

Security Vulnerabilities in Command and Control Interoperability

John A. “Drew” Hamilton, Jr., Ph.D., *member, IEEE*

Abstract - This paper will outline issues associated with communication system interoperability, the security vulnerabilities associated with improved interoperability, network simulation to support the evaluation of the effects of worm-induced packet storms and the experimental design, implementation and results using the OPNET simulation environment.

I. INTRODUCTION

Interoperability is such a challenge, that the security implications of enhanced interoperability are often not considered. Once coalition networks are established, the vulnerability of information systems may increase. Internal propagation of a worm with the characteristics of, for example, “Nimble” or “Code Red” can generate internal broadcast storms behind the network firewalls. There are significant limitations to applying simulation to information security issues. Most security vulnerabilities occur at the end points and evaluating the actual systems best assesses these vulnerabilities. However, network simulation is an obvious choice for evaluating the impact of primary and secondary packet storms on large, internal networks.

II. INTEROPERABILITY

Communication system interoperability, the ability of two or more systems or components to exchange data and use information [1], is inherently software-based. Interoperability implies the existence of diverse systems that need to exchange data and services. Much is written about “systems of systems.” System interoperability is what makes heterogeneous systems of systems a reality. All of these systems are composed of hardware and software. Hardware is not easily changed. Furthermore, fielded hardware systems often cannot be wholly replaced.

Diverse hardware-based communications systems require an overall software architecture in order to interoperate. As noted in IEEE Standard 12207.0-1996 *Software Lifecycle Processes*, software architecture describes the top-level structure of the over-arching system and describes the software components. Specifically, developers adhering to

the standard are required to develop and document a top-level design for the interfaces external to the software item and between the software components of the software item. This is an essential first step in achieving interoperability between any two systems.

In Defense applications, interoperability is seriously hindered by the sheer number of systems, standards, and system developers /procurers. Figure 1 illustrates some, though no means all, of the commands developing/fielding software-intensive communications systems.



Figure 1. *Some organizations involved in command system acquisition: DISA, NSA, three service C2 acquisition commands and ten combatant commands (USNORTHCOM not included).*

An exemplar “system of systems” is the Global Command and Control System (GCCS) as shown below in Figure 2 (with just its first tier applications). A key factor in the success of GCCS is the discipline to which interface standards are maintained by the Defense Information Systems Agency. GCCS is a non-trivial system with at least twenty-three systems exchanging information with the top-level GCCS application. Further complicating this the situation are the systems that feed the systems that feed GCCS, of which Figure 2 is only a partial illustration. There are literally hundreds of second tier systems that feed into the twenty-three first tier systems that feed into the top-level GCCS application. It should be stressed that the decomposition of GCCS results in a graph that is neither acyclic nor directed. Major Paul Summers’ GCCS decomposition efforts are further discussed in [2] and [3].

procedures and sophisticated tools to allow information exchange while protecting our national and allied data.”

III. INTEROPERABILITY & SECURITY RISK

As Admiral Blair noted, in the previous GCCS example, the interoperability issue was a policy issue, not a technical issue. Policy and technology together can be potent challenges. Consider a requirement US Pacific Command articulated needing a secure email system to exchange sensitive but unclassified information between Headquarters, Australian Theater and Headquarters, US Pacific Command as illustrated in Figure 4.



Figure 4 *Secure email in the Pacific*

The alternatives to secure emails were frequent, incredibly long facsimile messages exchanged over secure telephone lines. From a policy perspective, senior officers became suddenly reluctant to empower junior officers as de facto authorities for foreign release.

Unfortunately, even limiting data exchange to email applications increases risk to all stations enabled for interoperability. Viruses can spread through email so the more stations connected, the greater the risk for attack. Worms, malicious programs specifically designed to replicate across networks are continuing threat.

The Morris Worm, which attacked networked stations in October 1988, was extensively documented [6] and [7]. Although a well-documented phenomena, network vulnerability to worm attacks has only increased. New worms continue to proliferate and virus scanners are updated *after* the initial attacks. Antivirus responses while eventually effective are purely defensive measures.

Table 1 Lifecycle of a typical Nimda Worm variant

Stage 1	Local executable file infection (prepending)
Stage 2	Search for IP addresses from registry
Stage 3	Elevate privilege and execute on vulnerable remote computers
Stage 4	Compromise local security settings

Thirteen years later, the Nimda Worm demonstrated the vulnerability of modern email systems and web servers. The Nimda Worm was identified on 11 October 2001 and

was found to have a four-stage lifecycle as shown above in Table 1.

Consequently, it is straightforward to plan and execute a timed distributed denial of service attack (DDoS). Significant effects have been achieved through ad hoc, and sometimes amateurish attacks. King, Dalton and Osmanoglu [8] note that there are many variants of denial of service attacks such as:

- Programming mistakes that take 100% of CPU time.
- System memory usage may continually increase due to a memory leak.
- Malformed data requests such as Web requests or remote procedure calls (RPCs).
- Large packets such as email addresses and Internet Control Message Protocol (ICMP) requests.
- Non-stop network traffic User Datagram Protocol (UDP) and ICMP (broadcast storms and network flooding).
- Forging routing information or unresponsive connection requests.
- Incorrectly configured wiring, power, router, platform, or application.

The authors further note that the Computer Emergency Response Team (CERT) has documented more than 318 DoS attacks.

A distributed DoS is merely the employment of a distributed set of remote computers to intensify the impact of the attack. A DDoS attack launched in conjunction with an email-propagated worm with a specific military objective could successfully “jam” military computer networks at a critical time of the attacker’s choosing.

Consider the following attack scenario:

- Select desired date/time for denial of service attack initiation in worm payload.
- Study resources on the Internet to assist in the design of the worm.
- Use lessons learned from various white hat and black hat security sites to maximize surprise elements of worm.
- Use “human engineering” to ensure that worm is introduced undetected on as many target machines as possible.
- Reinforce DoS attacks remotely through any backdoors opened by the worm at h-hour.

Even if the target network is only affected for a few hours, it may be a militarily significant impact. It is a reasonable precaution to evaluate network vulnerabilities to DoS/DDoS from internal as well as external stations.

IV. NETWORK SIMULATION AND DDoS VULNERABILITIES

A research group at Auburn University undertook a serious analysis of one network on campus. After modeling the network in OPNET, one pair of students studied the predicted impact of increased multimedia traffic on the network while the other pair focused on evaluating the impact of DoS attacks on the network.

OPNET may be described as a communications-oriented simulation language. The name OPNET is derived from Optimized Network Engineering Tools. The single most significant aspect of OPNET is that it provides direct access to the source code coupled with an easy-to-use front end. This capability allows the introduction of multiple traffic sources, from PDFs, from network emulators and from observed traffic.

OPNET uses the following modeling hierarchy as shown in Figure 5 [9]. Node models were used to represent the CISCO 5500 switches found in Auburn's Broun Hall as shown in Figure 6.

Network Models	networks and subnetworks
Node Models	individual nodes and stations
Process Models	STD that defines a node

Figure 5 OPNET model hierarchy

Broun Hall has four layer three switches that handle all of the traffic between the internal computers. Three of these connect to a main switch via gigabit fiber optic cable. The main switch connects to the main campus switches via gigabit fiber optics as well.

The students modeled the switches as follows: Each wiring closet was represented by a single switch, each LAN by a single node, and RoTW represented the source/destination of all inbound/outbound traffic respectively.

The students used Sargent's work [10] as a guide to validate their model. Validating and verifying their model was the hardest part of their project. Beyond face validity, the students attempted to achieve historical validity by comparing actual inputs and outputs from the network of interest with the

Verification of a simulation is the process of assessing the degree to which the implementation transforms inputs into outputs *as specified by the model*. The ultimate verification

test is to model a known system and run the same sets of inputs through the actual system and the simulation. If the results are statistically the same, then you have reasonable assurance that you have implemented the model correctly. This was the approach used by the students using data collected by the College of Engineering. In the course of the experiment, the students learned first-hand that network monitoring is a non-trivial effort. In trying to verify the model, it was subsequently determined that the actual time scale of the network monitoring system was not properly understood by the system administrator. The students successfully verified that their model accurately represented the actual network for a given load.

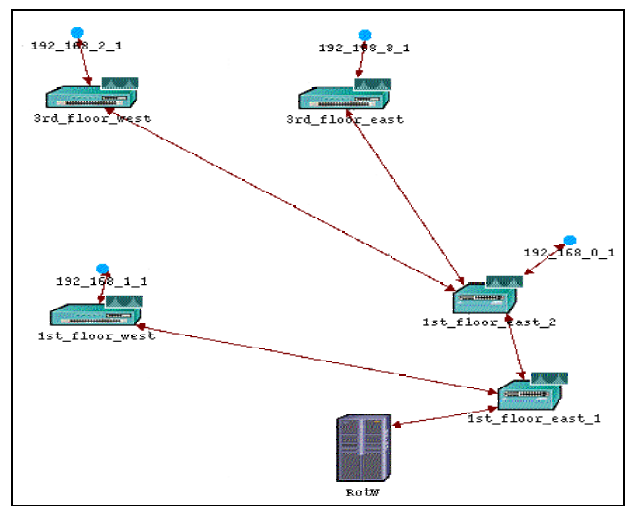


Figure 6 Topology of the Broun Hall network as simulated using OPNET [11].

Validation is the process that establishes the extent to which a model does (or does not) acceptably represent the phenomenon of interest. Once we know we have a good model, how do we gauge its predictive power? Generally a good traffic model is necessary for a network simulation to achieve any meaningful predictive power. Network traffic patterns cannot be relied upon to follow a standard probability distribution. Paxson and Floyd, among others, outline the failure of Poisson processes to model network traffic [12].

Accurate modeling of network traffic can limit many network simulation studies, but is not an issue in modeling DoS attacks. For a given worm, it is possible to exactly replicate the initiation time and traffic volume of each infected node. Network simulation is particularly appropriate for studying large-scale distributed DoS attacks since it is usually too costly to use large distributed systems for live DoS/DDoS testing.

Both groups wanted to see the effects of increased traffic on the network topology. They used the built-in traffic scaling

functionality of OPNET to scale traffic with 1000% and 5000% increases [13]. Increasing traffic beyond 5000% overwhelmed the network.

The network selected to study was very lightly loaded, usually averaging traffic of 3.5 megabits/sec percent loaded based on the data collected by the university network staff. The students observed that the Cisco switches had greater capacity than the gigabit links. The Cisco white paper regarding the Cisco 5500 switch, rates the switch at 50 Gbps maximum throughput. The technical specifications of those switches were found below at:

<http://www.cisco.com/univercd/cc/td/doc/pcat/ca5000.htm>.

It was clear that there were simply not enough gigabit links feeding into the switches to overload the switches, thus the throughput on the links was the theoretical throughput limit.

The teams reported the following results: Figure 7 shows that the throughput (bits/sec) maxes out at 1 Gbps.

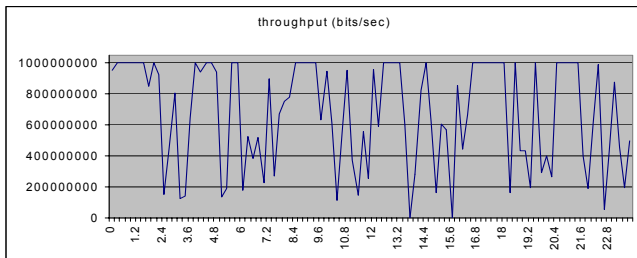
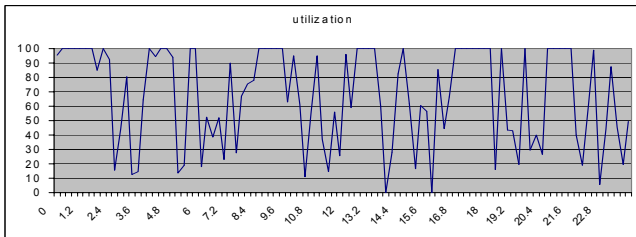


Figure 7 Throughput limited to 1 Gbps

This is expected because the gigabit link can have no higher capacity than its rated capacity of 1 Gbps. Consequently, in Figure 8 the group observed that utilization behaves in



direct proportion to maximum capacity.

Figure 8 Utilization in direct proportion to max capacity

Based on this study several conclusions were drawn regarding the vulnerability of the network to an internal DoS/DDoS attack. First it was observed that the network was generally very lightly loaded. Second it was observed that each switch had significantly greater processing capacity than a single gigabit connection. Given the current topology (to include the limited number of stations on the network), it would be very unlikely to that an internal DoS attack could overwhelm a switch. However individual link failures could be expected.

System interoperability is a force multiplier for command and control systems, but added capability increases risk. More connected stations increases the risk. Assessing the vulnerability of a network to DDoS attacks is prudent. This work demonstrated one practical means to evaluate such vulnerabilities using OPNET.

Although simulation can be used to study denial of service attacks, simulations themselves can be a source of vulnerability. Distributed or networked simulations in particular are vulnerable.

V. CONCLUSIONS

Those pundits who glibly extol the virtues of commercial-of-the-shelf (COTS) hardware and software to the Department of Defense should seriously reconsider their arguments. The same COTS hardware and software enables potential adversaries to rehearse attacks against DOD systems in the same manner that we used to assess our own vulnerability.

Modeling and simulation is clearly becoming a mainstream tool in many domains. However, the nature of computer/network security issues is such that often it is best to simply work with the actual systems rather than trying model those systems. Effectively modeling buffer overflow attacks requires a high fidelity model. Testing and evaluating the actual systems seems a better strategy.

However, network simulation is a mature, mainstream tool for the study and evaluation of networks. It is simply too costly to flood operational networks and measure the results. A validated network simulation is clearly a more practical means of evaluation.

System interoperability is a force multiplier for command and control systems. But added capability increases risk. More connected stations increases the risk. Assessing the vulnerability of a network to DDoS attacks is prudent. Students at Auburn University have demonstrated one practical means to evaluate such vulnerabilities using OPNET.

VI. REFERENCES

- [1] IEEE Standard 610.12-1990, *IEEE Standard Glossary of Software Engineering Terminology*, Institute of Electrical and Electronics Engineers, Piscataway, NJ, p 42.
- [2] Hamilton, John A., Jr., "Simulation to Support Security Issues Related to System Interoperability," *Summer Computer Simulation Conference*, July 14-18, 2002 • San Diego, California, USA

[3] Hamilton, John A., Jr., Melear, John, Endicott, George, "C2 Interoperability: Simulation, Architecture and Information Security," *7th International Command and Control Research and Technology Symposium*, Quebec, QC, Canada, 16 – 20 Sept. 2002.

[4] Summers, Major Paul, Director, Joint Forces Program Office, Space and Naval Warfare Systems Command, San Diego, Calif. Unpublished correspondence with the author.

[5] Blair, Admiral Dennis C., Commander-in-Chief, US Pacific Command, Statement before the US Senate Armed Services Committee, 27 March 2001.

[6] Spafford, Eugene H., "Crisis and Aftermath," *Communications of the ACM*, Association of Computer Machinery, New York, vol. 32, no. 6, pp 678 – 687.

[7] Rochlis, Jon A., Eichin, Mark W., "With Microscope and Tweezers: The Worm from MIT's Perspective," *Communications of the ACM*, Association of Computer Machinery, New York, vol. 32, no. 6, pp 689 – 698.

[8] King, Christopher M., Dalton, Curtis E., Osmanoglu, T. Ertrem, *Security Architecture*, RSA Press, Osborne/McGraw-Hill, New York, 2001, p 455.

[9] Hamilton, J.A., Jr., Nash, D.A., Pooch, U.W., *Distributed Simulation*, CRC Press, Boca Raton, Fla., 1997, pp 332 – 338.

[10] Sargent, Robert G., "Verification and Validation of Simulation Models," Winter Simulation Conference, Washington DC, December 13-16, 1998.

[11] Rouse, Ken, Tidwell, Chris, Term Paper, "Simulation of a DoS Attack on a Real System," COMP 8700, Simulation of Computer Networks, Auburn University, Spring 2002.

[12] Paxson, V. and Floyd S., "Wide Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Transactions on Networking*, no. 3 (June) 1995, pp 226 - 244.

[13] D'Amico, Jean, Taylor, Brandon, Term Paper, "Simulating Broun Hall: An OPNET Experiment," COMP 8700, Simulation of Computer Networks, Auburn University, Spring 2002.

his retirement from the US Army, he served as the first Director of the Joint Forces Program Office and on the Staff and Faculty of the United States Military Academy. CRC Press publishes his book, *Distributed Simulation*, written with Lieutenant Colonel David A. Nash and Dr. U. W. Pooch.

VII. AUTHOR

John A. "Drew" Hamilton, Jr., Ph.D., is an associate professor of computer science and software engineering at Auburn University. He has a B.A. in Journalism from Texas Tech University, an M.S. in Systems Management from the University of Southern California, an M.S. in Computer Science from Vanderbilt University and a Ph.D. in Computer Science from Texas A&M University. Prior to