration to remedy the discovered vulnerabilities.

In our code assessment, we found several high-impact vulnerabilities. Some of the vulnerabilities we found and reported include the following weaknesses:

1. HTTP traffic was not encrypted. As a consequence, the system was vulnerable to:
    - Session hijacking: HTTP sessions are tracked using session ID cookies. The server determines client identity and state by associating data with a particular session. If traffic is unencrypted, the value of this session ID can be recorded by an attacker. The attacker can then send requests using that session ID to effectively impersonate the victim, gaining access to all resources available to the victim whose session was hijacked.

- Password sniffing: A user’s username and password is transmitted in plain text when logging into the system. Any devices connected to the same physical (or virtual) network as a client or server will be able to read the username and password of any user that logs into the system via that network.
- Sensitive information exposure: Because all system traffic is unencrypted, an attacker can observe all of the transactions and requests made to the system without directly accessing the system. For example, if a port administrator requested a schedule of dangerous goods while connected to a public network, then any device on that public network could also view that schedule.
- Password compromise: Instead of using a computationally-expensive, salted, one-way hash function, the system stores passwords using an insecure form of two-way encryption, including a decryption key stored locally on the server. In the case of a stolen or compromised database file (which may be made possible by weakness 4), an attacker could trivially decrypt the passwords stored in the database. This would lead to full compromise of all accounts and disclosure of users’ (potentially reused) passwords.

2. Improper authorization and authentication design allowed illegal access to the system’s database. As a consequence, the following issues arose:
    - Any user could change any other user’s password. By circumventing client-side validation, an attacker could request a password change for another user without providing a correct current password. This vulnerability was a result of faulty validation logic on the server.
    - Users could access unauthorized services by tampering with client-supplied request metadata. For example, an attacker could craft a request for Service A with metadata that indicated Service B. The server would authorize the request based on the metadata indicating Service B, but then invoke Service A. This is an example of a trust boundary violation; the server is trusting that the metadata from the client is consistent with the service request’s destination. Since client applications can easily be replaced or compromised, the server must assume it is untrusted. For this reason, any validation, authorization, or authentication performed by the client must also be rechecked by the server.

Design issues such as these are often the most expensive and time-consuming to fix.

3. Use of vulnerable versions of third-party software components exposed the system to existing exploits for those components. In any modern software system, third-party components such as framework libraries, operating systems, compilers, and protocols make up a large part of the *software supply chain*. Many of these components contain dangerous vulnerabilities that may compromise the systems depending on them. The presence of dynamic dependencies and non-standard update channels make it difficult to track vulnerable components.

4. Improper validation in custom file services allowed any user to modify or delete files throughout the server's filesystem. An attacker could generate a legitimate file download request using the client's user interface and then modify it to specify deleting, downloading, or overwriting any specific file on the server. This vulnerability was a result of both improper sanitation of the filename to prevent path traversal and lenient access control for the i/o services. Note that the combination of this weakness along with the password compromise vulnerability in weakness 1 would allow an attacker to steal the username and password for every user of the system.

6. Conclusions

In this paper, we surveyed the state of the art in cyber-security for container shipping ports, identified the main problems and current initiatives, and proposed a new research direction and approach for improving the security of our freight ICT systems. In that regard, we created a transactional description of shipping logistics to improve understanding of the electronic (and paper) documents that are involved in maritime freight shipping. We then applied the First Principles Vulnerability Assessment (FPVA) methodology to part of the TOS and PCS of a well-known software that manages maritime freight shipping and found several significant vulnerabilities in the code.

For the first time, we crossed the line (in red in Figure 1) between physical/information infrastructures security and code level software systems security in the maritime transport sector, which constitutes a gap and vulnerability itself, and represents an increasing concern for port and terminal operators. Our research tackled this issue for the first time in this domain.

In addition to continue finding vulnerabilities in TOS and PCS, and therefore narrowing the gap of cyber security of maritime transportation, we believe that our work should produce the first set of recommendations and guidelines for the maritime sector on low level cyber-security (code level) of their freight management systems.

7. References

[1] eurostat, "Freight Transport Statistics - Statistics Explained," April 2017. [Online]. Available: http://ec.europa.eu/eurostat/statistics-explained/index.php/Freight_transport_statistics. [Accessed 13 September 2017].

[2] A. Aylward, A. Fine, K. Mulder, L. Rainville, D. Hackett and D. Smith, "Port Performance Freight Statistics Program - Annual Report to Congress," U.S. Department of Transportation - Bureau of Transportation Statistics, Washington, DC, 2016.

[3] T. Bateman, "Police warning after drug traffickers' cyber-attack," 16 October 2013. [Online]. Available: http://www.bbc.com/news/world-europe-24539417. [Accessed 13 September 2017].

[4] Interactive Computing Laboratory, "Project Medusa," ENAC, 2014. [Online]. Available: http://lii-enac.fr/en/projects/medusa/. [Accessed 13 September 2017].

[5] "Project Mitigate," European Commission Community Research and Development Invormation Service (CORDIS), 2015. [Online]. Available: www.mitigateproject.eu. [Accessed 13 September 2017].

[6] International Maritime Organization, "International Ship and Port Facility Security (ISPS) Code," in *Conference of Contracting Governments to the International Convention for the Safety of Life at Sea*, 2002.

[7] International Maritime Organization, "Consideration and Adoption of Amendments to the International Convention for the Safety of Life at Sea," in *Conference of Contracting Governments to the International Convention for the Safety of Life at Sea*, 2002.

[8] D. Polemi, T. Ntouskas, E. Georgakakis, C. Douligeris, M. Theoharidou and D. Gritzalis, "S-Port: Collaborative Security Management of Port Information Sysems," in *2013 Fourth International Conference on Information, Intelligence, Systems and Applications (IISA)*, Piraeus, Greece, 2013.

[9] T. Ntouskas and N. Polemi, "S-PORT: "A Secure, Collaborative Environment for the Security Management of Port Information Systems"," in *2010 Fifth International Conference on Internet and Web Applications and Services (ICIW)*, Barcelona, Spain, 2010.

[10] N. Polemi and T. Ntouskas, "Open Issues and Proposals in the IT Security Management of Commercial Ports: The S-PORT National Case," in *27th Information Security and Privacy Conference*, Crete, Greece, 2012.

[11] T. Ntouskas and N. Polemi, "Collaborative Security Management for Port Information Systems," in *International Conference on e-Business,* Rome, 2012.

[12] International Maritime Organization, "Maritime Cyber Risk Management in Safety Management Systems," US Coast Guard; The Maritime Safety Committee, 2017.

[13] International Maritime Organization, "Interim Guidelines on Maritime Cyber Risk Management," 1 June 2016. [Online]. Available: http://www.gard.no/Content/21323229/MSC.1-Circ.1526.pdf. [Accessed 13 September 2017].

[14] B. Downs, "The Maritime Security Risk Analysis Model: Applying the Latest Risk Assessment Techniques to Maritime Security," in *Proceedings of the Marine Safety & Security Council Vol. 64*, 2007.

[15] M. H. Kang, M. Li, B. Montrose, A. Khashnobish, S. Elliott, M. Bell and S. Pieper, "Overview of the Security Architecture of the Comprehensive Maritime Awareness System," in *2009 IEEE Military Communications Conference*, Boston, 2009.

[16] Z. Yazar, "A Qualitative Risk Analysis and Management Tool - CRAMM," SANS InfoSec Reading Room, 2002.

[17] C. J. Alberts and A. J. Dorofee, "OCTAVE Method Implementation Guide Version 2.0," June 2001. [Online]. Available: https://resources.sei.cmu.edu/asset_files/UsersGuide/2001_012_001_51572.pdf. [Accessed 13 September 2017].

[18] "ISO/IEC 27005: 2011 -- Information Technology -- Security Techniques -- Information Security Risk Management," International Organization for Standardization.

[19] Joint Task Force Transformation Initiative Working Group, "NIST SP 800-30: Guide for Conducting Risk Assessments," September 2012. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf. [Accessed 13 September 2017].

[20] J. A. Kupsch, B. P. Miller, E. Heymann and E. César, "First Principles Vulnerability Assessment," in *Proceedings of the 2010 ACM Workshop on Cloud Computing Security*, Chicago, 2010.

[21] B. P. Miller, "Vulnerability Assessment of Open Source Wireshark and Chrome Browser," Defense Technical Information Center, Fort Belvoir, Virginia, 2013.

[22] "HTCondor," University of Wisconsin - Madison, [Online]. Available: https://research.cs.wisc.edu/htcondor/index.html.

[23] "Ministry of Shipping and Aegean," 2013. [Online]. Available: need_url_or_other_identifier.