# ARMY BATTLE COMMAND SYSTEMS (ABCS) SECURITY TESTING: A SYSTEMS-OF-SYSTEMS APPROACH

Sam Nitzberg and John Skrletts
NetCentric Technology, Inc.
Eatontown, NJ

## ABSTRACT

*This paper outlines and describes a successful approach used to perform Army Battle Command Systems (ABCS) security systems-of-systems testing. The methodologies used can be applied to systems-of-systems testing in general, and is not limited to supporting ABCS networks. In particular, three areas of critical importance to deployed systems are addressed by this model: (1) Operational benefits (or shortcomings) of the implemented defense-in-depth security model. Protections offered by services can be confirmed to be operating properly – or can be identified as deficiencies in the security architecture, (2) The networked devices themselves – do they possess intrinsic security shortcomings that must be mitigated?, and (3) Interactions in a deployed environment – since the actual deployed environment provides the model and basis for our testing, we can test real-world and complex attack-and-defense scenarios. The limitations of performing stand-alone black-box testing are not a constraint.*

## INTRODUCTION

Historically, the method of performing system security certification and accreditation testing has been performed on a per-system basis. This is particularly true for military environments that employ primarily black box testing and some level of white box testing techniques. While this type of approach is useful for assessing the individual system's security posture, it does not necessarily provide the system user a clear view of their overall risk in a networked environment. This approach does not allow one to take into consideration network protection mechanisms that may mitigate some of the risks identified in the individual systems.

Over a period of time, the Army, and in particular, Program Executive Office, Command, Control & Communications - Tactical (PEO C3T) has developed a methodology for assessing networked systems. The goal of PEO C3T has been to provide a methodology to: (1) Assess the security posture of systems in their intended network environment; (2) Evaluate the effectiveness of the network protection mechanisms; (3) Identify strengths and weakness of the security architecture in the networked environment; and (4) Develop programs and architectural changes to address any shortcomings identified.

This process has evolved greatly over the years through input from many supporting contractors and Army organizations. The process has now emerged into a mature process that has been adopted as the Army model. While it has evolved for complicated tactical Army networks, it is process that is equally applicable to all networks.

## WHITE BOX TESTING

White box analysis and testing is also known as " glass box, structural, clear box, open box analysis and testing." These methods make use of explicit knowledge of the internal workings of items to analyze systems and select test data [Wikipedia]. Systems requirements, designs, and / or specifications are necessary to accomplish white box testing.

There are a number of approaches to the employment of "White Box" methods. Code Reviews and Walkthroughs can be one aspect of White Box analysis [Nitzberg]. Source code reviews can be one tool when considering the use of open-source software. This can be perceived as a risk-mitigator in the absence of a party directly responsible for the quality, reliability, and security of the software.

Tests are built based on inspection (of specification, code, and interfaces), and analyzed to determine the degree of compliance with the designer's intent. The tests designed can then specifically examine test cases typical of expected values, as well as at the extremes of their ranges. Invalid and special-cases can also be specifically accounted for.

White box testing is generally not the preferred method for the examination of interacting entities. These methods only provide relevant data for Certification and Accreditation (C&A) efforts in a localized context – how specific systems [or subsystems] respond. The results provided are generally not relevant to defense-in-depth security con-

cerns, and these methods tend to be of little practical interest systems are deployed or ready for deployment.

## BLACK BOX TESTING

Black box testing, or functional testing, is used in computer programming, software engineering and software testing to check that the outputs of a program, given certain inputs, conform to the functional specification of the program. In black box testing or functional testing, the internal workings are not known to the test team. The test team only knows inputs and expected outputs, but not how they are arrived at [Wikipedia].

In the area of security black box testing, this step is usually accomplished primarily with vulnerability and port scanners. Sometimes, this testing is further supplemented with specific exploits targeted at known vulnerable applications. Ultimately, the effective results of these tests are dependant on the experience and skills of the test team. For example, if not all of the applications are running on the subject system, the vulnerability scanners will not discover them. If the team is unaware of an application vulnerability, the team may not run the correct supplemental tests to verify its existence.

So as one can see, the results from black box testing and white box testing may not result in a complete picture. However, assuming that these steps are complete and thorough, there is still a need for post testing analysis. After all, security is process of risk management and one can not realistically eliminate all vulnerabilities from a system. Therefore, someone must perform a risk assessment of the system and determine if the discovered vulnerabilities can be mitigated to an acceptable level. This process typically requires the evaluator to make assumptions about the environment that the systems will be employed. These assumptions include such factors as the systems interaction with other elements of the network, trust relationship models of the network, protection mechanisms provided by the network, etc. Since this is a paper analysis, the experience and skills of the evaluator will again play a major role in the "correctness" of these assumptions. Finally, the evaluator must apply a subjective analysis of these factors to determine their impact on mitigating the risks identified in testing.

## SYSTEMS-OF-SYSTEMS TESTING

Systems-of-Systems Testing (SOST) methodologies comprise a very powerful and practical applied set of methods for use in testing systems that are ready for deployment, or have already been deployed. These methods could even be judiciously applied pre-deployment to "shake-out" and identify weaknesses that were not previously identified in the greater scope of the projects, or which have appeared through errors. These methods are presently being used to evaluate the true bearing and impact of security exposures for networked systems.

SOST methodologies model in the lab the real-world relationships and configurations of systems. When funding, space, or schedules impose constraints, appropriate subsets of deployed systems-of-systems and families of systems can be stood-up in the lab. Comprehensive security testing often has destructive effects (even when not intended) and should be performed in non-operational and well-segregated environments.

To make effective use of SOST methodologies (in terms of systems analysis), an understanding of the security-relevant components must be achieved. Practical accommodations can also be made in order to facilitate in-the-lab testing. For example, a satellite channel between ground-based devices and ships-at-sea may be emulated by appropriate devices. As long as an appropriately accurate emulation is performed (for example, to render models of denial-of-service attacks meaningful), the substitution can be made in the lab in order to conserve resources.

Each component (computer systems, routers, firewalls, servers, etc…) that may be abused, undermined, or manipulated into furthering information security attacks should be present in the lab environment. An appropriate subset of these may have to be used based on practical constraints: funding, the time available to perform tests, availability of properly configured equipment, and any logistical exigencies.

Some of the benefits of this SOST approach include the operational benefits of addressing the benefits of security in real defense-in-depth situations. Attacks against exposed devices, systems, or network elements which are intrinsic to those items will be identified. Further, security tests can be performed against systems which are interacting - providing a sophisticated testbed against which to attack systems under both their normal operational modes, as well as under stress.

Types of attack that may be modeled include denial-of-service attacks, attempts to obtain unauthorized access to systems, and other cases. The tests themselves will leverage the results of earlier black-box testing results. The results of these tests will either validate or invalidate the underlying security architecture's actual implementation.

The degree to which the security mechanisms and risk mitigation steps are deployed in the proscribed architecture and successfully contribute to defense – in – depth may be determined by the use of SOST in the networked environment.

## KEY STEPS

First, an inventory of the hardware and software items present in the actual environment should be collected. This may be quite involved for networks with many types of systems, servers, and hardware. Appropriate hardware and software items of concern can be identified. These are the targets of opportunity. They may be further prioritized based on : (a) known hazards as a result of prior black-box testing, (b) their criticality in the network architecture, and (c) other potential measures of vulnerability. The focus of the greater testing effort may be reflected in assigning priorities to the developed test cases.

Hardware, software of interest is identified. Items not relevant to any security interests may be eliminated from the working model towards developing the test floor. Appropriate substitutions may be made to address practical realities associated with changing hardware and software baselines, the availability of personnel and equipment, and other software and developmental scheduling requirements.

Once the test environment is designed and laid out, tests are planned. For each test, a map of the relevant elements of the test floor should be identified, as well as the source of the attack and its location, the target of the attack (software, hardware, or both), required hardware or software to provide an appropriate environment for the test at-hand, the nature of the attack, and the expected result. With experience, a test team can perform the required tasks efficiently.

The focus of each test will vary and the priorities within any given test should reflect / dictate the priorities in test selection. Examples of such focus include : Denial-of-Service (DOS), vulnerability information exchange between networked systems, specific protection mechanisms, and other critical aspects of information security for deployed systems.

Teams are established: A designated “red” team will perform the actual attacks. Its responsibilities include obtaining conventional programs and scripts to attack systems and network devices, as well as preparing any customized attacks. Naturally, the precise steps to be taken by the red team will vary based on an organization's natural structure, teams, and priorities. A designated “white” team may be used to provide independent monitoring and recording of results. Additional teams may be utilized, as appropriate.

The test floor will have to be appropriately established and configured, and the tests performed. Attacks that systems are vulnerable to will be identified.

## AN EXAMPLE – NETWORK LAMBIC

An example is presented to demonstrate the practical use of this model and approach. This is presented only as a model for the purposes of example, and does not reflect any specific deployed system. Any similarity in name to any system or systems under development is non-intentional, and is purely coincidental.

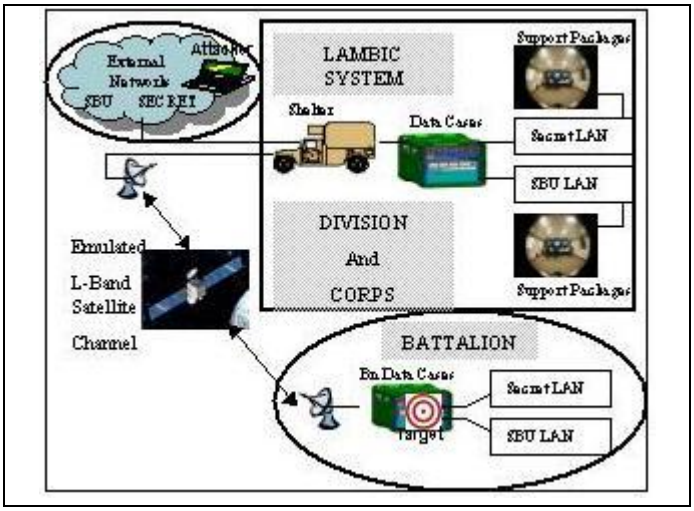


Figure 1. LAMBIC Test Floor Model

The model represented in Figure 1. shows a typical (but small) model for a systems-of-systems test floor.

Key items of interest shown are:

- The location of an attacking node: This attack is coming from an “External Network.”
- The location of the target: The device or software item being targeted is present in the test system environment items designated as “Data Cases.”
- The presence of emulated communications links.

A single test may be planned as follows:

| | |
|---|---|
| Objective: | Undermine router security |
| Source: | External Networks |
| Target: | Router in Data Case |

| | |
|---|---|
| Description: | Attack will be against router IOS |
| Procedure: | Use conventional scans and attacks. |
| | Attempt to authenticate to router |

A significant number of such tests may be produced (and categorized) to form a comprehensive systems-of-systems test.

## CONCLUSIONS

The techniques for performing a SOST discussed in this paper provide for a reliable, repeatable, and cost effective methodology. This method is reliable and repeatable since it provides a framework for identifying and documenting the tests to be performed. It is cost effective in that one tailors the test environment to adequately emulate the network of interest as well as focusing on the testing required for the specific objectives of the test.

From past experiences, the test results have proven extremely beneficial, providing valuable insights in keeping with the focus of each test event. The results have included:

a. Identifying that the security mechanisms' management systems actions were too difficult to correlate.
b. Validating the effectiveness of host based protection mechanisms.
c. Validating the effectiveness of perimeter based protection mechanisms in the network architecture.
d. Validating the effectiveness of the configurations of the perimeter protection mechanisms.
e. Identifying the need for better configuration management on host systems (i.e. patch management).

As one can see, the results were not always positive. In fact, many of the results were to identify items requiring improvement. PEO C3T has been evolving this methodology since 1999. The important trend in this testing approach has been a continuing improvement in the security posture of the tactical Army C2 networks.

## REFERENCES

Nitzberg, Sam, et. al., Improving Computing Security of DOD Computerized Weapons Platforms, National Information Systems Security Conference (NISSC), 1998

Nitzberg, Sam, et. al.,Trusting Software: Malicious Code Analyses, Milcom 1999.

Wikipedia, White Box Testing

Wikipedia, Black Box Testing