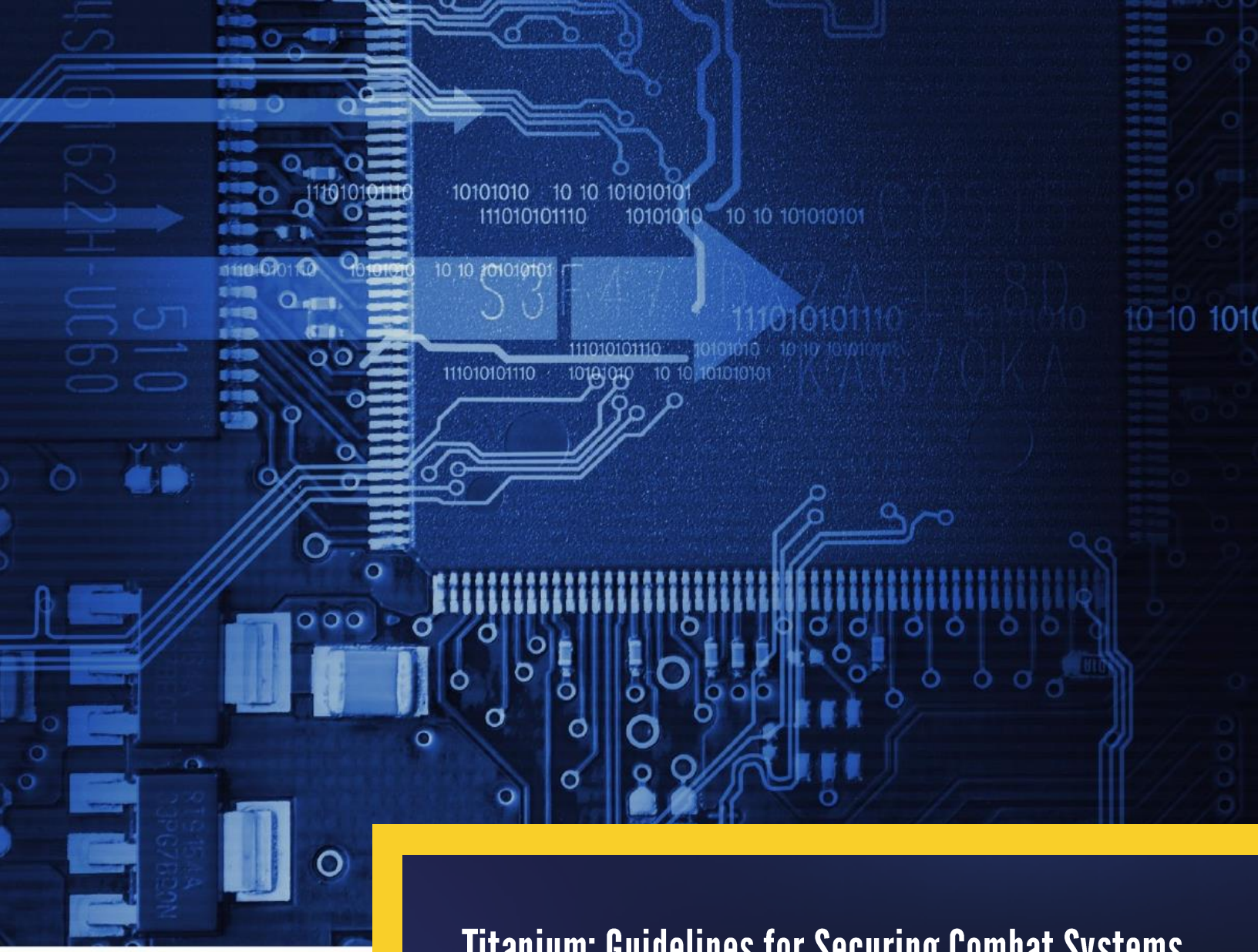

Titanium: Guidelines for Securing Combat Systems

STAR LAB SOFTWARE

Introduction

Cyber-attacks against mission-critical combat systems are a growing concern across the Department of Defense. Existing approaches to securing and defending these systems almost universally adopt methods used to protect enterprise systems such as: basic network firewalls, persistent threat monitoring and auditing. These solutions most often do not anticipate the real-time requirements, operational remoteness, and resource constraints often associated with combat systems.

Combat systems differ from enterprise systems in several other ways as well. First, combat systems overwhelmingly rely on the Linux or other non-Windows operating systems. Enterprise networks, on the other hand, are predominately comprised of Microsoft Windows hosts. Second, the number of authorized users is significantly smaller. Enterprise networks for large businesses or organizations have thousands of users; combat system users, i.e., individuals requiring a username / password, on the other hand, range in the dozens if they have users at all. This number shrinks further when one considers the requirements for concurrent users, i.e., individuals using the system at the same time. Next, the behaviors of combat systems can be characterized as less random. This, of course, being a function of the small number of users and the overall purpose of the system. Fourth, combat systems, large and small, have the risk of being lost to the enemy during overseas operations or in battle. As a result, system designers cannot approach security with only a remote attacker in mind. Instead, security must include technology protection and anti-tamper solutions.

Taking these differences into account, designers of a security solution for combat systems should consider solutions that follow the guidelines below.

Use what is available

Combat system security is often improved by adding proprietary security software. These solutions implement security and anti-tamper by modifying important system calls (a technique often referred to as hooking). This approach significantly increases the cost of maintenance, testing, and integration as it effectively changes the Linux kernel in unanticipated ways—ways the Linux open source community is unable to support long-term. As a result, these products require revision with the release of every new Linux release.

In contrast to this approach, there exist several frameworks and auditing components supported by the Linux open source community that can be leveraged for a variety of security purposes, e.g., monitoring, data protection, etc. These components, such as Linux Security Modules (LSMs), auditd, and overlay file systems can all aid in enhancing the security of combat systems.

Eliminate and de-privilege users

The majority of user accounts on combat systems are often used for testing and configuration prior to fielding. These accounts can be removed prior to installation / deployment. The root or administrator account can also be de-privileged to de-incentivize privilege escalation attacks. Privilege escalation attacks pose a significant security risks to both combat systems and enterprise systems. These attacks take place once an intruder gains an initial foothold on a target system. Their purpose is to obtain privileges

equal to the privileges of the root or administrator account in order to establish persistent access, collect intelligence, or to cause an effect. This is often accomplished by exploiting a bug, design flaw, or configuration oversight in an operating system or application.

De-privileging the root / administrator account means an intruder cannot use the elevated privileges given to the root / administrator account to by-pass standard discretionary access controls. Effectively, an intruder who successfully executes a privilege escalation attack will not be allowed to delete other users' files, view sensitive information, or deploy malware with system-level access.

Employ mandatory access controls

Discretionary access controls, however, are limited. They give users the power to set the access controls for their files. Unfortunately, it is more likely that users will set overly permissive controls on their files -- something that weakens the overall security of the system. Additionally, discretionary access controls are not fine grained enough to provide for more complicated control scenarios. While this may be necessary in enterprise environments, combat systems often do not require this flexibility.

Mandatory access controls, in contrast to discretionary access controls, give control of file access to the operating system, overriding any users' controls. This means a malicious or compromised process running as an authorized user with permissions to access a resource or modify a configuration file, can still be denied access. This is particularly important and achievable with combat systems since malicious or compromised processes are a greater threat than users.

Embed anti-tamper protections

As was previously mentioned, combat systems are unique in that they come in close contact with adversaries. Sometimes this contact takes place during battlefield operations, but it can also take place through foreign intelligence operations or the inadvertent misplacing of smaller systems like lost mobile devices. One of the most well-known examples of this close contact is the 2001 China-US Aircraft Collision Incident. In April of 2001, a U.S. Navy EP-3E (a signals intelligence aircraft) made an emergency landing in China's Hainan province after colliding with a Chinese fighter aircraft. After the EP-3E landed the crew was detained and the Chinese thoroughly stripped and examined (reverse engineered) the aircraft's equipment. While the disassembled aircraft was eventually returned, the event highlights the need for combat systems to use anti-tamper technologies to protect hardware and software.

Software protections range in capability and intrusiveness. Sound approaches to software protection include just-in-time encryption/decryption of binaries, disabling debugging, and proper key management. Other important elements of a good software protection plan include minimizing or encrypting logging, denying core dumps, and limiting the use of untrusted memory.

Star Lab's Titanium Security Suite for Linux

Star Lab's Titanium Security Suite for Linux follows the above guidelines while also addressing 100% of technical RMF security controls. Titanium leverages several Linux-provided security components and supports both standard (i.e. RedHat) and vendor-specific Linux distributions for Intel and ARM.

Additionally, Titanium ensures that even privileged users (or processes) cannot interact with or access the memory contents of a protected application. The mandatory access controls implemented within Titanium deprivilege the root / administrator user, ensuring that combat systems can be configured and deployed as single-purpose systems. Additional controls to protect access to applications, data, and configuration files are also easily defined using Titanium's feature-rich tooling.

Finally, Titanium keeps applications, libraries, and data encrypted at rest. Titanium employs just-in-time decryption to verify and decrypt applications, libraries, and data before they are executed or accessed. This ensures applications or libraries that have been modified will be prevented from being executed. Titanium also prevents debuggers from accessing applications, memory segments, and libraries. Titanium even keeps adversaries from copying or removing protected applications from the system.