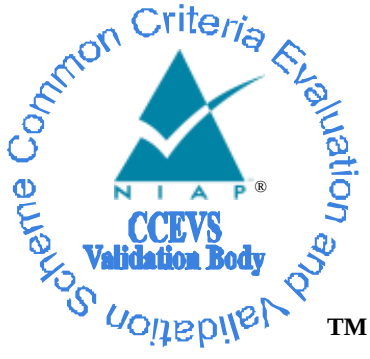
National Information Assurance Partnership



TM

Common Criteria Evaluation and Validation Scheme Validation Report

Wind River

Wind River Linux Secure 1.0

Report Number: CCEVS-VR-VID10430-2011

Dated: 2011-04-05

Version: 1.0

National Institute of Standards and Technology
Information Technology Laboratory
100 Bureau Drive
Gaithersburg, MD 20899

National Security Agency
Information Assurance Directorate
9800 Savage Road STE 6940
Fort George G. Meade, MD 20755-6940

ACKNOWLEDGEMENTS

Validation Team

Daniel Faigin
The Aerospace Corporation
El Segundo, California

Jerome Myers
The Aerospace Corporation
Columbia, Maryland

Evaluation Team

Jeremy Powell
atsec Information Security Corporation
Austin, Texas

Andreas Siegert
atsec Information Security Corporation
Munich, Germany

Table of Contents

1. EXECUTIVE SUMMARY

This report documents the NIAP validators' assessment of the evaluation of Wind River Secure Linux 1.0. It presents the evaluation results, their justifications, and the conformance results. This validation report is not an endorsement of the information technology (IT) product by any agency of the U.S. Government and no warranty of the IT product is either expressed or implied.

The evaluation was performed by the atsec Information Security Corporation, and was completed during March 2011. atsec Information Security Corporation is an approved NIAP Common Criteria Testing Laboratory (CCTL). The evaluation was conducted in accordance with the requirements of the Common Criteria for Information Technology Security Evaluation, version 3.1. The information in this report is largely derived from the Evaluation Technical Report (ETR) and associated test report, both written by the CCTL. The evaluation determined the product to be **Part 2 extended, Part 3 conformant**, and to meet the requirements of **EAL4 augmented by ALC_FLR.3**. Additionally, the TOE was shown to satisfy the requirements of the U.S. Government Protection Profile for General-Purpose Operating Systems in a Networked Environment (version 1.0, 2010-08-30) (GPOSPP).

Wind River Linux Secure is a general purpose, multi-user, multi-tasking Linux based operating system. It provides a platform for a variety of applications in the governmental and commercial environment. Wind River Linux Secure is available on a broad range of computer systems, ranging from multi-processor servers to embedded platforms.

The validation team agrees that the CCTL presented appropriate rationales to support the Results of Evaluation presented in Section 4, and the Conclusions presented in Section 5 of the ETR. The validation team therefore concludes that the evaluation and the Pass result for Wind River Linux Secure 1.0 is complete and correct.

2. IDENTIFICATION

The Common Criteria Evaluation and Validation Scheme (CCEVS) is a National Security Agency (NSA) effort to establish commercial facilities to perform trusted product evaluations. Under this program, security evaluations are conducted by commercial testing laboratories called Common Criteria Testing Laboratories (CCTLs) using the Common Evaluation Methodology (CEM) for Evaluation Assurance Level (EAL) 1 through EAL 4 in accordance with National Voluntary Laboratory Assessment Program (NVLAP) accreditation granted by the National Institute of Standards and Technology (NIST).

The NIAP Validation Body assigns validators to monitor the CCTLs to ensure quality and consistency across evaluations. Developers of information technology products desiring a security evaluation contract with a CCTL and pay a fee for their product's evaluation. Upon successful completion of the evaluation, the product is added to NIAP's Validated Products List.

Table 1 provides information needed to completely identify the product, including:

- The Target of Evaluation (TOE): the fully qualified identifier of the product as evaluated;
- The Security Target (ST), describing the security features, claims, and assurances of the product;
- The conformance result of the evaluation;
- The Protection Profile to which the product is conformant;
- The organizations and individuals participating in the evaluation.

Table 1: Evaluation Identifiers

Item	Identifier
Evaluation Scheme	United States NIAP Common Criteria Evaluation and Validation Scheme
Target of Evaluation	Wind River Linux Secure 1.0 running on one of the following platforms: • Dell D630 (Intel Core 2 Duo processor) • Intel 'Hanlan Creek' Dual Processor Xeon 5500 Series Pedestal Server Motherboard (S5520HCR) (using Intel Nehalem processor) • PPC_32 MPC8572DS (using Freescale MPC8572 PowerPC 32 bit processor) • ARM TI OMAP3530 (using ARM Cortex-A8 processor) • SolCORE ITAR-restricted board
Protection Profile	U.S. Government Protection Profile for General-Purpose Operating Systems in a Networked Environment
Security Target	*Wind River Linux Secure 1.0 Security Target v 1.17, 2011-04-05*
Evaluation Technical Report	*Evaluation Technical Report for a Target of Evaluation* *Wind River Secure Linux 1.0* *ETR Version 4.0 as of 2011-04-05*
Conformance Result	CC V3.1, Part 2 extended, Part 3 conformant, EAL 4 augmented by ALC_FLR.3, and GP-OSPP compliant
Sponsor	Wind River
Developer	Wind River
Evaluators	Jeremy Powell and Andreas Siegert atsec information security corporation
Validators	Daniel Faigin and Jerome Myers The Aerospace Corporation

3. CLARIFICATION OF SCOPE

This section details the scope of the evaluation and describes the logical and physical boundaries of the TOE.

3.1. Physical Scope

The physical scope of the evaluated configuration consists of:

- Software:
  - Wind River Linux Secure version 1.0
- Hardware (one of the following):
  - Dell D630 (using Intel Core 2 Duo processor)
  - Intel 'Hanlan Creek' Dual Processor Xeon 5500 Series Pedestal Server Motherboard (S5520HCR) (using Intel Nehalem processor)
  - PPC_32 MPC8572DS (using Freescale MPC8572 PowerPC 32 bit processor)
  - ARM TI OMAP3530 (using ARM Cortex-A8 processor)
  - SolCORE ITAR-restricted board
- User documentation:
  - *Wind River EAL4 Evaluated Configuration Guide for WindRiver Linux Secure 1.0, April 5, 2011; v1.7*
  - *Wind River Linux Secure, Administrator's Guide version 1.0*
  - *Wind River Linux Secure, Configuration Guide version 1.0*
  - Linux manual pages describing usage of all interfaces

The *EAL4 Evaluated Configuration Guide for WindRiver Linux Secure 1.0* is the authoritative documentation that must be used in order to place the TOE into the evaluated configuration. In case of any contradictions with other documents, it supersedes all. It explains how to install, configure and administrate the TOE. Moreover, it provides explanations about the intended environment. It does rely on the Wind River Secure Configuration Guide, as it points to specific instructions on the configuration of the file system.

Note: Although the *Wind River Linux Secure, Protection Profiles Policy Guide version 1.0* ships with the product, it is an obsolete document and must be ignored during the installation of the TOE in the evaluated configuration. End users are informed of this with a slip of paper enclosed with the CD, as it was not possible to adjust the timing of CD manufacturing to remove the document.

3.2. Logical Scope

The description of the security features of the product are described in further details in Section 4. In summary, these functions are:

- Discretionary Access Control
- Mandatory Access Control
- Identification and Authentication
- Auditing
- Object Reuse
- Cryptographic Services
- TSF Management
- TSF Protection
- Resource Utilization
- TOE Access

The validation team has identified several features that, without clarification, may be considered as part of the evaluated product, even though there are no security claims against them and therefore not examined by the evaluation team. The following features are shipped with the TOE, but were not evaluated:

- There are several cryptographic modules found throughout the product that have not been considered in this evaluation nor validated according to FIPS 140-2. The vendor has asserted the correctness of these modules with the following statement in the security target:

  > "Cryptography in the product that is not related to the security functional requirements is not covered by the evaluation and their validity is vendor asserted. This applies to all applications not linked to the FIPS 140-2 validated NSS library provided with the Wind River Cryptographic framework. Examples identified during the evaluation include: OpenSSL, *beecrypt*, *gnutls*, *gnupg*, *duplicity*, OpenSSH, eCryptFS, IPSec, *stunnel*, *samhain*, and the *crypt()* function."

- GRSecurity is a hardening set of patches for the kernel to mitigate some common vulnerabilities. It is active in the product, but has not been evaluated.

- PaX is a hardening set of patches for the kernel to mitigate vulnerabilities associated with buffer overflow attacks. It is active in the product, but has not been evaluated.

- Linux kernel packet filtering is active in the product, but has not been evaluated.

- The Linux Kernel Module subsystem is active in the product, but has not been evaluated.

- BSDJail, a Linux Security Module providing hardened chroot jails, is shipped with the product, but has not been evaluated.

- SELinux Type Enforcement allows administrators to developer policies based on the types of subjects and objects and the transitions between these types. However, this portion of SELinux is not part of the evaluation.

- Mail server functionality is shipped but not active in this product, and has not been evaluated.

- The Ext2, configfs, and NFS file systems are shipped in the product, but have not been evaluated.

- SMACK, a replacement for SELinux, is shipped but not active in this product, and has not been evaluated.

The validation team also identified a number of features that do not ship with the product. However, as they are commonly found in a standard Linux operating system environment, the validation team felt it was useful to clarify their status:

- Printing support is not shipped with this product.

- Kernel key retention services are not shipped with this product.

- *rnano* and *vsftp* are user applications that are not shipped with this product.

Full details on the packages that are included with the TOE, as well as specifics of what is covered, may be found in the Security Target.

4. SECURITY POLICY

4.1. Security Policy Model

Wind River Linux imposes discretionary and mandatory access controls on users of the system, restricting what actions they may take on objects.

Subjects that act on behalf of users in this model are considered to be processes running on the system. These processes, through kernel-mediated system calls, are granted restricted rights to read, modify, execute, create, and delete objects.

The objects controlled by the kernel are named objects. These objects can be referenced globally by users and can possibly be shared among processes. The objects include general data files, system information and configuration files hosted by several non-storage-backed filesystems (i.e. procfs, sysfs, binfmt_misc, securityfs, and selinuxfs), device special files, inter-process communication (IPC) channels (i.e. semaphores, shared memory, message queues, named pipes, and sockets), and batch processing queues.

4.2. Discretionary Access Control

Wind River Linux Secure implements Discretionary Access Control (DAC) through the use of standard UNIX permission bits and the POSIX standard Access Control Lists (ACLs). A Discretionary Access Control policy requires mechanisms whereby the access of users (i.e., subjects) to system resources and data (i.e., objects) can be controlled on the basis of user identity, role, and explicit permissions. Mechanisms that implement a DAC policy provide the capability for users to specify the how their personal data objects are to be shared.

Permission bits are associated with objects and specify the permissions (typically, READ, WRITE, EXECUTE) for a specific user, the user's group affiliation, and all others (i.e., "world"). POSIX Access Control Lists provide the ability to specify traditional Unix access controls, but extend that functionality to support the ability to permit or deny access at the level of an individual user or group. They are considerably more flexible in that they can identify a number of group affiliations for a single user.

The standard UNIX DAC mechanism is permission bits, as is the case with Wind River Linux Secure. However, Wind River Linux Secure implements ACLs as an extended permission mechanism, available at the discretion of the file owner. Although ACLs are used to control the access to file-system based objects, inter-process communication channels are protected through only UNIX permission bits.

4.3. Mandatory Access Control

Wind River Linux Secure implements Mandatory Access Control (MAC) through the use of labels maintained for processes and storage objects maintained by the kernel. The MAC policy implements the rule-set based on the Bell-LaPadula model.

In order to provide a robust computing environment certain actions require the ability to break the Bell-LaPadula model to allow information to flow from high sensitivity labels to lower ones. Certain labels can be given to subjects and executable files that assign override attributes, allowing these actions to be taken. These labels, and the subjects and objects who obtain them, are considered to be trusted processes, and therefore were evaluated as part of the TSF.

In order to limit the possibility of inadvertent information leaks, the TOE implements poly-instantiated directories. This mechanism separates globally readable directories into segments that are only accessible by a particular security context. Even a single user who opens a sessions in one sensitivity level is not allowed to access files (or know of their existence) from the perspective a session in a different sensitivity level. The poly-instantiated directories in the system are: /tmp, /dev/shm, and all home directories.

Labeled networking is provided with the TOE through the use of IPSec.

4.4. I&A

Each user must have a unique identity (i.e., username plus password), and be authenticated prior to obtaining resources and services from the TOE. Note, however, that in a networked environment, user identities are unique to a server, and are neither known globally nor are universally unique. That is, each server maintains a server-specific set of users and their associated passwords and attributes. A user that has access to more than one server on a network will have a different user identity, and possibly different attributes, on each server for which access is authorized.

Users can change their own passwords. However, an administrator can define expiration and quality constraints on the users' passwords. The default password quality metrics require the following:

- Password verification is case sensitive.
- Passwords must be a minimum of 16 characters.
- Passwords must contain a mix of at least one upper-case, one lower-case, one number, and one special character.
- Passwords must differ by a minimum of 5 characters each time it is changes.
- Passwords expire every 60 days.
- Passwords cannot be changed more than once in a 24-hour period.

The evaluated configuration guide provides instructions on configuring password parameters to meet the requirements of the IAIA-1 control in DOD 8500.2 or the IA-5(1) control in NIST SP 800-53 as completed by CNSS 1253. This adjustment may be made without violating the evaluated configuration.

4.5. Auditing

The TOE audit mechanism allows the generation of audit records for security-related events. It allows the administrator to configure the audit mechanism to collect the events that are to be captured and specify the users that are to be audited. It is also possible for the administrator to identify specific users that are not to be audited.

Each audit record contains event-specific information, and identifies whether the request that caused the event was successful or failed, and. An audit record consists of a standard header that includes the following information:

- A unique audit identifier.
- The LoginID of the user who caused the audit record to be generated.
- The Effective User ID of the user at the time the record was generated.
- Date and time the audit record was generated.
- Type of event.

Audit records are stored in ASCII format, and can be searched through the use of the standard Unix/Linux ASCII processing tools and the *ausearch* tool. If the audit trail storage becomes full, an alarm generated by the TOE can be configured to generate a syslog message or trigger the execution of an administrator-specified application. This message or action of executing the application is generated when the audit trail capacity exceeds the limit defined in the auditd.conf file.

4.6. Residual Information Protection

Although the TOE supports several different types of objects, each is managed by the system such that no pre-existing content is provided to users to whom objects are allocated. That is, whenever an object (e.g., buffers, memory extents, disk space) is allocated to a user process, it is managed such that any data that had previously been in the object (i.e., from an earlier process) is unavailable to the new process.

Memory pages are initialized to all zeroes when allocated to a process, Inter-Process Communication (IPC) objects are also initialized to all zeroes, file system objects are created with no content (with the exception of directories and symbolic links).

This is also applied to data structures allocated to subjects to ensure that object reuse also applies to memory that represents the subject.

4.7. Cryptographic Services

The TOE provides the user access to a FIPS 140-2 validated cryptographic module, the Network Security Services library (NSS). The module has been validated on all platforms in the evaluated configuration. The certificate numbers for this module are 1475 and 1506.

The module is accessed by unprivileged users through a trusted cryptographic service daemon, the Wind River Cryptographic Framework (WCF). This framework provides an IPC-based interface that allows applications to perform trusted cryptographic functions with an instance of the NSS library.

This module has been validated to provide the following NIST approved cryptographic functions:

- SHA-1 (128 bit digests).

- SHA-2 (224, 256, 384, and 512 bit digests).
- Elliptic Curve Digital Signature Algorithm (P-256, P-384, P-521 NIST approved curves).
- Advanced Encryption Standard (CBC and ECB modes, with 128, 192, and 256 bit keys).
- FIPS Publication 140-2 Annex C random bit generator, seeded with a minimum of 440 bits of hardware-based entropy.

With these cryptographic primitives, the TOE provides the user with the ability to generate symmetric and asymmetric keys, cryptographic signatures, and cryptographic digests for integrity checking. Also, since the NSS library is FIPS 140-2 validated, key destruction is implemented in accordance to FIPS Publication 140-2.

4.8. TSF Management

Management of TSF is done by administrators, as well as users in certain cases. There are generally two types of TSF management that can be performed. The first type are actions that can only be performed by an administrator. These actions include managing the functionality of the audit subsystem and accessing audit records, managing the mandatory access control subsystem including object attributes, managing authentication requirements including password quality and account lockout thresholds, defining default TSF attribute values, and changing object ownership.

The second class of actions can be performed by both administrators and certain users. These actions include changing discretionary access control TSF attributes of objects, and editing a user's own authentication data.

These restrictions are implemented in a large part by the discretionary access control on the configuration files that hold these attributes. These files are owned by the administrator and cannot be read or written to by regular users. It is important to note that mandatory access control is superfluous for the restriction of these configuration files, as discretionary access control is entirely sufficient.

Other restrictions, such as users being allowed to change their authentication data, are prevented by TSF trusted programs that act on the users behalf. These programs cannot be changed by regular users, so their actions can be trusted. In some cases, particularly when changing the TSF attributes of objects directly, these restrictions are implemented by the kernel directly during the access request of a system call.

User security attributes and access permissions can be revoked by administrators and in some cases owners of objects. Revocation of user access attributes are enforced at the next login, whereas revocation of object access permissions is enforced at the next access check.

4.9. TSF Protection, Resource Utilization, and TOE Access

Several security functions required by the GPOSPP are based on the assumption that a single instance of the TOE may be distributed across several platforms. These functions deal with consistency and integrity monitoring of TSF data. The evaluated configuration of this TOE does not support such a configuration, and have therefore been claimed as trivially met.

In the case of a failure of service that may lead to a violation of the security policy, the TOE provides the administrator with the opportunity to manually recover the TOE before it is allowed to go back into service. This is implemented in several points throughout the system, but most importantly, the boot sequence is designed in such a way that, when the machine reboots due to a failure, the systems enters a maintenance mode. This mode does not spawn any services automatically, so access is restricted to the physical console, requiring the administrator's attention before proceeding into service.

A separate point where manual recovery is required is during the initialization of the FIPS 140-2 cryptographic module at the start up of the TOE. If the module fails tests demonstrating its correctness, the TOE enters the same maintenance mode entered during reboot. The administrator then has the opportunity to investigate the issue before re-deploying.

The system provides a reliable time stamp that cannot be tampered with by regular users. This provides assurance to administrators about the validity of the audit records.

Individuals are restricted to an administrator defined portion of persistent storage. This is implemented through the quota subsystem of the kernel, which performs limit checks on the usage of disk space.

When users access the TOE, an administrator defined access banner is displayed to the user. By default, it provides a consent warning regarding unauthorized usage of the TOE. As users log off or log on, their usage history is stored and a recent access is displayed to a user during the initialization of an interactive session. Finally, during interactive sessions, the TSF or the user can initiate locking of the screen. The TSF will automatically lock the screen during an administrator defined amount of inactivity. Users can lock and unlock the screen manually as they wish.

5. ASSUMPTIONS

The evaluation makes the following assumptions on the TOE environment and personnel managing the TOE:

- The TOE is afforded physical security so that it may trust users accessing its hardware and its physical console.
- The organization managing the TOE has established policies that assign appropriate clearance to users and assigning sensitivities levels to all user information placed in the charge of the TOE.

- The organization managing the TOE must provide a secure way for the TOE to connect to peripherals and network devices.

6. ARCHITECTURAL INFORMATION

The TOE is implemented with an architecture that prevents bypass or tampering of the TSF. This is accomplished through by the implementation of security domains through virtual memory address spaces and hardware-based privilege levels.

Each security domain on the TOE operates within a single virtual memory space. Each block of memory, or page, is mapped to a single physical memory page file. When a process accesses a virtual memory address, the processor computes the correct physical memory location through the use of memory mappings. These memory mappings are considered to be extremely sensitive, as inappropriate tampering would allow a malicious subject to map other subjects' memory into its own address space, allowing him full access. Therefore, the entire task of managing these memory mappings is reserved to the kernel.

A second major architectural feature is hardware-based privilege levels. These levels are part of the processor state at any given time; the transition from one level to another is enforced by the hardware and configured by the kernel. In this TOE, there are only two levels: *ring zero*, which indicates the processor is running in the privileged "kernel" state; and *ring three*, which indicates the processor is running in the unprivileged "user" state. When in the user state, a process is limited to the hardware instructions available to it, including instructions to write directly to devices or to configure processor registers.

In order for user processes to operate effectively, they will eventually need to access sensitive data structures and devices. Since they cannot directly access them from their hardware privilege level, they may delegate the sensitive function to the kernel through the use of system calls. System calls pass to the kernel a process's intent to perform a security sensitive operation. This intent is encoded in the system call number and data structures handed to the kernel. When the kernel receives this information, it will perform these actions on behalf of the process. This shows that security domains can only interact directly with the kernel. All communication channels with other processes and devices are managed by the kernel, and therefore can be effectively controlled through the kernel's security policy.

The kernel is made up of numerous subsystems. Each subsystem implements some kind of interaction with the rest of the TOE. Some subsystems handle some system calls directly. These subsystems include the virtual file system, the memory management subsystem, the scheduler, and the IPC subsystems. Often, these subsystems provide a common interface to lower-level subsystems that perform various actions. This is most clearly exemplified by the virtual file system.

The virtual file system surfaces an interface to user processes that are common to accessing all kinds of files. These operations, such as *open(), close(), read(), write(),* and *stat()* are then passed through the virtual file subsystem down to the drivers that implement specific file systems, such as ext3fs

and *tmpfs*. Continuing downward, these real file systems interact with virtual disk drivers, which then interact with real disk drivers. This modular approach allows the kernel to be extensible without changing or cloning very sensitive pieces of machinery that must be scrutinized during an evaluation.

Now that processes have the ability to interact with the kernel and one another, a user space is constructed to facilitate a general computation environment. This includes many user-space subsystems that implement or at least manage authentication, auditing, access to the TOE, configuration of services, and more. Each of these subsystems relies on the security features implemented inside the kernel to prevent unauthorized access to sensitive TSF data.

In user-space, privileged processes are processes that run with administrative user identifiers. Unprivileged user processes are disallowed access to certain administrative data and mechanisms by controlling of the transitioning of user identifiers. In some cases, it may be needed for a user to assume temporarily administrative authority in order to complete an action. These actions are controlled by restricting the transition only when certain trusted programs are run. This mechanism in the TOE is done through "setuid" bits associated with executable files. When a subject executes the program (which it cannot modify), it temporarily assumes the user identifier of the root user to perform the action. Once this is complete, the process is either killed, or the trusted program lowers the privilege state back to the subject's default privilege state.

Through these mechanisms, the TOE controls interactions and thus prevents tampering and bypass of security functionality. This allows the TOE to provide the user with it security functions with the assurance that, because they are properly implemented, their implementation cannot be changed or bypassed.

The TOE implements several different security mechanisms within the following subsystems:

- Kernel Subsystems:
    - File and I/O management
    - Process control management
    - Inter-process communication
    - Network subsystem
    - Memory management
    - Auditing
    - Kernel modules
    - Device drivers

    - SELinux security module
- User Space Subsystems
    - System initialization
    - Identification and authentication
    - Network applications
    - System management
    - Batch processing (at and cron)
    - Audit management
    - SELinux management
    - Wind River Cryptographic Framework
- Hardware Subsystems
    - Firmware
    - Hardware

7. PRODUCT TESTING

7.1. Sponsor Testing

7.1.1. Test configuration

The test results provided by the sponsor were generated on the following systems:

- Dell D630 (using Intel Core 2 Duo processor)
- Intel ‘Hanlan Creek’ Dual Processor Xeon 5500 Series Pedestal Server Motherboard (S5520HCR) (using Intel Nehalem processor)
- PPC_32 MPC8572DS (using Freescale MPC8572 PowerPC 32 bit processor)
- ARM TI OMAP3530 (using ARM Cortex-A8 processor)
- SolCORE ITAR-restricted board

The sponsor has performed his tests on the above listed hardware platform. The software was installed and configured as defined in the Configuration Guide with additional software packages identified in the Test Plan.

7.1.2. Testing approach

The Test Plan provided by the sponsor lists test cases by groups that reflects the mix of sources for the test cases. The mapping provided lists the TSF/TSFI with which the test cases were associated. The Test Plan is focused on the security functions of the TOE and ignores other aspects typically found in developer test plans. The test cases are mapped to the corresponding Functional Specification and High-Level Design (HLD).

The sponsor uses several test suites that are integrated into one test system that includes automatic and manual tests to test the TOE.

The LTP test suite is an adapted version of tests from the Linux Testing Project. The LTP tests have a common framework in which individual test cases adhere to a common structure for setup execution and cleanup of tests. Each test case may contain several tests of the same function, stressing different parts (for example, base functionality, behavior with illegal parameters and reaction to missing privileges). Each test within a test case reports PASS, OK or FAIL, and the test case summary in batch mode reports PASS if all the tests within the test case passed, otherwise FAIL.

The audit tests use their own testing framework, where each test is executed twice: once with a positive test goal and once with a negative test goal. The audit tests that do not cover system calls directly but the supporting tools use a similar approach of iterating over the various stages as far as applicable. For each of the areas in the audit test suite, a driver program will perform global setup and run the individual test cases. Results are collected into the log file showing pass or fail verdicts. Additionally, the audit tests also cover the multi-level security (MLS) logic. By verifying that certain permutations of allowed and denied access requests are audited, the MLS logic is verified as well.

The manual tests cover functionality that cannot easily be tested in an automated way, such as serial terminals.

The test results of the sponsor can be found in Test Report. All the tests were executed successfully (pass/ok) apart from the test cases that are documented to fail or be skipped in the Test Plan. The test systems were configured according to the ST and the instructions in Evaluated Configuration Guide. The manual test results included in Test Report also include PASS/FAIL labeling by the sponsor.

The test results provided by the sponsor were generated on the following above mentioned systems.

7.1.3. Testing results

The test results provided by the sponsor were generated on the hardware platforms listed above. As described in the testing approach, the test results of all the automated tests are written to files. In addition a log-file for the LTP tests reports more details on the flow of the tests.

The test results of the few manual tests have been recorded by the sponsor and those results have been presented in separate files.

All test results from all tested platforms show that the expected test results are identical to the actual test results, considering the expected failures stated in the developer's test plan.

7.1.4. Test coverage

The functional specification has identified the following TSFI:

- System calls
- Security critical configuration files (TSF databases)
- Trusted programs and the corresponding network protocols of SSHv2, TLSv1 and SSLv3
- IPsec labeled network protocols

A mapping provided by the sponsor shows that the tests cover all individual TSFI identified for the TOE. An extension to this mapping developed by the evaluator as documented in the test case coverage analysis document shows that also significant details of the TSFI have been tested with the sponsor's test suite. This therefore satisfies the requirements for the evaluation, as an exhaustive interface specification testing is not required.

7.1.5. Test depth

In addition to the mapping to the functional specification, the sponsor provided a mapping of test cases to subsystems of the high level design and the internal interfaces described in the high level design. This mapping shows that all subsystems the internal interfaces are covered by test cases.

The depth analysis between the components of the HLD and the available test cases is seen as sufficient by the evaluator, because each test case can be matched to a subsystem of the HLD and vice versa, as shown in FSP mapping. All TSF covered by the subsystems of the high-level design are covered with test cases. The security-relevant internal interfaces can be linked to the test cases. As stated above, the hardware subsystem is always implicitly tested by every software component that runs on it. Therefore, the hardware subsystem was not explicitly mentioned in FSP mapping.

Not all of the internal interfaces mentioned in the high-level design could be covered by direct test cases. Some internal interfaces can – due to the restrictions of the evaluated configuration – only be invoked during system startup. This includes especially internal interfaces to load and unload kernel

modules, to register /deregister device drivers and install / deinstall interrupt handler. Since the evaluated configuration does not allow to dynamically load and unload device drivers as kernel modules those interfaces are only used during system startup and are therefore implicitly tested there.

7.2. Evaluator Testing

7.2.1. TOE test configuration

The evaluator independently installed the test systems according to the documentation in the Evaluated Configuration Guide and the test plan.

The sponsor provided the following hardware for testing:

- Dell D630 (using Intel Core 2 Duo processor)
- Intel 'Hanlan Creek' Dual Processor Xeon 5500 Series Pedestal Server Motherboard (S5520HCR) (using Intel Nehalem processor)

This hardware is located at the evaluator facility in Austin,Texas. The hardware configuration is identical to the system used by the sponsor to perform testing.

The evaluator installed Wind River Linux Secure 1.0 on these systems.

7.2.2. Subset size chosen

The evaluator chose to reproduce all automated test cases of the sponsor test suite, as well as the manual test cases for labeled networking.

7.2.3. Evaluator tests performed

In addition to reproducing all the automated sponsor tests, the evaluator devised tests for a subset of the TOE. The tests are listed in the Evaluator Test Plan.

The evaluator has chosen these tests for the following reasons:

- The test cases examine some of the security functions of the TOE in more detail than the sponsor supplied test cases. (Residual Information Protection, DAC and MLS override capability enforcement)
- The test cases cover aspects not included in the developer testing (verification of the ACL support in the archival tool assessment of the consistency of the MLS policy with Bell-LaPadula model, assessment of override capabilities)

As the sponsor-supplied test cases already cover the TOE in a broad sense the evaluator has devised only a small set of test cases.

The evaluator created several test cases for testing a few functional aspects where the sponsor test cases were not considered by the evaluator to be broad enough. During the evaluator coverage analysis of the test cases provided by the sponsor, the evaluator gained confidence in the sponsor testing effort and the depth of test coverage in the sponsor supplied test cases. The analysis has shown a very wide coverage of the TSF, therefore the evaluator devised only a small number of test cases.

7.2.4. Summary of Evaluator Test Results

The evaluator testing effort consists of two parts. The first one is the reproducing of the sponsor test execution and the second is the execution of the tests created by the evaluator.

The tests were performed at the evaluators' facility in Austin. The systems available for testing are listed above.

In each case the system was accessible through SSH and the system's console. The TOE operating system with the required additional packages as well as the test cases and test tools were installed on the test machine by the evaluator according to the instructions in Evaluated Configuration Guide and Test Plan. During the evaluation, the file system type was used for hard disk partitions on the test system. The configuration script ensured the evaluation compliant system configuration. After running the automated configuration, no further system configuration was performed and only the tools required for testing have been installed. The test system was therefore configured according to the ST and the instructions in the Evaluated Configuration Guide. The evaluator reproduced the execution of the test cases. The log files generated by the test cases were analyzed for completeness and failures. The sponsor provided automated test cases.

All the test results conformed to the expected test results from the test plan, including expected failures.

In addition to running the tests that were provided by the sponsor according to the test plan from the sponsor, the evaluator decided to run some additional test cases on the provided test systems as defined in Evaluator Test Plan:

- Permission settings of relevant configuration files
- Proper creation of password digests
- Verification the SUID programs do not change the real UID
- Testing of residual information protection in regular file system objects
- Check for data import / export with DAC enforcement

- Verification that the permission check during *open()* is enforced during *read()* and *write()*
- Verification of cleaning of environment for SUID/SGID binaries
- Test that the MLS policy complies with Bell-LaPadula
- Test for MLS override attributes
- Test for trusted objects
- Test for ranged objects
- Test for 32-bit system calls when 32-bit runtime is not available
- Test for poly-instantiated directories

All tests passed successfully.

8. DOCUMENTATION

8.1. Product Guidance

The guidance documentation examined during the course of the evaluation and delivered with the TOE is as follows (documents shown with ‡ are not security relevant):

- *Wind River EAL4 Evaluated Configuration Guide for WindRiver Linux Secure 1.0*, April 5, 2011; v1.7
- *Wind River Linux Secure: Administrator's Guide 1.0*, August 20, 2010, Linux 3 Version
- *Wind River Linux Secure: Configuration Guide 1.0*, August 20, 2010, Linux 3 Version
- *Wind River Product Installation and Licensing: Administrator's Guide 2.2*, December 11, 2009‡
- *Wind River Product Installation and Licensing: Developer's Guide 2.2*, December 11, 2009‡
- *Host requirements for Wind River Linux Secure 1.0 - Installation Instructions*, undated
- *PAM Admin Guide*, 2010-07-26
- *Wind River Workbench By Example, 3.1.* February 18, 2009. Linux 3 Version‡
- *Wind River Linux Secure 1.0 Security Target*, v 1.17, 2011-04-05
- *Evaluation Technical Report for a Target of Evaluation Wind River Secure Linux 1.0*, ETR Version 4.0 as of 2011-04-05
- Linux manual pages describing usage of all interfaces

Note: Although the *Wind River Linux Secure, Protection Profiles Policy Guide version 1.0* ships with the product, it is an obsolete document and must be ignored during the installation of the TOE

in the evaluated configuration. End users are informed of this with a slip of paper enclosed with the CD, as it was not possible to adjust the timing of CD manufacturing to remove the document.

8.2. Evaluation Evidence

The following tables identify the additional documentation submitted as evaluation evidence by the vendor. With the exception of the Security Target, these documents may be proprietary and not available to the general public.

Design Documentation	Version	Date
Windriver Linux Secure 1.0 EAL4 High-Level Design	-	2010-11-29
Cortex™-A8 (Revision r3p2) Technical Reference Manual	J	2009-05-15
Cortex™-A8 Technical Reference Manual – ARM DDI 0344H Errata 01	-	-
ARM® Architecture Reference Manual ARM®v7-A and ARM®v7-R edition Errata markup	B	2010-07
Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 1: Basic Architecture, Order №: 253665-034US	-	2010-03
Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 2A: Instruction Set Reference, A-M, Order №: 253666-034US	-	2010-03
Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 2B: Instruction Set Reference, N-Z, Order №: 253667-034US	-	2010-03
Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3A: System Programming Guide, Part 1, Order №: 253668-034US	-	2010-03
Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3B: System Programming Guide, Part 2, Order №: 253669-034US	-	2010-03
Intel® 64 and IA-32 Architectures Software Developer's Manual Documentation Changes, Document №: 252046-027	-	2010-03
Intel® 64 and IA-32 Architectures Optimization Reference Manual, Order №: 248966-020	-	2009-11
MPC8572E PowerQUICC™ III Integrated Host Processor Family Reference Manual, Document № MPC8572ERM	2	2008-05
Power ISA™:	2.06	2009-01-30
OMAP3530/25 Applications Processor, Part № SPRS507F	-	2009-10
Stroustrup, Bjarne. *The C++ Programming Language: Special Edition*. ISBN 978-0201700732.	3rd	2000-02-11
Kernighan, Brian W.; Ritchie , Dennis M. *The C Programming Language*, ISBN 978-0131103627	2nd	1988-04-01
Mayer , Frank; MacMillan, Karl; Caplan, David. "*SELinux by Example:*	1st	2006-08-06

Using Security Enhanced Linux"		
@tsec Information Security/Wolfgang Mauerer. "*Linux Kernel Architecture"*	-	2010-10-12
@tsec Information Security, "*Linux Userspace Architecture*"	-	2010-09-21
NSS Design Specification	-	2010-06-01
User Manual Pages	-	2010-12-02
FSP Mapping - Wind River Linux Secure 1.0	-	2010-11-30
RFC 2409 - The Internet Key Exchange (IKE)	-	1998-11
RFC 3268 - Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)	-	2002-06
RFC 4253 - The Secure Shell (SSH) Transport Layer Protocol	-	2006-01
The SSL Protocol - Version 3.0	3.02	1996-11-18
Faigin, D. P.; Donndelinger, J. J.; Jones, J. R. "*A Rigorous Approach to Determining Objects*". In *Proc. 8th Annual Computer Security Applications Conference*	-	1993-12

Configuration Management Documentation	**Version**	**Date**
WRLinux Code Check-in Process	-	2010-03-13
Sample Kernel checkin log messages	-	2010-08-02
GIT CM List	-	2011-01-07
Git Pull Workflow	r1	2010-02-24
GIT Branches	-	2010-07-29
Using GIT	-	2010-06-10
Xylo documentation	-	2010-08-05
LPD-specific FAST Documentation	-	2010-03-01
Open Source Contributions	r80	2010-04-14
Peer Review Policy	-	2010-07-19
Static Analysis Policy	-	2010-07-19
Basic Peer Review Process	-	2010-07-19
Inspection Peer Review Process	-	2010-07-19
Prerequisites to using git	-	2010-05-05

Delivery and Operation Documentation	Version	Date
Definitions of Box Opening Phases	-	2010-07-22
CD Image Distribution (DMZ and Akamai (Speedera))	-	2010-07-19
Release Operations	-	2010-07-22
Scanning release CD images	-	2010-07-22
Wind River Linux 3.0 Daily and Weekly spins	-	-
Handoff to manufacturing	-	2010-08-12

Lifecycle Documentation	Version	Date
Wind River Desktop Security Policy	-	2010-07-15
Wind River System Electronic Data Systems Policy	-	2010-07-15
Wind River Systems Guidelines for Fileserver Usage	-	2005-03
Protecting Company Info	-	2010-07-15
Wind River System IT Security Policy	-	2010-07-15
Lab Management Best Practices	-	2010-08-25
Wind River Policy & Procedures: Password Change	-	2010-07-15
Development Engineering Quality System Portal	-	2010-07-15
Wind River Employment Checks	-	2010-07-15
Wind River Employment Handbook	-	2008-01-09
Customer Support User's Guide	10	-
Process to handle Security and CERT notifications	-	2010-07-28
Wind River - Security Vulnerability Response Policy	3.0	-
Directory of Known keywords used by Linux in Clearquest	-	2010-07-28
Service Request Resolution Process	-	2010-07-28
Defect Management: Enterprise and Project Defect Process and Tool	-	2008
Engineering Product Release Life Cycle (PRLC) Model	-	2010-07-19
Making a Product Orderable, Oracle Links, and Release Testing Template	-	2010-07-22
Key Concepts Used in the Wind River Process Library	-	2010-07-15
Using *as* - The GNU Assembler	-	1994-01
GNU make	-	2002-07-08

Bailey, Edward C. *Maximum RPM: Taking the Red Hat Package Manager to the Limit*	-	1997-02-17
Dashboard for Internal Engineering Tools and Applications	-	2010-08-25
DefectTracker Introduction	-	2010-07-28
Policy for Handling ITAR Software and Documentation	3	2008-08-12
Service Request - Conversion of Employee/Contractor	-	2010-08-18
Service Request - New Employee/Contractor	-	2010-08-18
Service Request Resolution Process	-	2010-08-03
Service Request - Terminate Employee/Contractor HR Service	-	2010-08-18
VPN SecureID Request Form	-	2010-08-18

Test Documentation	**Version**	**Date**
Evaluator test plan for Wind River Linux Secure version 1.0 EAL4 Evaluation	-	2011-03-30
Evaluator test results for subset of developer tests	-	2010-11-11
Test cases for Wind River Linux Secure version 1.0 EAL4 Evaluation	-	2010-11-11
Expected Failures	-	2011-02-25
WRLS Test Instructions	-	2011-02-18
Test results for Wind River Linux Secure version 1.0 EAL4 Evaluation	-	2011-02-18

Security Target	**Version**	**Date**
Wind River Linux Secure 1.0 Security Target	1.17	2011-04-05

9. RESULTS OF THE EVALUATION[1]

The evaluation team determined the product to be CC Part 2 extended, CC Part 3 conformant, GP-OSPP conformant, and to meet the requirements of EAL 4 augmented by ALC_FLR.3. In short, the product satisfies the security technical requirements specified in Wind River Linux Secure Version 1 Security Target on platforms listed in Table 1: Evaluation Identifiers.

[1] The terminology in this section is defined in CC Interpretation 008, specifying new language for CC Part 1, section/Clause 5.4.

10. VALIDATOR COMMENTS

10.1. Covert Channel Analysis

The TOE claims and implements information flow control policies in the evaluated configuration. However, as identifying covert channels is well beyond the expected robustness of a product evaluated at EAL4 with Enhanced-Basic attack potential resistance, the evaluation team did not perform a formal covert channel analysis of the product. Any consumer of the TOE should be aware of this fact and take this into consideration in all risk-assessments involving this product and the user data it is meant to protect.

10.2. UNIX STIG Analysis

The CCTL used the UNIX STIG standard as input to the vulnerability analysis. After the vulnerability analysis was complete, the CCTL concluded that no file permissions needed to be changed to prevent the vulnerabilities covered by the UNIX STIG.

10.3. Conformance with External Standards

The product has not been evaluated to be conformant with any standard external to the Common Criteria, except those that have been required by the claimed protection profile "US Government Protection Profile for General-Purpose Operating Systems in a Networked Environment". The accreditory or acquisition agency is advised to consider this fact in any risk assessments done involving this product. Specifically, the product has not been evaluated against:

- Unix Security Technical Implementation Guide version 5, release 1
- Department of Defense Instruction Number 8500.2
- NIST Special Publication 800-53 Revision 3

11. SECURITY TARGET

The ST, *Wind River Linux Secure 1.0 Security Target v 1.15* is included here by reference.

12. LIST OF ACRONYMS

ACL	Access Control List
AES	Advanced Encryption Standard
ARM	Advanced RISC Machine

ASCII	American Standard Code for Information Interchange
BSD	Berkeley Software Distribution
CBC	Cipher Block Chaining
CC	Common Criteria
CCEVS	Common Criteria Evaluation and Validation Scheme
CCTL	Common Evaluation Testing Laboratory
CD	Compact Disk
CEM	Common Evaluation Methodology
CNSS	Committee on National Security Systems
Configfs	A RAM-based virtual file system provided by the 2.6 Linux kernel. Not evaluated.
DAC	Discretionary Access Control
EAL	Evaluation Assurance Level
ECB	Electronic Code Book
ECG	Evaluated Configuration Guide
eCryptFS	A POSIX-compliant enterprise-class stacked cryptographic filesystem for Linux, derived from Erez Zadok's Cryptfs. The cryptography of this filesystem is not covered by the evaluation.
ETR	Evaluation Technical Report
ext2	Second extended filesystem. A file system for the Linux kernel. Not evaluated.
Ext3fs	Third extended filesystem. A journaled file system that is commonly used by the Linux kernel.
FIPS	Federal Information Processing Standard
FSP	Functional Security Policy
GIT	Distributed revision control system with an emphasis on speed.
GPOSPP	U.S. Government Protection Profile for General-Purpose Operating Systems in a Networked Environment
HLD	High Level Design

I&A	Identification and Authentication
ID	Identifier
IKE	Internet Key Exchange
IPC	Inter-process Communication
IPSec	Internet Protocol Security
IT	Information Technology
ITAR	International Traffic in Arms Regulations
LTP	Linux Testing Project
MAC	Mandatory Access Control
MLS	Multi-Level Security
NFS	Network File System
NIAP	National Information Assurance Partnership
NIST	National Institute of Standards & Technology
NSA	National Security Agency
NSS	Network Security Services
NVLAP	National Voluntary Laboratory Assessment Program
POSIX	Portable Operating System Interface [for Unix]
PP	Protection Profile
PPC	PowerPC
PRLC	Product Release Life Cycle
RAM	Random-access memory
RFC	Request for Comments
RPM	Red Hat Package Manager
SELinux	Security-Enhanced Linux
SGID	Set Group ID

SHA	Secure Hash Algorithm
SMP	Symmetric Multiprocessing
SP	Special Publication
SSH	Secure Shell
SSL	Secure Sockets Layer
ST	Security Target
STIG	Security Technical Implementation Guide
SUID	Set User ID
TC	Test Cases
TLS	Transport Layer Security
Tmpfs	A common name for a temporary file storage facility on many Unix-like operating systems. It is intended to appear as a mounted file system, but stored in volatile memory instead of a persistent storage device.
TOE	Target of Evaluation
TP	Test Plan
TR	Test Report
TSF	TOE Security Function
TSFI	TOE Security Function Interface
UID	User ID
Unix	Computer operating system originally developed in 1969 by a group of AT&T employees at Bell Labs, including Ken Thompson, Dennis Ritchie, Brian Kernighan, Douglas McIlroy, and Joe Ossanna.
UP	Uniprocessor
WCF	Wind River Cryptographic Framework
WRLinux	Wind River Linux

13. BIBLIOGRAPHY

[1] Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and general model, Version 3.1.

[2] Common Criteria for Information Technology Security Evaluation – Part 2: Security functional requirements, Version 3.1.

[3] Common Criteria for Information Technology Security Evaluation – Part 3: Security assurance requirements, Version 3.1.

[4] Common Evaluation Methodology for Information Technology Security – Part 1: Introduction and general model, Version 3.1.

[5] Common Evaluation Methodology for Information Technology Security – Part 2: Evaluation Methodology, Version 3.1.

[6] Wind River Linux Secure 1.0 Security Target v 1.15

[7] Evaluation Technical Report for a Target of Evaluation Wind River Secure Linux 1.0 ETR Version 3.0 as of 2011-03-31

[8] Wind River EAL4 Evaluated Configuration Guide for WindRiver Linux Secure 1.0, March 28, 2011; v1.6

[9] Wind River Linux Secure, Administrator's Guide version 1.0

[10] Wind River Linux Secure, Configuration Guide version 1.0

[11] FIPS Pub 140-2