Host-Oriented Security Test Suite (HOSTS)

Version 1.5.0.0

Standard Operating Procedures Under SunOS 5.8 and HP-UX 11.0

February 2002

The MITRE Corporation
7515 Colshire Drive
McLean, VA 22102-7508

LICENSE

Host-Oriented Security Test Suite (HOSTS)
Standard Operating Procedures (SOP)

Executive Summary

The Host-Oriented Security Test Suite (HOSTS) is an interactive utility that automates many aspects of security testing performed within the Common Operating Environment (COE). The utility is both flexible and easily customizable, requiring only the Perl programming language and common Perl modules. By using HOSTS as part of the security evaluation process, a level of consistency and repeatability in testing can be readily achieved. The additional benefit of a reduced probability for an operator-induced error, which can skew tests results, is also achieved. This is a direct consequence of the reduction for both the amount of time and level of effort required to perform COE security testing.

This document provides the Standard Operating Procedures (SOP) for using HOSTS under Sun Microsystems' SunOS 5.8 and Hewlett-Packard's HP-UX 11.0. It contains sufficient detail to enable a junior to mid-level Security Engineer to install, configure, execute, and maintain HOSTS for evaluating a system's security profile. Any extensive modifications will require both Bourne shell script and Perl programming experience.

Introduction

Purpose and Scope

This document provides the Standard Operating Procedures (SOP) for a utility, known as the Host-Oriented Security Test Suite (HOSTS) Version 1.4.0.2, which can be used to create and maintain a security profile of a target system's operating environment. While this utility has been designed for use in evaluating implementation compliance with the *Defense Information Infrastructure (DII) Common Operating Environment (COE) Security Requirements Specification (SRS)*, it can be easily adapted for compliance evaluation against other security specifications. The HOSTS[1] utility has been written using Perl 5.005 and the Bourne shell, and it has been used under later versions of Perl. This SOP applies to HOSTS using the Sun Microsystems' SunOS 5.8 and Hewlett-Packard's HP-UX 11.0.

HOSTS can be used for both security baseline evaluation and operational compliance with the baseline. The utility is based on the premise that, in general, the majority of existing security vulnerabilities are the result of failing to configure a system properly. Consequently, the ability to test for the presence of these "improper configuration" vulnerabilities can be done in a logical and repeatable manner. Through the periodic execution of HOSTS, the System Administrator can monitor both how well a given system complies with applicable security requirements and how a given system's configuration may change over time.

The HOSTS utility provides the System Administrator with the capability of enhancing a system's security through the tightening of that system's configuration. It also provides a measure of reassurance that security-related tasks function in accordance with expectations (e.g., non-privileged users cannot change someone else's password).

This version of HOSTS is in no way meant to be an all-inclusive test product. For example, this release does not examine configurable security-related items not currently evaluated within the *COE Kernel Platform Compliance (KPC) Security Test Procedures*[2]. Nor does this release examine binary executables for known exploitable weaknesses within the various library calls. Finally, HOSTS will not remove any of the configuration vulnerabilities it does identify.

The current HOSTS distribution includes the main driver program, several common task plugin modules, and test input files based on the five COE KPC security test procedures. (The common task plugin modules are subprograms that use a common approach for performing a given test.

---

[1] HOSTS was developed by James Finegan of The MITRE Corporation as part of a Defense Information Systems Agency (DISA)-sponsored COE KPC project.

[2] Future releases of HOSTS will address many of these items. In final form, HOSTS will track closely with the *Consolidated Unix Security Practices (CUSP)* document currently under development by MITRE.

For example, plugins exist for evaluating file and directory attributes such as protections and ownership).

HOSTS currently provides the following capabilities:

1. Define pass/fail criteria/conditions.
2. Tag a given test step to a specific security requirement.
3. Perform security testing with minimal operational intrusion or disruption.
4. Add, modify, and remove test steps as requirements, operational environments, system services, and configurable options change.
5. Add new common task subprograms through the creation of new plugin modules.
6. Track anomalous files[3] over time.

Approach

HOSTS can be made available to a system through either of the following formats:

1. HOSTS can be executed directly from a mounted partition (e.g., directly from a compact disk [CD] or via a Network File Service [NFS]-mounted remote partition).
2. HOSTS can be installed locally using the Unix `tar` command. The `tar` format was chosen to facilitate portability between Unix variants. The tar file itself will fit on a single floppy diskette.

The following steps define the installation, execution, and evaluation process:

1. Specify pre-installation requirements and considerations.
2. Install HOSTS.
3. Execute HOSTS.
4. Evaluate the results.
5. Customize the input test files.
6. Define available plugin modules.

The main sections of this document are structured according to these steps. The appendices provide supplementary material.

[3] Lists of files matching a definable anomaly pattern are created when HOSTS examines the local system partitions. Examples include files with non-standard file names (e.g., "..."), Set User Identifier (SetUID) and Set Group Identifier (SetGID) shell scripts, world-write-enabled files, and files not owned by a local system user.

Assumptions

This document provides the Standard Operating Procedures for using HOSTS. It contains sufficient detail to enable a junior to mid-level Security Engineer to install, configure, execute, and maintain HOSTS for evaluating a system's security profile. Any extensive modifications will require both Bourne shell script and Perl programming experience.

While the information detailed within this SOP applies to HOSTS as installed under Sun Microsystems' SunOS 5.8 and Hewlett-Packard's HP-UX 11.0 operating systems, the basic functionality will port to other Unix variants. Modifications to the input test files and, potentially, to the plugin modules will be required. The HOSTS utility has been shown to run equally well under SunOS 5.8 on both SPARC and X86 hardware architectures.

HOSTS *must* be installed and run from a privileged account (e.g., `root`).

Conventions Used

User entries are **boldfaced**. *Italics* are used for emphasis and pre-defined values.

Part 1
Specify Pre-Installation Requirements and Considerations

This section provides pre-installation requirements and considerations that must be addressed prior to the installation and use of HOSTS. Sufficient disk space resources must be available to hold the utility, the created test component (e.g., accounts, audit data files), and the result files.

Table 1 provides the minimum hardware and software requirements for HOSTS.

Table 1. Minimum Hardware and Software for HOSTS

Hardware	Software
Minimum Recommended Disk Space: ?? 2 MB for the utility ?? 50 MB for audit files ?? 5 MB for scan results files	?? SunOS (5.5.1 or above) and HP-UX 11.0, including the Bourne shell. The test series included with this distribution have been optimized for SunOS 5.8 and HP-UX 11.0. ?? Perl 5.005 with the following modules and functions: ?? File::Find – Traverses a file tree ?? File::Basename – File specification parser ?? ctime.pl – Perl-based time conversion function

Hardware

Since HOSTS will be generating both audit logs and test files, sufficient space needs to be available for use while the audit function is being tested. Additional space is required for storage of the scan result files.

Disk Space. Observation has shown that under normal operational conditions, a busy system will consume less than 25 megabytes (MB) of new disk space for its audit data files over the duration of these tests. Following test conclusion, the residual audit log files will need to be manually purged following optional archiving.

Under the as-distributed Solaris configuration, audit data files are collected in the `/var/audit` directory. Under a typical COE installation, the audit data collection point changes to `/security1` and, if available, `/security2`. By default, the `/security1` partition is also

used by HOSTS to house its collected audit data files. Consequently, `/security1` must be large enough to hold both the original audit data files and all those created by the test utility. Some audit tests may fail if there is less space than the recommended amount available under `/security1` and no additional space is available elsewhere on the system.

By default, HOSTS will install in the `/opt` partition. As distributed, its installation size is less than 2 MB.

Depending on the number of anomalous files detected during the file scans performed by the utility, a percentage of space available within `/tmp` will be consumed. To be safe, at least 25 MB should be available under `/tmp`. (Note: These residual files are not automatically removed upon completion of the testing.)

Software

HOSTS was written entirely with Perl 5.005 and Bourne shell commands. While the standard Solaris operating environment is sufficient to run the Bourne shell component, the installation of Perl will be required for SunOS releases prior to SunOS 5.8 (Solaris 8) as well as HP-UX 11.0. HOSTS does expect to find the Perl binary at `/usr/bin/perl`.

How HOSTS Works

HOSTS consists of three components that provide a method for performing a sequence of tests. This sequence is repeatable (e.g., the sequence is re-executed each time a test is run). The following list defines the three HOSTS components:

Driver Engine	This Perl-based driver reads the input test files, processes the test records within the specified input file, and declares each test performed as either a pass or a fail. Counts are kept recording the number of passes and failures.
Input Test File	The input test file is a collection of one-line test records processed in a sequential manner. Each record includes identifying information (e.g., a test number and description), the test to be performed, and the criteria required to declare a pass. The test may be a parameter-based call to the one of the test plugins or it may contain an actual environment-specific command that will be passed to the OS for execution.
Test Plugins	A test plugin is a modular block of code that performs a single function. The plugin serves as the actual interface between the test

driver and the OS being evaluated. The plugin converts the test to the native OS for processing.

The initial objective was to provide a tool that could be easily ported from one operating environment to another with minimal modification. While the initial release of this tool includes several example input test files that contain numerous environment-specific command test records, the plugin library should eventually expand to allow for most of these to be replaced with a standardized plugin call. Approximately 90+ percent of the tests that need to be performed can be handled via a plugin call rather than through an environment-specific command.

Each test record within an input test file is processed as follows:

1. The test record is read from the input test file. Each record is split into its constituent parts.
2. The test is performed by executing the test plugin extracted from the test record.
3. The plugin returns results from the test.
4. The returned results are compared to the expected value(s). If the returned results match the expected results, a match is declared and the test is flagged as passed.

Figure 1 provides a graphical representation of the HOSTS process flow.

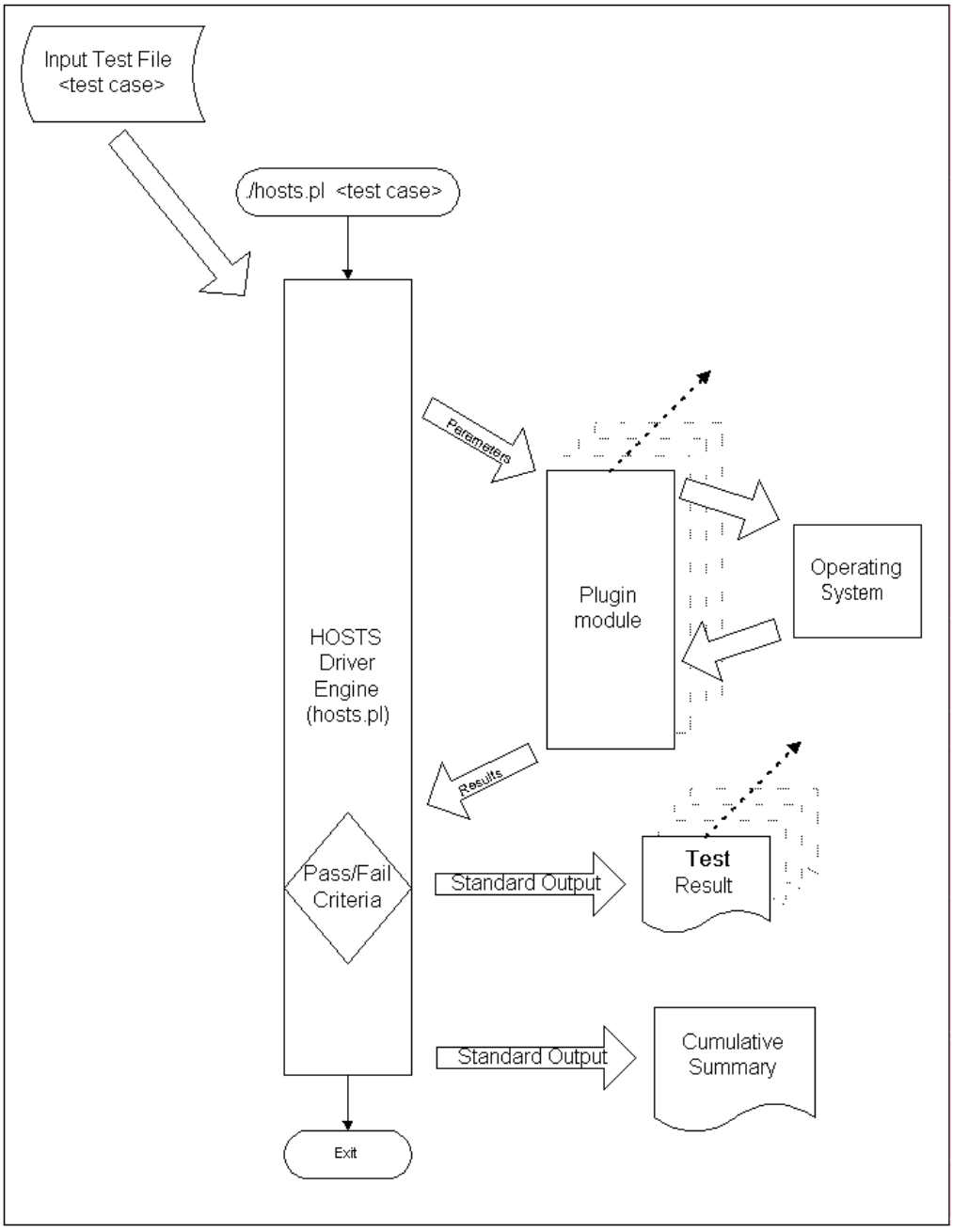

Figure 1. HOSTS Process Flow

Part 2
Install HOSTS

This section provides instructions for installing HOSTS.

The steps required are summarized as follows:

1. Access the software repository.
2. Install HOSTS on a local file system.

Access the Software Repository

Neither the mounting of removable media nor the mounting of a remote partition from a file server is covered in this document.

Install HOSTS on a Local File System

Local HOSTS installation requires access to the `Hosts.tar` tarball. Once the tarball has been obtained, the installation consists of the following steps:

1. Create the target base directory.
2. Extract HOSTS into the target base directory.

Below is the detailed recommended installation procedure for HOSTS under Solaris 8. (See Appendix A for a sample listing from an actual HOSTS installation.)

1. Become `root` on the local system. Typically, this is accomplished by entering the following at the prompt:

```
prompt -->: /usr/bin/su -
```

2. Ensure the medium containing HOSTS is available and mounted. Change the default directory to the directory containing HOSTS.

3. Ensure that Perl is accessible via `/usr/bin/perl`:

```
prompt -->: /bin/ls -l /usr/bin/perl
```

The system should display a listing similar to the following:

```
prompt -->: /bin/ls -l /usr/bin/perl
-r-xr-xr-x   3 root     bin       14312 Jan  8  2000 /usr/bin/perl
```

If `/usr/bin/perl` is not defined, the system should display a listing similar to the following:

```
prompt -->: /bin/ls -l /usr/bin/perl
/usr/bin/perl: No such file or directory
```

4. If `/usr/bin/perl` is not defined, a soft link must be created so that Perl may be accessed as `/usr/bin/perl`[4]:

   ```
   prompt -->: /bin/ln -s <perl path> /usr/bin/perl
   ```

   where: `<perl path>` is the fully qualified path to the location where Perl is installed.

   For example, if Perl 5.005 is installed as `/usr/local/bin/perl`, the command becomes:

   ```
   prompt -->: /bin/ln -s /usr/local/bin/perl /usr/bin/perl
   ```

5. Create the HOSTS target directory:

   ```
   prompt -->: /bin/mkdir -p <HOSTS target dir>
   ```

   where: `<HOSTS target dir>` is the HOSTS target directory name

   For example, if the target directory is `/opt/hosts`, the command becomes:

   ```
   prompt -->: /bin/mkdir -p /opt/hosts
   ```

6. Set the default working directory to the HOSTS target directory:

   ```
   prompt -->: cd <HOSTS target dir>
   ```

   where: `<HOSTS target dir>` is the HOSTS target directory name

   For example, if the target directory is `/opt/hosts`, the command becomes:

   ```
   prompt -->: cd /opt/hosts
   ```

7. Using the `tar` command, add the HOSTS package:

[4] This is a hard-coded path. This was done to ensure that the Perl executed is the system version of Perl, not a version of Perl from an untrusted source.

```
prompt -->: /bin/tar -xvf <Hosts.tar>
```

where: `<Hosts.tar>` is the fully qualified HOSTS distribution tar file name

For example, if the tarball is on a CD that is mounted as `/cdrom`, the CD mount point is `hostsv1.5`, and the tarball is named `Hosts.tar`, the command becomes:

```
prompt -->: /bin/tar -xvf /cdrom/hostsv1.5/Hosts.tar
```

8. A series of messages will now be displayed:

```
x ., 0 bytes, 0 tape blocks
x hosts.pl, 36248 bytes, 71 tape blocks
x sample_runs, 0 bytes, 0 tape blocks
x sample_runs/hpux_system, 0 bytes, 0 tape blocks
x sample_runs/hpux_system/dac_series_log.txt, 23769 bytes, 47 tape blocks
x sample_runs/hpux_system/os_series_log.txt, 13593 bytes, 27 tape blocks
x sample_runs/hpux_system/misc_series_log.txt, 3917 bytes, 8 tape blocks
x sample_runs/hpux_system/audit_series_log.txt, 5188 bytes, 11 tape blocks
x sample_runs/hpux_system/ia_series_log.txt, 5325 bytes, 11 tape blocks
x sample_runs/hpux_system/combined_series_log.txt, 58444 bytes, 115 tape blocks
x sample_runs/solaris_system, 0 bytes, 0 tape blocks
x sample_runs/solaris_system/ia_series_log.txt, 5282 bytes, 11 tape blocks
x sample_runs/solaris_system/combined_series_log.txt, 38918 bytes, 77 tape blocks
x sample_runs/solaris_system/misc_series_log.txt, 3875 bytes, 8 tape blocks
x sample_runs/solaris_system/os_series_log.txt, 11437 bytes, 23 tape blocks
x sample_runs/solaris_system/dac_series_log.txt, 7252 bytes, 15 tape blocks
x sample_runs/solaris_system/audit_series_log.txt, 4487 bytes, 9 tape blocks
x sample_runs/solaris_system/srs_series_log.txt, 151119 bytes, 296 tape blocks
           .
           .   All the files installed will be listed.
           .
x hpux/baseline, 0 bytes, 0 tape blocks
x hpux/baseline/8_Recommended, 650 bytes, 2 tape blocks
x hpux/baseline/known-c-shells, 2440 bytes, 5 tape blocks
x hpux/baseline/known-non-binary-setuid, 11 bytes, 1 tape blocks
x hpux/baseline/known-setuid-setgid-files, 14559 bytes, 29 tape blocks
x hpux/baseline/known-strange-name-files, 2583 bytes, 6 tape blocks
x hpux/baseline/known-non-sb-ww-directories, 827 bytes, 2 tape blocks
x COPYING, 26430 bytes, 52 tape blocks
LICENSE linked to COPYING
x doc, 0 bytes, 0 tape blocks
x doc/test_cross_ref.pdf, 16535 bytes, 33 tape blocks
x doc/hosts_sop_v14.pdf, 213117 bytes, 417 tape blocks
x 00_README_XEQ.txt, 5463 bytes, 11 tape blocks
x 00_README.txt, 909 bytes, 2 tape blocks
x bin, 0 bytes, 0 tape blocks
bin/hosts.pl linked to hosts.pl
```

9. As a conformation of a successful installation, list the `/opt/hosts` directory:

```
prompt -->: /bin/ls –la /opt/hosts
total 208
drwxr-xr-x   7 root     staff        512 Feb 11 09:45 ./
drwxr-xr-x  73 root     staff       2560 Feb  5 11:49 ../
-rw-r--r--   1 root     staff        909 Feb  6 12:33 00_README.txt
-rw-r--r--   1 root     staff       5463 Sep 28 15:22 00_README_XEQ.txt
-rw-r--r--   2 root     staff      26430 Aug 30 14:25 COPYING
```

```
-rw-r--r--   2 root     staff      26430 Aug 30 14:25 LICENSE
drwxr-xr-x   2 root     staff        512 Feb  4 14:05 bin/
drwxr-xr-x   2 root     staff        512 Jan 14 14:15 doc/
-rwxr-x---   2 root     staff      36248 Feb  4 16:19 hosts.pl*
drwxr-xr-x   4 root     staff        512 Feb  6 12:40 hpux/
drwxr-xr-x   6 root     staff        512 Feb  5 12:13 sample_runs/
drwxr-xr-x   4 root     staff        512 Feb  4 16:16 solaris/
```

10. If applicable, dismount the removable medium or remote file system from which HOSTS was loaded.

11. Log off the system.

Part 3
Execute HOSTS

This section provides instructions for executing HOSTS. It assumes that HOSTS has been installed locally and mounted for execution either from a CD-ROM or as an NFS partition from a remote server.

Qualifiers and Control Variables

HOSTS provides one qualifier and one global variable for controlling program execution.

Qualifiers. Qualifiers are used to control HOSTS behavior as the utility executes. Table 2 lists the command line qualifiers available for HOSTS.

Table 2. HOSTS Command Line Qualifiers

Qualifier	Description
`-verbose`	When this qualifier is entered on the command line, HOSTS will display the results of all tests (both passed and failed). When it is omitted, HOSTS will only display test results if the test fails.
`-autogen`	When this qualifier is entered on the command line, HOSTS will automatically update the SKIPTEST definition within the executed test series. All tests within the series will be executed. Upon completion, SKIPTEST will be updated to include all failed tests. A summary will be displayed after each series is executed listing the changes made to SKIPTEST. Both newly passed tests and newly failed tests will be highlighted in a table as a reminder that tests must not be blindly skipped simply because they are failing.[5] The SKIPTEST definition must already exist within the test series file before this qualifier can be used. A null entry (e.g., SKIPTEST=;) will suffice.

[5] It is important that all failed tests be examined to insure the reason the test is failing is not associated with an error in the test. If the failure is related to an error, it should be corrected.

Global Variables. Global variables are used to control where HOSTS will expect to find subcomponents such as the input test files. They can also be used to control the behavior of HOSTS. Table 3 lists the global variables available for HOSTS.

Table 3. HOSTS Global Variables

Variable	Description
`HOSTS_TEST_PATH`	This global variable can be set to the fully qualified path where HOSTS will find the input tests files. By default, this variable is not set. In this case, HOSTS expects the input test files to be located in `<host_base_directory>/<os>` Where `<hosts_base_directory>` is the directory in which the `hosts.pl` executable is located and `<os>` is the operating system of the candidate under test. This release of HOSTS is configured to support the value "sunos" for Solaris 8 and "hpux" for HP-UX 11.0.

Running HOSTS

The steps required to run HOSTS are summarized as follows:

1. Run HOSTS for each of the desired tests.
2. Review the results.
3. Correct identified configuration problems as applicable.
4. Repeat steps 1 through 3.

See Appendix B for sample listings from an actual run. The steps are listed below:

1. Log into the system as `root`. (All tests MUST be executed as root.)

2. Ensure that Perl is accessible via `/usr/bin/perl`:

   ```
   prompt -->: /bin/ls -l /usr/bin/perl
   ```

   The system should display a listing similar to the following:

   ```
   prompt -->: /bin/ls -l /usr/bin/perl
   -r-xr-xr-x   3 root     bin       14312 Jan  8  2000 /usr/bin/perl
   ```

   If `/usr/bin/perl` is not defined, the system should display a listing similar to the following:

   ```
   prompt -->: /bin/ls -l /usr/bin/perl
   /usr/bin/perl: No such file or directory
   ```

3. Capture a log of the interactive session for later review:

   ```
   prompt--> /bin/script <session log>
   ```

   where `<session log>` is the name of the file in which the session log will be captured.

   For example, to capture the session log as `/var/tmp/hosts_test_1`, enter the following:

   ```
   prompt--> /bin/script /var/tmp/hosts_test_1
   ```

4. Change the default directory to the directory containing HOSTS:

   ```
   prompt--> cd <HOSTS directory>
   ```

   where `<HOSTS directory>` is the name of the first test file to be executed.

For example, if HOSTS is installed locally in `/opt/hosts`, enter the following:

```
prompt--> cd /opt/hosts
```

5. Execute the HOSTS main program:

```
prompt--> ./hosts.pl [<verbose>] <test 1> [<test 2> ... <test n> ]
```

where	**<verbose>**	is the optional verbose mode qualifier (e.g., `-verbose`).
	<test 1>	is the name of the first test file to be executed[6].
	<test 2>	is the name of the second test file to be executed.
	<test n>	is the name of the n^{th} input test file to be executed.

Multiple tests may be executed through a single activation of HOSTS. To do this, simply list the names of the input test files to be executed on the command line. Separate the input test file names with a space. A global summarization of the results will be displayed after all tests have been run.

For example, to execute a test in which the input test file is named "sendmail", enter:

```
prompt--> ./hosts.pl sendmail
```

To execute three tests sequentially under verbose mode where the input test files have been named "sendmail", "audit", and "account", enter:

```
prompt--> ./hosts.pl -verbose sendmail audit account
```

6. Respond to any manual input required for the test sequence. For example, if the test attempts to set a password via the `passwd` command, manually entered input will be required[7].

---

[6] If the test file to be executed is not located in the tests subdirectory beneath the installation directory for the `hosts.pl` executable, then the entry must be a fully qualified path.

[7] The SVR4 `passwd` function, as a security precaution, will only accept interactive (e.g., from the keyboard) input for a new password. Consequently, any test using `passwd` to set a password on a user account will require manual password entry.

7. Header information will be displayed as each section is processed. In addition, summary information will be displayed as the test process completes one test series and begins another. The following provides an example of the beginning headers.

```
                    The MITRE Corporation
              Center for Integrated Intelligence Systems
                    Network Security Engineering

               Host Oriented Security Test Suite (HOSTS)
             Version 1.5.0.0    Released 04 February, 2002


Generated on:                  Output Mode:                  Generated by:
testhost                     Failures Only                           root

Test Input File:                                            Test Run Time:
./solaris/paranoid_series                        Mon Feb  4 16:24:24 EST 2002

================================================================================
================================================================================

Now Processing: …
```

The output mode in the above example is given as “Failures Only”. This means that only failed tests will be displayed. Had HOSTS been activated with the verbose qualifier, the term “Verbose” would have been displayed instead.

8. As each section header is encountered within the test input file, a section message will be displayed.

   For example, the section header messages may appear similar to the following:

```
Now Processing: Test Setup
Now Processing: WWW-XXX-1: Verify some very interesting facet of system
                operation though inspection of files and subdirectories.
```

   In addition to the section header, comments be displayed. Comments appear similar to the following comment calling for operator action:

```
----
OPERATOR ACTION:
----
If some sort of manual input is needed, then please do it
here.  Press return when done.  There may be times that
nothing will be displayed on the screen.
```

9. By default, unless the verbose qualifier was entered on the command line, only failed tests will be displayed.

The results of each test will be summarized in output similar to the following:

```
       Test SomeTest.101        Test Description:
================================================================================
CUSP:       5.1.2.2              Verify daemon X was configured to
Req. Spec.: 3.2.1                execute in a secure mode.  The secure
                                 qualifier must be present in its
                                 configuration file /etc/daemonx.conf.

  Test Command(s):
  ------------------------------------------------------------------------------
  test_parameter '^secure=true' /etc/daemonx.conf

  <------------------------------ Results ------------------------------->
  Expected           Actual   Data Returned:
  -------------------------   ------------------------------------------
  found           not found   secure=false

     !  !  !  !  !  !  !    Test Pass: false  !  !  !  !  !  !  !
```

As can be seen in the example above, failed tests are emphasized by a sequence of seven exclamation marks (!) on either side of the “Test Pass” indication. Under verbose mode, these exclamation marks are omitted when a test passes.

10. When the test sequence is complete, a summary of the test session will be displayed. The results from the entire input test file will be summarized in output similar to the following:

```
================================================================================
================================================================================

Start Date/Time:                                        Completion Date/Time:
Wed Oct 31 10:17:57 EDT 2001                      Wed Oct 31 10:18:31 EDT 2001

   <------------------------ Tests Run Summary ------------------------->
        Executed             Passed (100%)               Failed (  0%)
   ================================================================================
          167                     167                          0

   Failed Tests:     None

   <---------------------- Requirements Summary ------------------------->
        Executed             Passed (100%)               Failed (  0%)
   ================================================================================
          131                     131                          0


   Req. Status    Requirement List
   ============== ============================================================
   Met:           3.2.1.1, 3.2.1.1.1, 3.2.1.1.2, 3.2.1.2, 3.2.1.2.1,
                  3.2.1.3, 3.2.1.4.1.1.1, 3.2.1.4.1.1.2, 3.2.1.4.1.1.2.1,
                  3.2.4.3, 3.2.5.11, 3.2.5.11.1, 3.2.5.15, 3.2.5.15.2,
                  3.2.5.2, 3.2.5.3, 3.2.5.4, 3.2.5.6, 3.2.5.7, 3.2.5.8,
                  3.2.5.9, 3.2.5.9.1, I4.2.1.2.3, I4.2.2.3.1, I4.2.2.3.1P1,
                  I4.2.2.3.2P1, I4.2.2.4P1
   Partially Met: Not Applicable
   Not Met:       None

        Met:            Requirements where ALL tests associated with
                        the specified requirement PASSED.
        Partially Met:  Requirements where ONE OR MORE tests associated
```

```
                    with the specified requirement FAILED.
          Not Met:  Requirements where ALL tests associated with
                           the specified requirement FAILED.
```

See Part 4 of this document for a discussion on how to interpret the summary results.

11. If more than one test series is specified when HOSTS is executed, or if no series are specified, a global summarization of the results from all executed tests series will be displayed. The summarization will be similar to the following:

```
*******************************************************************************
*******************************************************************************

                    Multi-Test Global Requirements Summary

================================================================================
================================================================================

Start Date/Time:                                        Completion Date/Time:
Wed Oct 31 14:40:26 EDT 2001                      Wed Oct 31 15:50:42 EDT 2001

   <------------------------- Tests Run Summary -------------------------->
        Executed             Passed (100%)               Failed (  0%)
   ================================================================================
          1702                    1702                         0

   Failed Tests:     None

   <---------------------- Requirements Summary ------------------------->
        Executed             Passed (100%)               Failed (  0%)
   ================================================================================
          1299                    1299                         0

   Req. Status    Requirement List
   ============== ============================================================
   Met:           3.2.1.1, 3.2.1.1.1, 3.2.1.1.2, 3.2.1.2, 3.2.1.2.1,
                  3.2.1.3, 3.2.1.4.1.1.1, 3.2.1.4.1.1.2, 3.2.1.4.1.1.2.1,
                  3.2.1.4.1.1.3, 3.2.1.4.1.1.3.1, 3.2.1.4.1.1.4,
                  3.2.1.4.1.1.4.1, 3.2.1.4.1.2, 3.2.1.4.1.4, 3.2.1.4.1.5,
                  3.2.1.4.1.7.1.1, 3.2.1.4.5, 3.2.1.5.1, 3.2.1.5.2,
                  3.2.16.1.4.1, 3.2.16.1.4.2, 3.2.16.1.5, 3.2.16.10,
                  3.2.16.2.1, 3.2.16.2.2, 3.2.16.2.3, 3.2.16.2.3.1,
                  3.2.16.2.3.2, 3.2.16.2.4, 3.2.16.5.1, 3.2.16.5.2,
                  3.2.16.5.3, 3.2.16.7.1, 3.2.16.7.1.1, 3.2.16.7.1.2,
                  3.2.16.7.1.4, 3.2.16.7.1.5, 3.2.16.7.1.6, 3.2.16.7.1.7,
                  3.2.16.7.1.8, 3.2.16.7.2, 3.2.16.7.2.1, 3.2.16.7.2.2,
                  3.2.16.7.2.4, 3.2.16.7.2.5, 3.2.16.8, 3.2.16.9, 3.2.2.1,
                  3.2.2.2, 3.2.3.1, 3.2.3.1.1, 3.2.3.1.2.1, 3.2.3.1.2.2,
                  3.2.3.1.2.3, 3.2.3.1.3.1.3, 3.2.3.1.3.3, 3.2.3.1.3.3.1,
                  3.2.3.1.4, 3.2.3.1.4.1.1, 3.2.3.1.4.1.4, 3.2.3.1.5
   Partially Met: Not Applicable
   Not Met:       None

        Met:            Requirements where ALL tests associated with
                        the specified requirement PASSED.
        Partially Met:  Requirements where ONE OR MORE tests associated
                        with the specified requirement FAILED.
        Not Met:        Requirements where ALL tests associated with
                               the specified requirement FAILED.
```

Again, see Part 4 of this document for a discussion on how to interpret the summary results.

12. Terminate the session log capture by entering “exit” at the prompt:

```
prompt--> exit
```

13. Use the Unix `more` command to review results:

```
prompt--> /bin/more <session log>
```

where `<session log>` is the name of the file containing the captured session log.

For example, if the name of the file containing the captured session log was `/var/tmp/hosts_test_1`, enter the following:

```
prompt--> /bin/more /var/tmp/hosts_test_1
```

Other commands can be substituted for the Unix `more` command. For example, the captured session log can be printed. The `more` command is used here since it will display the captured session log one screen at a time.

14. As applicable, correct any identified problems.

15. Rerun HOSTS to ensure the corrections have been applied.

16. Log off the system.

Part 4
Evaluate the Results Generated Through The HOSTS Process

This section provides instructions on evaluating the pass/fail results generated by the HOSTS utility during HOSTS testing.

Individual Test Summary

Each failed test within the test file will be marked by HOSTS via a summarization in the following format:

```
         Test: <Test Number>     Test Description:
================================================================================
CUSP:       <CUSP>               <Description>
Req. Spec.: <SRS>

  Test Command(s):
  ------------------------------------------------------------------------------
  <Test Command>

  <------------------------------- Results ------------------------------->
  Expected           Actual   Data Returned:
  -------------------------   ------------------------------------------------
  <Expected>       <Actual>   <Data Returned>

     !  !  !  !  !  !  !    Test Pass: false  !  !  !  !  !  !  !
```

The summary provides the tester with following information:

<Actual>	The actual value returned when the test record was evaluated.
<CUSP>	The reference within the *Consolidated Unix Security Practices (CUSP)* document for the test record. More than one reference may be listed.
<Data Returned>	Additional data returned by the executed command. Multiple lines of output may be displayed for this entity.
<Description>	Description of the test record. Typically it describes what is being tested. Multiple lines of output may be displayed for this entity.
<Expected>	The expected value that should be returned when the test record is evaluated.
<SRS>	The paragraph reference number within the *DII COE Security Requirements Specification* document for the test record. More than one reference may be listed.

<Test Command> The command used to perform the test. Multiple lines of output may be displayed for this entity.
<Test Number> The value used to identify which test record within the test input file/test series.

If the verbose command line qualifier is used, all tests within the test file, passed and failed, will be displayed in the report. Failed records will be identified by a string of exclamation marks on either side of the "Test Pass" indication.

Test Run Summary from the Test File

After HOSTS has processed an entire test file, it will summarize how many tests passed, how many tests failed, which tests failed, which tests partially failed, and which tests passed. The summary of results will be displayed in the following format:

```
================================================================================
================================================================================

Start Date/Time:                                          Completion Date/Time:
<Start Date/Time>                                               <End Date/Time>

   <------------------------- Tests Run Summary ------------------------->
        Executed             Passed ( WW%)               Failed ( XX%)
   ================================================================================
       <Run Total>           <Run passed>                <Run Failed>

   Failed Tests:     <Failed Test List>

   <----------------------- Requirements Summary ------------------------->
        Executed             Passed ( YY%)               Failed ( ZZ%)
   ================================================================================
       <SRS total>           <SRS passed>                <SRS failed>

   Req. Status     Requirement List
   ==============  ============================================================
   Met:            <Met List>
   Partially Met:  <Partially Met List>
   Not Met:        <Not Met List>

        Met:            Requirements where ALL tests associated with
                        the specified requirement PASSED.
        Partially Met:  Requirements where ONE OR MORE tests associated
                        with the specified requirement FAILED.
        Not Met:        Requirements where ALL tests associated with
                        the specified requirement FAILED.
```

<End Date/Time> The clock time when execution of the test completes.
<Failed Test List> The list of the individual tests, sorted by test number, that actually failed. Multiple lines of output may be displayed for this entity.

<Met List>	The list of SRS requirements that were fully met (e.g., all tests associated with the specific requirement passed). Multiple lines of output may be displayed for this entity.
<Not Met List>	The list of SRS requirements that were not met (e.g., tests associated with the specific requirement failed). Multiple lines of output may be displayed for this entity.
<Partially Met List>	The list of SRS requirements that were only partially met (e.g., one or more tests associated with the specific requirement failed while others passed). Multiple lines of output may be displayed for this entity.
<Run Total>	The total number of tests executed.
<Run Passed>	The total number of tests executed that passed. The percentage passing will be displayed immediately above this item. (See WW in the example output shown above.)
<Run Failed>	The total number of tests executed that failed. The percentage failing will be displayed immediately above this item. (See XX above.)
<SRS Total>	The total number of tests executed that are tagged to an SRS requirement.
<SRS Passed>	The total number of tests executed, tagged to an SRS requirement, that passed. The percentage passing will be displayed immediately above this item. (See YY above.)
<SRS Failed>	The total number of tests executed, tagged to an SRS requirement, that failed. The percentage failing will be displayed immediately above this item. (See ZZ above.)
<Start Date/Time>	The clock time when execution of the test begins.

Global Requirements Summary from the Test File

After HOSTS finished processing multiple (two or more) test files, a global consolidated summarization of the overall results will be displayed. This summarization includes the total number of tests passed and tests failed, which tests actually failed, and a summary on how the tests performed against requirements. The global summary of results will be displayed in the following format:

```
                Multi-Test Global Requirements Summary

================================================================================
================================================================================

Start Date/Time:                                        Completion Date/Time:
<Start Date/Time>                                             <End Date/Time>

   <------------------------ Tests Run Summary -------------------------->
        Executed             Passed ( WW%)              Failed ( XX%)
   ================================================================================
       <Run Total>           <Run passed>               <Run Failed>
```

```
Failed Tests:     <Failed Test List>

<----------------------- Requirements Summary ------------------------->
     Executed             Passed ( YY%)              Failed ( ZZ%)
================================================================================
    <SRS total>           <SRS passed>               <SRS failed>

Req. Status    Requirement List
=============  ============================================================
Met:            <Met List>
Partially Met:  <Partially Met List>
Not Met:        <Not Met List>

            Met:            Requirements where ALL tests associated with
                            the specified requirement PASSED.
            Partially Met:  Requirements where ONE OR MORE tests associated
                            with the specified requirement FAILED.
            Not Met:        Requirements where ALL tests associated with
                             the specified requirement FAILED
```

<End Date/Time> The clock time when execution of the test completes.

<Failed Test List> The list of the individual tests, sorted by test number, that actually failed. Multiple lines of output may be displayed for this entity.

<Met List> The list of SRS requirements that were fully met (e.g., all tests associated with the specific requirement passed). Multiple lines of output may be displayed for this entity.

<Not Met List> The list of SRS requirements that were not met (e.g., tests associated with the specific requirement failed). Multiple lines of output may be displayed for this entity.

<Partially Met List> The list of SRS requirements that were only partially met (e.g., one or more tests associated with the specific requirement failed while others passed). Multiple lines of output may be displayed.

<Run Total> The total number of tests executed.

<Run Passed> The total number of tests executed that passed. The percentage passing will be displayed immediately above this item. (See WW in the example output shown above.)

<Run Failed> The total number of tests executed that failed. The percentage failing will be displayed above this item. (See XX above.)

<SRS Total> The total number of tests executed tagged to an SRS requirement.

<SRS Passed> The total number of tests executed, tagged to an SRS requirement, that passed. The percentage passing will be displayed immediately above this item. (See YY above.)

<SRS Failed> The total number of tests executed, tagged to an SRS requirement, that failed. The percentage failing will be displayed immediately above this item. (See ZZ above.)

<Start Date/Time> The clock time when execution of the test begins.

Part 5
Customize the Input Test Files

This section provides instructions for customizing the input test files, often referred to as a test series, used by HOSTS.

Each input test file within HOSTS consists of a series of one-line "test" steps known as a test record. Each test record is executed sequentially as part of a functional test series. When all of the test records have been executed, the summarized results from the series are used to determine the overall functional pass or failure.

Customizing the input test files is a straightforward process, which is to modify an existing test record or add/remove test records from one of the input test files. New input test files can be constructed by adding test records for sequential execution.

Each test executed during the evaluation is controlled by a test record within the input test file. Each test record uses the following format:

```
<test.number>;<description>;<plugin>;<expected>;<cusp>;<srs>;
```

The contents of each field within the test record are described in Table 4. The semicolon (;) is used as the field separator and cannot be used within a test.

Table 4. HOSTS Input File Test Entry

Column	Description
test.number	Assigned test name and sequence number to uniquely identify the test. It should be unique for each test record in the file.
description	Text description of what is being examined.
plugin	Name of the test plugin used to perform the test. (Plugins outside the utility plugin directory, `/bin` or `/usr/bin`, must be fully qualified.)
expected	Text-string return result. If the value returned from the plugin matches this value, a pass is declared. Otherwise, a fail is declared. Multiple expected values are separated with the pipe character (|).
cusp	Reference paragraph within the *Consolidated Unix Security Practices (CUSP)* document. Multiple references must be comma separated.

Table 4. HOSTS Input File Test Entry	
Column	**Description**
srs	Reference paragraph within the SRS. Multiple references must be comma separated.

To help an observer identify which steps are being executed at a given time, header and comment capabilities have been integrated into HOSTS. Each header record is denoted by the following format:

```
SECTION=<header>;
```

Each comment record is denoted by the following format:

```
COMMENT=<header>;
```

The contents of each field within the header and comment records are described in Table 5. The use of `SECTION=` or `COMMENT=` is mandatory to differentiate this specialized record from the test record. Both record formats use the semicolon (;) as the field separator/terminator.

An third optional record type, known as a SKIPTEST record, directs HOSTS to skip over one or more specific tests. The SKIPTEST record is denoted by the following format:

```
SKIPTEST=<test number 1>[, <test number 2>, …, <test number n>];
```

where: <test number n> is the assigned test name and sequence number that is to be skipped. Multiple tests may be skipped. Either a comma or a space may be used as the separator. *Only one SKIPTEST line may be used per individual test file.*

The SKIPTEST feature is useful in conditions where mutually exclusive conditions represent the same secure environment. For example, if /etc/default/sulogin does not exist, the default Solaris behavior is to require root's password to access single user mode. Similarly, if /etc/default/sulogin does exist and the PASSREQ definition is set to "YES" within this file, root's password is required to access single user mode. Thus, the test to see if /etc/default/sulogin does not exist is just as valid as the test to verify that PASSREQ is correctly defined when /etc/default/sulogin does exists. Depending upon a given system's configuration, if both tests are performed, one of the above tests would always fail resulting in a false negative. The SKIPTEST definition must exist within a test file if the "-autogen" command line qualifier is to be used.

Do not use tabs within the test file. Comment lines begin with the pound sign. See Appendix C for a sample of a test input file.

Table 5. HOSTS Header and Comment Record Structures	
Column	**Description**
`SECTION="header";`	Section title to be added to the displayed output. This can stretch across multiple lines.
`COMMENT="comment";`	Comment to be added to the displayed output. This can stretch across multiple lines.
`SKIPTEST=<test numbers>;`	Command to skip over one or more specified tests. Where <test numbers> is a list of one or more comma and/or space separated test numbers to be skipped. Only one is allowed per test sequence. The list of skipped tests will only be displayed under verbose mode.

Part 6
Define Available Plugin Modules

This section provides a listing of the plugin modules available in the current release of HOSTS.

New modules can be added as long as the modules return a string value that can be compared with an expected value. If the returned value matches the expected value, a pass is declared. If the values do not match, a fail is declared.

The list of available plugins and a description of each plugin are provided in Table 6. Unless otherwise noted in the plugin description, all plugins are written in Perl. See Appendix D for a sample of one of these plugin files and Appendix E for a complete description of the calling sequence and return values for each plugin.

Table 6. HOSTS Plugin Modules

Name	Description
`active_daemon`	Tests to see if a specified daemon is active. The module filters out other sessions that may contain the same daemon test string (e.g., `vi`, `grep`).
`active_port`	Determines if specified ports are active. The port may be specified either by its `/etc/services` name or as an integer.
`create_accounts`	Exercises the account creation capabilities.
`delete_accounts`	Tests the account deletion capabilities.
`empty_file`	Tests to see if a specified file is an empty file (e.g., has a length of 0).
`extract_account_profile`	Extracts a profile of the specified user account. The profile includes account attributes (e.g., user and group identifiers), and password status information.
`File_exists`	Tests to see if a specified file exists on a system.
`files_differ`	Tests to see if two specified files differ in content from each other. Differences found will be returned.
`files_differ_feild`	Tests to see if the specified field (e.g., column) in the specified files differ. If the lengths of the files are the same, the differences in the examined field will be returned.

Table 6. HOSTS Plugin Modules	
Name	**Description**
`find_all_c_shell_scripts`	Tests to see if C shell scripts exist under the specified starting point. It differs from `find_c_shell_scripts` in that it will attempt to examine the contents of files normally tagged as text files to determine whether or not these files may actually be C shell scripts. It will not scan remotely mounted file systems.
`find_c_shell_scripts`	Tests to see if C shell scripts exist under the specified starting point. It will not scan remotely mounted file systems.
`find_file_and_test`	Parses down local partitions looking for files matching the specified pattern string. If any are found, the secondary test is applied to the detected file(s).
`find_misgrouped_files`	Tests to see if files exist in the specified directories that are not owned buy a GID found within the specified GID range. This is typically used to find files belonging to a non-root group within root directories.
`find_misowned_files`	Tests to see if files exist in the specified directories that are not owned buy a UID found within the specified UID range. This is typically used to find files owned by a non-root account within root directories.
`find_misprotected_files`	Tests to see if files exist in the specified directories that have protections granting group or world greater access than owner or world greater access than group. This is sometimes referred to as uneven protections.
`find_privileged_files`	Tests to see if privileged files and/or directories exist under the specified starting point. It will not scan remotely mounted file systems.
`find_privileged_scripts`	Tests to see if privileged shell scripts exist under the specified starting point. It will not scan remotely mounted file systems.

Table 6. HOSTS Plugin Modules	
Name	**Description**
`find_suspicious_file_names`	Tests to see if files and/or directories exist with strange names under the specified starting point. It will not scan remotely mounted file systems. Examples of strange names include names containing spaces, the plus symbol, and control characters.
`find_unowned_files`	Tests to see if unowned files and/or directories exist under the specified starting point. It will not scan remotely mounted file systems.
`find_world_writables`	Tests to see if world-writable files and/or directories exist under the specified starting point. It will not scan remotely mounted file systems.
`modify_accounts`	Tests the account modification capabilities.
`multi_part_test`	Runs two specified commands comparing the returned output with the specified expected output. If both results match the expected values, a match is declared.
`os_version`	This script ascertains which OS is installed on the system. If the OS is as specified, the test passes. If the OS is any other version, the test fails. If enabled, the plugin will issue a "die" thereby aborting test execution.
`print_pid`	This Bourne shell script finds and prints the process ID number for a specified process. The module filters out other processes that may contain the same passed test string (e.g., `vi`, `grep`).
`run_command`	Executes a command in the native operating environment.
`run_command_as_user`	Uses the `su` command to assume another user's profile before executing the specified command. This includes the option of assuming the target user's login environment.
`run_command_as_user_chprot`	Uses the `su` command to assume another user's profile before changing the protections on the specified file. This includes the option of assuming the target user's login environment.

Table 6. HOSTS Plugin Modules	
Name	**Description**
`run_command_as_user_create`	Uses the `su` command to assume another user's profile before creating the specified file. This includes the option of assuming the target user's login environment.
`run_command_as_user_redirect`	Uses the `su` command to assume another user's profile before executing the specified command. The results of the command are redirected as specified. This includes the option of assuming the target user's login environment.
`run_command_as_user_remove`	Uses the `su` command to assume another user's profile before removing the specified file. This includes the option of assuming the target user's login environment.
`run_command_as_user_write`	Uses the `su` command to assume another user's profile before writing data into the specified file. This includes the option of assuming the target user's login environment.
`run_command_redirect`	Executes a command allowing the results to be redirected.
`startup_file_exists`	Tests to see if boot time startup files exist with a name that contains the candidate string. It is used to verify candidate startup files that are not run at boot time.
`test_IFS`	This Bourne shell script tests input field separator (IFS) behavior under the specified shell. The IFS is the internal field separator definition (e.g., the semicolon normally).
`test_dormant_accounts`	This Bourne shell script finds and prints the names of accounts that have either been idle for a period of time in excess of one month or have never been used.
`test_for_checksums`	Computes a file's 16-bit and cyclic redundancy check (CRC) checksums. These values are then compared against argument passed values.
`test_for_string`	Tests a file to see if a specified American National Standard Code for Information Interchange (ASCII) string is detected within the specified file. The file may be any file type (e.g., binary or data). Strings of three or more characters are examined.

Table 6. HOSTS Plugin Modules	
Name	**Description**
`test_for_system_calls`	Tests to see if system calls with known vulnerabilities are contained within candidate binary file(s). To function, the binary file(s) must not be stripped.
`test_for_trojans`	Tests a file to see if a specified ASCII string commonly associated with Trojan horse binaries is detected within the specified file.
`test_gid_values`	Tests the local group file looking for unexpected GID values less than 20. In addition, it ensures that all GID values identified in `/etc/passwd` have an entry in `/etc/group`. GID values not in compliance will be displayed.
`test_group`	Tests to see if a specified group is defined for a specified file. The group is passed as a string matching the expected value (e.g., `staff`).
`test_home_dir_files`	Examines the user's home directory verifying the file attributes on the specified file. The minimal protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`). Logic has also been added to allow for privileged Unix accounts. If an account has a UID of less than 10, matches will be declared if the file(s) is owned by root and has a GID value of less than 10.
`test_logins_duplicate_uid`	Tests the local password file looking for duplicate UID definitions (e.g., more than one account with the same UID). The number of accounts with duplicate UID values is then compared to an expected value. Accounts with duplicate values are displayed.
`test_logins_duplicate_usernames`	Tests the local password file looking for duplicate usernames (e.g., more than one account with the same account username). The number of accounts with duplicate usernames is then compared to an expected value. Accounts with duplicate usernames are displayed.

Table 6. HOSTS Plugin Modules	
Name	**Description**
`test_logins_maximum_age_limit`	Tests the local shadow file looking for users with maximum password change thresholds greater than the allowable maximum (e.g., accounts not being forced to change their passwords as often as a given project or application stipulates.) The number of accounts with a threshold above the maximum threshold is then compared to an expected value. Accounts with a threshold above the maximum are displayed.
`test_logins_minimum_age_limit`	Tests the local shadow file looking for users with minimum password change threshold less than the allowable minimum (e.g., accounts allowed to change their passwords more often than a given project or application stipulates). The number of accounts with a threshold below the minimum threshold is then compared to an expected value. Accounts with a threshold below the minimum are displayed.
`test_logins_null_password`	Tests the local password file looking for accounts with null passwords. The number of accounts found to have null passwords is then compared to an expected value. Accounts with null passwords are displayed.
`test_logins_system_account`	Tests the local password file looking for system accounts (UID < 100). The number of accounts found to be system accounts is compared to an expected value. Accounts defined as system accounts are displayed.
`test_logins_world_writable_in_path`	Tests non-system accounts from the local passwd file to see if any users have a world-writable directory in their search path (`PATH`) definition. The number of accounts with world-writable directories within their `PATH` is compared to an expected value. Accounts with a world-writable directory in their `PATH` definition are displayed.
`test_multi_strings`	Tests to see if multiple ASCII strings, defined by using an input file, are found within a common test file. Each of the test strings is then defined by a regular expression.

Table 6. HOSTS Plugin Modules	
Name	**Description**
`test_network_attribute`	Tests to see if a specified network attribute for the specified device is set as expected.
`test_owner`	Tests to see if a specified owner is defined for a specified file. The owner information is passed as a string matching the expected value (e.g., `root`).
`test_parameter`	Tests to see if a specified parameter is defined within a specified file. The parameter is passed as a regular expression.
`test_parameter_count`	Tests to see how many times a specified parameter occurs within a specified file. The parameter is passed as a regular expression. This module assumes that the parameter will only occur once on a given line.
`test_parameter_filter_count`	Tests to see how many times a specified parameter occurs within a specified file after applying a filter rule. The parameter and rule are passed as regular expressions. This module assumes that the parameter will only occur once on a given line.
`test_parameter_nonmatch_count`	Tests to see how many times lines within the specified parameter occur within a specified file. The parameter is passed as a regular expression. This module assumes that the parameter will only occur once on a given line.
`test_protection`	Tests to see if a specified protection is set for a specified file. The protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`).
`test_protection_d`	Tests to see if a specified protection is set for a specified file. The protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`). Matching records are displayed.

Table 6. HOSTS Plugin Modules	
Name	**Description**
`test_protection_minimum`	Tests to see if a specified file has a protection equal to or more stringent than the specified protection. The minimum protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`).
`test_protection_minimum_d`	Tests to see if a specified file has a protection equal to or more stringent than the specified protection. The minimal protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`). Matching records are displayed.
`test_status`	Runs a specified command and then compares the returned output with the specified expected output.
`test_status_as_user`	Uses the `su` command to assume another user's profile before executing the specified command. This includes the option of assuming the target user's login environment.
`test_string`	Tests to see if a specified string is defined within a specified file. The string is passed as a regular expression. This module is identical to `test_parameter`.
`test_users_for_file`	Examines the user's home directory looking for the specified file.

Appendix A
Sample Installation

This appendix provides a listing for a sample installation of HOSTS (see Figure A-1). In this sample, `/usr/bin/perl` already exists.

Figure A-1. Sample Installation of HOSTS

```
Script started on Mon  Feb 11 10:20:52 2002
myhosts# whoami
root
myhosts# /bin/ls -l /usr/bin/perl
-r-xr-xr-x   3 root     bin        14312 Jan  8  2000 /usr/bin/perl
myhosts # cd /opt
myhosts # /bin/mkdir -p /opt/hosts
myhosts # /bin/ls /cdrom/hostsv1.5.0.0
Hosts.tar  cusp  hosts  requirements  tools
myhosts # cd /opt/hosts
myhosts # --> /bin/tar -xvf /cdrom/hostsv1.5.0.0/Hosts_15.tar
x ., 0 bytes, 0 tape blocks
x hosts.pl, 36248 bytes, 71 tape blocks
x sample_runs, 0 bytes, 0 tape blocks
x sample_runs/hpux_system, 0 bytes, 0 tape blocks
x sample_runs/hpux_system/dac_series_log.txt, 23769 bytes, 47 tape blocks
x sample_runs/hpux_system/os_series_log.txt, 13593 bytes, 27 tape blocks
x sample_runs/hpux_system/misc_series_log.txt, 3917 bytes, 8 tape blocks
x sample_runs/hpux_system/audit_series_log.txt, 5188 bytes, 11 tape blocks
x sample_runs/hpux_system/ia_series_log.txt, 5325 bytes, 11 tape blocks
x sample_runs/hpux_system/combined_series_log.txt, 58444 bytes, 115 tape blocks
x sample_runs/solaris_system, 0 bytes, 0 tape blocks
x sample_runs/solaris_system/ia_series_log.txt, 5282 bytes, 11 tape blocks
x sample_runs/solaris_system/combined_series_log.txt, 38918 bytes, 77 tape blocks
x sample_runs/solaris_system/misc_series_log.txt, 3875 bytes, 8 tape blocks
x sample_runs/solaris_system/os_series_log.txt, 11437 bytes, 23 tape blocks
x sample_runs/solaris_system/dac_series_log.txt, 7252 bytes, 15 tape blocks
x sample_runs/solaris_system/audit_series_log.txt, 4487 bytes, 9 tape blocks
x sample_runs/solaris_system/srs_series_log.txt, 151119 bytes, 296 tape blocks
x solaris, 0 bytes, 0 tape blocks
x solaris/plugins, 0 bytes, 0 tape blocks
x solaris/plugins/active_daemon, 2250 bytes, 5 tape blocks
x solaris/plugins/startup_file_exists, 2383 bytes, 5 tape blocks
x solaris/plugins/file_exists, 2224 bytes, 5 tape blocks
x solaris/plugins/test_protection, 4058 bytes, 8 tape blocks
x solaris/plugins/test_owner, 4581 bytes, 9 tape blocks
x solaris/plugins/test_parameter, 2825 bytes, 6 tape blocks
x solaris/plugins/test_for_string, 2926 bytes, 6 tape blocks
x solaris/plugins/test_for_system_calls, 3725 bytes, 8 tape blocks
x solaris/plugins/delete_accounts, 3202 bytes, 7 tape blocks
x solaris/plugins/create_accounts, 4026 bytes, 8 tape blocks
x solaris/plugins/modify_accounts, 3769 bytes, 8 tape blocks
x solaris/plugins/test_group, 4625 bytes, 10 tape blocks
x solaris/plugins/test_for_trojans, 3422 bytes, 7 tape blocks
x solaris/plugins/test_for_checksums, 3904 bytes, 8 tape blocks
x solaris/plugins/test_status, 2804 bytes, 6 tape blocks
```

Figure A-1. Sample Installation of HOSTS

```
x solaris/plugins/run_command, 2005 bytes, 4 tape blocks
x solaris/plugins/test_protection_minimum, 6756 bytes, 14 tape blocks
x solaris/plugins/test_network_attribute, 2927 bytes, 6 tape blocks
x solaris/plugins/test_IFS, 2563 bytes, 6 tape blocks
x solaris/plugins/print_pid, 1565 bytes, 4 tape blocks
x solaris/plugins/test_dormant_accounts, 4174 bytes, 9 tape blocks
x solaris/plugins/test_users_for_file, 4056 bytes, 8 tape blocks
x solaris/plugins/multi_part_test, 4715 bytes, 10 tape blocks
x solaris/plugins/run_command_redirect, 2305 bytes, 5 tape blocks
x solaris/plugins/test_home_dir_files, 12498 bytes, 25 tape blocks
x solaris/plugins/find_unowned_files, 6039 bytes, 12 tape blocks
x solaris/plugins/find_world_writables, 5930 bytes, 12 tape blocks
x solaris/plugins/find_privileged_files, 5998 bytes, 12 tape blocks
x solaris/plugins/find_suspicious_file_names, 6791 bytes, 14 tape blocks
x solaris/plugins/find_privileged_scripts, 7369 bytes, 15 tape blocks
x solaris/plugins/find_c_shell_scripts, 7231 bytes, 15 tape blocks
x solaris/plugins/find_all_c_shell_scripts, 8614 bytes, 17 tape blocks
x solaris/plugins/test_protection_minimum_d, 6888 bytes, 14 tape blocks
x solaris/plugins/test_protection_d, 4016 bytes, 8 tape blocks
x solaris/plugins/test_parameter_count, 3116 bytes, 7 tape blocks
x solaris/plugins/test_parameter_filter_count, 3393 bytes, 7 tape blocks
x solaris/plugins/test_parameter_nonmatch_count, 3232 bytes, 7 tape blocks
x solaris/plugins/run_command_as_user, 3370 bytes, 7 tape blocks
x solaris/plugins/run_command_as_user_create, 3340 bytes, 7 tape blocks
x solaris/plugins/run_command_as_user_remove, 3334 bytes, 7 tape blocks
x solaris/plugins/run_command_as_user_chprot, 3480 bytes, 7 tape blocks
x solaris/plugins/run_command_as_user_write, 3466 bytes, 7 tape blocks
x solaris/plugins/test_status_as_user, 3788 bytes, 8 tape blocks
x solaris/plugins/run_command_as_user_redirect, 4543 bytes, 9 tape blocks
x solaris/plugins/files_differ, 3601 bytes, 8 tape blocks
x solaris/plugins/empty_file, 3450 bytes, 7 tape blocks
x solaris/plugins/test_logins_duplicate_uid, 3258 bytes, 7 tape blocks
x solaris/plugins/test_logins_null_password, 3208 bytes, 7 tape blocks
x solaris/plugins/test_logins_system_account, 3216 bytes, 7 tape blocks
x solaris/plugins/extract_account_profile, 2771 bytes, 6 tape blocks
x solaris/plugins/test_logins_duplicate_usernames, 3741 bytes, 8 tape blocks
x solaris/plugins/test_logins_maximum_age_limit, 5350 bytes, 11 tape blocks
x solaris/plugins/test_logins_minimum_age_limit, 5376 bytes, 11 tape blocks
x solaris/plugins/test_logins_world_writable_in_path, 7211 bytes, 15 tape blocks
x solaris/plugins/test_gid_values, 6260 bytes, 13 tape blocks
x solaris/plugins/test_multi_strings, 4838 bytes, 10 tape blocks
x solaris/plugins/test_string, 2757 bytes, 6 tape blocks
x solaris/plugins/find_file_and_test, 8516 bytes, 17 tape blocks
x solaris/plugins/active_port, 2850 bytes, 6 tape blocks
x solaris/plugins/find_misprotected_files, 6736 bytes, 14 tape blocks
x solaris/plugins/find_misgrouped_files, 6465 bytes, 13 tape blocks
x solaris/plugins/find_misowned_files, 6427 bytes, 13 tape blocks
x solaris/plugins/files_differ_field, 6109 bytes, 12 tape blocks
x solaris/plugins/os_version, 2375 bytes, 5 tape blocks
x solaris/os_series, 113143 bytes, 221 tape blocks
x solaris/misc_series, 32520 bytes, 64 tape blocks
x solaris/dac_series, 96333 bytes, 189 tape blocks
x solaris/audit_series, 58674 bytes, 115 tape blocks
x solaris/ia_series, 32583 bytes, 64 tape blocks
x solaris/stg_series, 61358 bytes, 120 tape blocks
solaris/audit_series.txt linked to solaris/audit_series
solaris/dac_series.txt linked to solaris/dac_series
```

Figure A-1. Sample Installation of HOSTS

```
solaris/ia_series.txt linked to solaris/ia_series
solaris/misc_series.txt linked to solaris/misc_series
solaris/os_series.txt linked to solaris/os_series
solaris/stg_series.txt linked to solaris/stg_series
x solaris/srs_series, 69377 bytes, 136 tape blocks
solaris/srs_series.txt linked to solaris/srs_series
x solaris/00_README.txt, 695 bytes, 2 tape blocks
x solaris/version_series, 3322 bytes, 7 tape blocks
solaris/version_series.txt linked to solaris/version_series
x solaris/baseline, 0 bytes, 0 tape blocks
x solaris/baseline/8_Recommended, 650 bytes, 2 tape blocks
x solaris/baseline/known-c-shells-full, 2727 bytes, 6 tape blocks
x solaris/baseline/known-setuid-setgid-files-end-user, 7743 bytes, 16 tape blocks
x solaris/baseline/known-strange-name-files-full, 216 bytes, 1 tape blocks
x solaris/baseline/known-c-shells-end-user, 2541 bytes, 5 tape blocks
x solaris/baseline/known-strange-name-files-end-user, 613 bytes, 2 tape blocks
x solaris/baseline/known-non-sb-ww-directories-full, 995 bytes, 2 tape blocks
x solaris/baseline/known-setuid-setgid-files-full, 9071 bytes, 18 tape blocks
x solaris/baseline/known-non-binary-setuid-end-user, 78 bytes, 1 tape blocks
x solaris/baseline/known-non-binary-setuid-full, 78 bytes, 1 tape blocks
x solaris/baseline/known-non-sb-ww-directories-end-user, 82 bytes, 1 tape blocks
x hpux, 0 bytes, 0 tape blocks
x hpux/plugins, 0 bytes, 0 tape blocks
x hpux/plugins/active_daemon, 2250 bytes, 5 tape blocks
x hpux/plugins/startup_file_exists, 2383 bytes, 5 tape blocks
x hpux/plugins/file_exists, 2224 bytes, 5 tape blocks
x hpux/plugins/test_protection, 4058 bytes, 8 tape blocks
x hpux/plugins/test_owner, 4581 bytes, 9 tape blocks
x hpux/plugins/test_parameter, 2825 bytes, 6 tape blocks
x hpux/plugins/test_for_string, 2926 bytes, 6 tape blocks
x hpux/plugins/test_for_system_calls, 3725 bytes, 8 tape blocks
x hpux/plugins/delete_accounts, 3320 bytes, 7 tape blocks
x hpux/plugins/create_accounts, 4081 bytes, 8 tape blocks
x hpux/plugins/modify_accounts, 3769 bytes, 8 tape blocks
x hpux/plugins/test_group, 4625 bytes, 10 tape blocks
x hpux/plugins/test_for_trojans, 3422 bytes, 7 tape blocks
x hpux/plugins/test_for_checksums, 3904 bytes, 8 tape blocks
x hpux/plugins/test_status, 2804 bytes, 6 tape blocks
x hpux/plugins/run_command, 2005 bytes, 4 tape blocks
x hpux/plugins/test_protection_minimum, 6756 bytes, 14 tape blocks
x hpux/plugins/test_network_attribute, 3010 bytes, 6 tape blocks
x hpux/plugins/test_IFS, 2563 bytes, 6 tape blocks
x hpux/plugins/print_pid, 1565 bytes, 4 tape blocks
x hpux/plugins/test_dormant_accounts, 4256 bytes, 9 tape blocks
x hpux/plugins/test_users_for_file, 4166 bytes, 9 tape blocks
x hpux/plugins/multi_part_test, 4715 bytes, 10 tape blocks
x hpux/plugins/run_command_redirect, 2305 bytes, 5 tape blocks
x hpux/plugins/test_home_dir_files, 12472 bytes, 25 tape blocks
x hpux/plugins/find_unowned_files, 6063 bytes, 12 tape blocks
x hpux/plugins/find_world_writables, 5954 bytes, 12 tape blocks
x hpux/plugins/find_privileged_files, 6022 bytes, 12 tape blocks
x hpux/plugins/find_suspicious_file_names, 6815 bytes, 14 tape blocks
x hpux/plugins/find_privileged_scripts, 7406 bytes, 15 tape blocks
x hpux/plugins/find_c_shell_scripts, 7255 bytes, 15 tape blocks
x hpux/plugins/find_all_c_shell_scripts, 8621 bytes, 17 tape blocks
x hpux/plugins/test_protection_minimum_d, 6888 bytes, 14 tape blocks
x hpux/plugins/test_protection_d, 4016 bytes, 8 tape blocks
```

Figure A-1. Sample Installation of HOSTS

```
x hpux/plugins/test_parameter_count, 3116 bytes, 7 tape blocks
x hpux/plugins/test_parameter_filter_count, 3393 bytes, 7 tape blocks
x hpux/plugins/test_parameter_nonmatch_count, 3232 bytes, 7 tape blocks
x hpux/plugins/run_command_as_user, 3370 bytes, 7 tape blocks
x hpux/plugins/run_command_as_user_create, 3340 bytes, 7 tape blocks
x hpux/plugins/run_command_as_user_remove, 3334 bytes, 7 tape blocks
x hpux/plugins/run_command_as_user_chprot, 3480 bytes, 7 tape blocks
x hpux/plugins/run_command_as_user_write, 3466 bytes, 7 tape blocks
x hpux/plugins/test_status_as_user, 3788 bytes, 8 tape blocks
x hpux/plugins/run_command_as_user_redirect, 4543 bytes, 9 tape blocks
x hpux/plugins/files_differ, 3601 bytes, 8 tape blocks
x hpux/plugins/empty_file, 3450 bytes, 7 tape blocks
x hpux/plugins/test_logins_duplicate_uid, 3357 bytes, 7 tape blocks
x hpux/plugins/test_logins_null_password, 3308 bytes, 7 tape blocks
x hpux/plugins/test_logins_system_account, 3315 bytes, 7 tape blocks
x hpux/plugins/extract_account_profile, 2868 bytes, 6 tape blocks
x hpux/plugins/test_logins_duplicate_usernames, 3741 bytes, 8 tape blocks
x hpux/plugins/test_logins_maximum_age_limit, 5304 bytes, 11 tape blocks
x hpux/plugins/test_logins_minimum_age_limit, 5309 bytes, 11 tape blocks
x hpux/plugins/test_logins_world_writable_in_path, 7211 bytes, 15 tape blocks
x hpux/plugins/test_gid_values, 6260 bytes, 13 tape blocks
x hpux/plugins/test_multi_strings, 4838 bytes, 10 tape blocks
x hpux/plugins/test_string, 2757 bytes, 6 tape blocks
x hpux/plugins/find_file_and_test, 8547 bytes, 17 tape blocks
x hpux/plugins/active_port, 2850 bytes, 6 tape blocks
x hpux/plugins/find_misprotected_files, 6760 bytes, 14 tape blocks
x hpux/plugins/find_misgrouped_files, 6489 bytes, 13 tape blocks
x hpux/plugins/find_misowned_files, 6427 bytes, 13 tape blocks
x hpux/plugins/files_differ_field, 6106 bytes, 12 tape blocks
x hpux/plugins/os_version, 2375 bytes, 5 tape blocks
x hpux/os_series, 98274 bytes, 192 tape blocks
x hpux/misc_series, 27049 bytes, 53 tape blocks
x hpux/dac_series, 102709 bytes, 201 tape blocks
x hpux/audit_series, 49331 bytes, 97 tape blocks
x hpux/ia_series, 31605 bytes, 62 tape blocks
x hpux/00_README.txt, 662 bytes, 2 tape blocks
hpux/audit_series.txt linked to hpux/audit_series
hpux/dac_series.txt linked to hpux/dac_series
hpux/ia_series.txt linked to hpux/ia_series
hpux/misc_series.txt linked to hpux/misc_series
x hpux/version_series, 3364 bytes, 7 tape blocks
x hpux/srs_series, 58866 bytes, 115 tape blocks
hpux/os_series.txt linked to hpux/os_series
hpux/srs_series.txt linked to hpux/srs_series
hpux/version_series.txt linked to hpux/version_series
x hpux/baseline, 0 bytes, 0 tape blocks
x hpux/baseline/8_Recommended, 650 bytes, 2 tape blocks
x hpux/baseline/known-c-shells, 2440 bytes, 5 tape blocks
x hpux/baseline/known-non-binary-setuid, 11 bytes, 1 tape blocks
x hpux/baseline/known-setuid-setgid-files, 14559 bytes, 29 tape blocks
x hpux/baseline/known-strange-name-files, 2583 bytes, 6 tape blocks
x hpux/baseline/known-non-sb-ww-directories, 827 bytes, 2 tape blocks
x COPYING, 26430 bytes, 52 tape blocks
LICENSE linked to COPYING
x doc, 0 bytes, 0 tape blocks
x doc/test_cross_ref.pdf, 16535 bytes, 33 tape blocks
x doc/hosts_sop_v14.pdf, 213117 bytes, 417 tape blocks
```

Figure A-1. Sample Installation of HOSTS

```
x 00_README_XEQ.txt, 5463 bytes, 11 tape blocks
x 00_README.txt, 909 bytes, 2 tape blocks
x bin, 0 bytes, 0 tape blocks
bin/hosts.pl linked to hosts.pl
myhosts # /bin/ls -la
total 208
drwxr-xr-x   7 root     staff        512 Feb 11 10:22 .
drwxr-xr-x  73 root     staff       2560 Feb  5 11:49 ..
-rw-r--r--   1 root     staff        909 Feb  6 12:33 00_README.txt
-rw-r--r--   1 root     staff       5463 Sep 28 15:22 00_README_XEQ.txt
-rw-r--r--   2 root     staff      26430 Aug 30 14:25 COPYING
-rw-r--r--   2 root     staff      26430 Aug 30 14:25 LICENSE
drwxr-xr-x   2 root     staff        512 Feb  4 14:05 bin
drwxr-xr-x   2 root     staff        512 Jan 14 14:15 doc
-rwxr-x---   2 root     staff      36248 Feb  4 16:19 hosts.pl
drwxr-xr-x   4 root     staff        512 Feb  6 12:40 hpux
drwxr-xr-x   6 root     staff        512 Feb  5 12:13 sample_runs
drwxr-xr-x   4 root     staff        512 Feb  4 16:16 solaris
# exit

script done on Mon Feb 11 10:30:32 2002
```

Appendix B
Sample Execution of HOSTS on a COE Client

This appendix provides a sample execution of HOSTS[8], in which the sample test file detailed in Appendix C was used as the input test file. The produced results are provided in Figure B-1.

In this sample, all included tests were passed. As can be seen in the results, the section headers provide a convenient vehicle for monitoring the test process. They can also provide direction when user input is required.

Figure B-1. Sample HOSTS Test Run

```
Script started on Wed Oct 31 09:14:56 2001
# ./hosts.pl ia_series


                        The MITRE Corporation
                  Center for Integrated Intelligence Systems
                         Network Security Engineering

                  Host-Oriented Security Test Suite (HOSTS)
                   Version 1.3    Released 30 October, 2001


 Generated on:                    Output Mode:                    Generated by:
 knapdale                        Failures Only                            root

 Test Input File:                                                Test Run Time:
 ./tests/ia_series                                 Wed Oct 31  9:15:05 EDT 2001

 ================================================================================
 ================================================================================

 Now Processing: Test Setup
 Now Processing: Sol-IA-1: Account management per established system security
                 policy parameters.
                 ----
                 Partial: APM interface testing of account creation must be
                 performed manually.
                 ----
                 OPERATOR ACTION:
                 ----
                 If rsh authentication bypassing has not been set up (e.g.,
```

[8] This run shown in this example was created using HOSTS 1.3. The results that would be produced by later releases of HOSTS are identical.

Figure B-1. Sample HOSTS Test Run

```
                /.rhost does not exist), you will need to enter the root
                password.  Then press return.  Nothing will be displayed on
                the screen.
Now Processing: Sol-IA-2: Password management per system security policy.
                ----
                Partial: APM interface and interactive testing of password
                changing must be performed manually.
                ----
                OPERATOR ACTION:
                ----
                Please enter a password, press return, enter the password
                again and press return a second time. Nothing will be
                displayed on the screen.
Now Processing: Sol-IA-3: Evaluate consecutive failed login attempts.
                ----
                Partial: APM interface and interactive account lock-out
                testing must be performed manually.
Now Processing: Sol-IA-4: User account management by a trusted user.
                ----
                Partial: APM interface testing of account
                creation/modification/deletion must be performed manually.
                ----
                OPERATOR ACTION:
                ----
                Please enter a password, press return, enter the password
                again and press return a second time. Nothing will be
                displayed on the screen.
Now Processing: Test Cleanup

================================================================================
================================================================================

Start Date/Time:                                        Completion Date/Time:
Wed Oct 31  9:15:05 EDT 2001                      Wed Oct 31  9:16:25 EDT 2001

   <------------------------- Tests Run Summary -------------------------->
        Executed             Passed (100%)               Failed (  0%)
   ================================================================================
          167                     167                          0

   Failed Tests:     None

   <----------------------- Requirements Summary ------------------------->
        Executed             Passed (100%)               Failed (  0%)
   ================================================================================
          131                     131                          0
```

Figure B-1. Sample HOSTS Test Run

```
    Req. Status    Requirement List
    ============== ============================================================
    Met:           3.2.1.1, 3.2.1.1.1, 3.2.1.1.2, 3.2.1.2, 3.2.1.2.1,
                   3.2.1.3, 3.2.1.4.1.1.1, 3.2.1.4.1.1.2, 3.2.1.4.1.1.2.1,
                   3.2.1.4.1.1.3, 3.2.1.4.1.1.3.1, 3.2.1.4.1.1.4,
                   3.2.1.4.1.1.4.1, 3.2.1.4.1.2, 3.2.1.4.1.4, 3.2.1.4.1.5,
                   3.2.1.4.1.7.1.1, 3.2.1.5.2, 3.2.1.6.3, 3.2.1.6.5,
                   3.2.15.2, 3.2.16.1, 3.2.16.1.1, 3.2.16.1.2, 3.2.16.1.4,
                   3.2.16.1.4.1, 3.2.16.1.4.2, 3.2.16.1.5, 3.2.16.10,
                   3.2.16.2.1, 3.2.16.2.2, 3.2.16.2.3, 3.2.16.2.3.1,
                   3.2.16.2.3.2, 3.2.16.2.4, 3.2.16.5.1, 3.2.2.2, 3.2.3.3.1,
                   3.2.4.3, 3.2.5.11, 3.2.5.11.1, 3.2.5.15, 3.2.5.15.2,
                   3.2.5.2, 3.2.5.3, 3.2.5.4, 3.2.5.6, 3.2.5.7, 3.2.5.8,
                   3.2.5.9, 3.2.5.9.1, I4.2.1.2.3, I4.2.2.3.1, I4.2.2.3.1P1,
                   I4.2.2.3.2P1, I4.2.2.4P1
    Partially Met: Not Applicable
    Not Met:       None


         Met:            Requirements where ALL tests associated with
                         the specified requirement PASSED.
         Partially Met:  Requirements where ONE OR MORE tests associated
                         with the specified requirement FAILED.
         Not Met:        Requirements where ALL tests associated with
                         the specified requirement FAILED.

# exit
script done on Wed Oct 31 09:16:52 2001
```

Appendix C
Sample HOSTS Test Input File

This appendix provides a sample test input file. Figure C-1 contains a sample extraction from the Identification and Authentication test process associated with the COE KPC. Some wrapping has occurred when lines exceeded the width of the figure. While this wrapping improves readability, it gives the impression that individual test entries may span multiple lines. In actuality, they do not. Each entry is restricted to a single line.

This sample also shows the use of a module in which user input is required (e.g., changing a password in this case). To guide the tester, section headers have been integrated at these strategic points instructing the tester on what must be entered.

Finally, the comments at the beginning of the file provide on-line documentation on how each entry is structured. These are ignored when the input test file is processed by HOSTS.

Figure C-1. Sample HOSTS Input Test File

```
#
#         KPC Test Suite                                HOSTS V1.2
#         Identification and Authentication Series
#
# This file contains system level non-intrusive tests, extracted from
# the 17 July, 2001, draft of the KPC Security Test Plan.  This test
# script is run against the local installation under KPC evaluation.
# It examines the binary and supporting configuration files for known
# security vulnerabilities and verifies, when possible, security
# related behavior and functionality.
#
# The allowed formats of each test entry in this file are as follows:
#
# Primary Test Entry Format:
#
#        test.number ; description ; plugin ; expected ; cusp ; srs ;
#
#     where: test.number    Assigned test name and sequence number to
#                           uniquely identify test (see below).  Each
#                           test number needs to be unique.  Do not
#                           use the equal sign (=) in the test number!
#            description    Text description of what is being examined.
#            plugin         Name of the test plugin used to perform the
#                           test.  (Plugins outside the utility plugin
#                           directory, /bin or /usr/bin must be fully
#                           qualified.)
#            expected       Text string return result.  If the value
#                           returned from the plugin matches this value,
#                           a "pass" is declared.  Otherwise, a "fail" is
#                           declared.  Multiple expected values are
#                           separated with the pipe character (|).
#            cusp           Reference paragraph within the Consolidated
#                           Unix Security Practices (CUSP) document.
#            srs            Reference paragraph within the security
#                           requirements specification (SRS) under
#                           evaluation.
#
# Section Title Format:
#
#        SECTION=<title>;
#
#     where: <title>        Section title to be added to the displayed output.
#
# Comment Format:
#
#        COMMENT=<comment>;
#
#     where: <comment>      Comment to be added to the displayed output.
#
# Skip Test Format:
```

Figure C-1. Sample HOSTS Input Test File

```
#
#       SKIPTEST=<test numbers>;
#
#     where: <test numbers>     List of comma and/or space separated test
#                               numbers to be skipped.  Only one is allowed
#                               per test sequence.  The list of skipped tests
#                               will only be displayed under verbose mode.
#
# In all cases, the field delimiter is the semi-colon character (;).
#
# The MITRE Corporation, 1820 Dolley Madison, McLean, VA  22102
# ==============================================================================
# ==============================================================================
#
#
# Defined skipped tests
# ================================================================================================================================
SKIPTEST=IA-1.C.3a, IA-1.C.5a, IA-3.P.1.1, IA-3.P.1.2, IA-3.P.1.3, IA-3.P.3.4.1, IA-3.P.3.4.2, IA-3.P.3.6.1, IA-3.P.3.6.2, IA-
3.P.3.7.1, IA-3.P.3.7.2;
#  Baseline - Hosts 1.2 Kernel 4200P4
#SKIPTEST=IA-1.C.3a, IA-1.C.5a;
#  All tests enabled
#
#  Note - Test pairs IA-1.C.3a/IA-1.C.3b, IA-1.C.5a/IA-1.C.5b are mutually
#         exclusive.
#
#
SECTION=Test Setup;
# Create two user accounts for later use.
# ================================================================================================================================
IA-Setup.1;Create first unique account;create_accounts "IAaccnt1" "1234" "10" "/bin/sh" "/tmp/IAaccnt1";created;;;
IA-Setup.2;Create second unique account;create_accounts "IAaccnt2" "1235" "10" "/bin/sh" "/tmp/IAaccnt2";created;;;
#
#
SECTION=Sol-IA-1: Account management per established system security policy parameters.;
COMMENT=Partial: APM interface testing of account creation must be performed manually.;
# ================================================================================================================================
IA-1.A.8;Verify account with existing (duplicate) UID can not be created;create_accounts "baduid" "1234" "10" "/bin/sh"
"/tmp/baduid";not created;;3.2.1.1, 3.2.1.2, 3.2.1.2.1, 3.2.1.3;
IA-1.A.9.1;Verify account with unique name and unused UID can be created;create_accounts "IAaccnt3" "1236" "60001" "/bin/sh"
"/tmp/IAaccnt3";created;;3.2.1.1, 3.2.1.2, 3.2.1.2.1, 3.2.1.3;
IA-1.A.9.2;Remove created account;delete_accounts "IAaccnt3";removed;;;
IA-1.A.15;Verify account with existing name can not be created;create_accounts "IAaccnt1" "1237" "10" "/bin/sh"
"/tmp/IAaccnt1";unexpected;;3.2.1.1, 3.2.1.2, 3.2.1.2.1, 3.2.1.3;
IA-1.A.16-18;Verify no duplicate UID accounts are detected in the password file;test_logins_duplicate_uid 0;match;;3.2.1.1,
3.2.1.2, 3.2.1.2.1, 3.2.1.3;
#
IA-1.B.1;Verify the number of system accounts detected is limited;test_logins_system_account 11;match;;I4.2.2.3.1;
IA-1.B.3;Verify no accounts with null passwords are detected in the password file;test_logins_null_password
0;match;;3.2.1.4.1.7.1.1, 3.2.16.5.1, I4.2.1.2.3;
IA-1.B.12;Verify no guest accounts detected in password file;test_parameter_count "^(guest|visit|temp|tmp|generic|other)" 0
/etc/passwd;match;;I4.2.2.4P1;
IA-1.B.14;Verify no guest accounts detected in shadow file;test_parameter_count "^(guest|visit|temp|tmp|generic|other)" 0
/etc/shadow;match;;I4.2.2.4P1;
#
IA-1.C.3a;Verify /etc/hosts.equiv does not exist;file_exists /etc/hosts.equiv;not found;;3.2.1.1.1, 3.2.2.2, I4.2.1.2.3;
IA-1.C.3b.1;If /etc/hosts.equiv exists, verify protections.;multi_part_test 'tests/plugins/file_exists /etc/hosts.equiv'
'found' 'tests/plugins/test_protection_minimum -rw-r---- /etc/hosts.equiv' 'match';match|no match both;;3.2.1.5.2, 3.2.5.6;
IA-1.C.3b.2;If /etc/hosts.equiv exists, verify owner.;multi_part_test 'tests/plugins/file_exists /etc/hosts.equiv' 'found'
'tests/plugins/test_owner root /etc/hosts.equiv' 'match';match|no match both;;3.2.5.3, 3.2.5.6;
IA-1.C.3b.3;If /etc/hosts.equiv exists, verify group.;multi_part_test 'tests/plugins/file_exists /etc/hosts.equiv' 'found'
'tests/plugins/test_group "root|other|sys" /etc/hosts.equiv' 'match';match|no match both;;3.2.5.3, 3.2.5.6;
IA-1.C.3b.4;If /etc/hosts.equiv exists, verify proper value (minus sign).;multi_part_test 'tests/plugins/file_exists
/etc/hosts.equiv' 'found' 'tests/plugins/test_parameter "^-" /etc/hosts.equiv' 'found';match|no match both;;3.2.1.1.1,
3.2.2.2, I4.2.1.2.3;
IA-1.C.3b.5;If /etc/hosts.equiv exists, verify there is only one minus sign.;multi_part_test 'tests/plugins/file_exists
/etc/hosts.equiv' 'found' 'tests/plugins/test_parameter_count "^-" 1 /etc/hosts.equiv' 'match';match|no match both;;3.2.1.1.1,
3.2.2.2, I4.2.1.2.3;
IA-1.C.3b.6;If /etc/hosts.equiv exists, verify plus sign is not detected.;multi_part_test 'tests/plugins/file_exists
/etc/hosts.equiv' 'found' 'tests/plugins/test_parameter "^\+" /etc/hosts.equiv' 'not found';match|no match both|no match
one;;3.2.1.1.1, 3.2.2.2, I4.2.1.2.3;
IA-1.C.4;Verify there are no personal authentication bypass files belonging to users;test_users_for_file '.rhosts .shosts';not
found;;3.2.1.1.1, 3.2.2.2, I4.2.1.2.3;
IA-1.C.5a;Verify /.rhosts does not exists;file_exists /.rhosts;not found;;3.2.1.1.1, 3.2.2.2, I4.2.1.2.3;
IA-1.C.5b.1;If /.rhosts exists, verify protections.;multi_part_test 'tests/plugins/file_exists /.rhosts' 'found'
'tests/plugins/test_protection_minimum -rw-r---- /.rhosts' 'match';match|no match both;;3.2.1.5.2, 3.2.5.6;
IA-1.C.5b.2;If /.rhosts exists, verify owner.;multi_part_test 'tests/plugins/file_exists /.rhosts' 'found'
'tests/plugins/test_owner root /.rhosts' 'match';match|no match both;;3.2.5.3, 3.2.5.6;
IA-1.C.5b.3;If /.rhosts exists, verify group.;multi_part_test 'tests/plugins/file_exists /.rhosts' 'found'
'tests/plugins/test_group "root|other|sys" /.rhosts' 'match';match|no match both;;3.2.5.3, 3.2.5.6;
IA-1.C.5b.4;If /.rhosts exists, verify proper value (minus sign).;multi_part_test 'tests/plugins/file_exists /.rhosts' 'found'
'tests/plugins/test_parameter "^-" /.rhosts' 'found';match|no match both;;3.2.1.1.1, 3.2.2.2, I4.2.1.2.3;
IA-1.C.5b.5;If /.rhosts exists, verify there is only one minus sign.;multi_part_test 'tests/plugins/file_exists /.rhosts'
'found' 'tests/plugins/test_parameter_count "^-" 1 /.rhosts' 'match';match|no match both;;3.2.1.1.1, 3.2.2.2, I4.2.1.2.3;
IA-1.C.5b.6;If /.rhosts exists, verify plus sign is not detected.;multi_part_test 'tests/plugins/file_exists /.rhosts' 'found'
```

Figure C-1. Sample HOSTS Input Test File

```
'tests/plugins/test_parameter "^\+" /.rhosts' 'not found';match|no match both|no match one;;3.2.1.1.1, 3.2.2.2, I4.2.1.2.3;
IA-1.C.6;Verify no accounts with null passwords are detected in the password file;test_logins_null_password
0;match;;3.2.1.4.1.7.1.1, 3.2.16.5.1, I4.2.1.2.3;
IA-1.C.7-9;Verify root account is restricted to direct login on the console only;test_parameter
"^CONSOLE=(none|/dev/console|/dev/null)" /etc/default/login;found;;3.2.1.1.2, 3.2.5.15, 3.2.5.15.2, I4.2.2.3.2P1;
IA-1.C.9.1;Save current /etc/inetd.conf for later restoration;run_command '/bin/cp /etc/inet/inetd.conf /etc/inet/inetd.conf-
IA-1';match;;;
IA-1.C.9.2;Remove current rsh command option;run_command_redirect '/bin/egrep -v "^shell" /etc/inet/inetd.conf-IA-1 >
/etc/inet/inetd.conf';match;;;
IA-1.C.9.3;Insure rsh daemon is available for use in this test;run_command_redirect '/bin/echo "shell        stream    tcp
        nowait    root      /usr/sbin/in.rshd in.rshd" >> /etc/inet/inetd.conf';match;;;
IA-1.C.9.4;Tickle the inetd daemon to activate test change;run_command '/bin/kill -HUP `tests/plugins/print_pid
inetd`';match;;;
COMMENT=OPERATOR ACTION:;
COMMENT=If rsh authentication bypassing has not been set up (e.g., /.rhost does not exist), you will need to enter the root
password.  Then press return.  Nothing will be displayed on the screen.;
IA-1.C.9.5;Verify restriction of root login to console logic works";test_status '/bin/rsh -l root localhost /bin/ls 2>&1 |
/bin/egrep -v "root@127.0.0.1"' 'ROOT LOGIN REFUSED FROM localhost' 'permission denied';match;;3.2.1.1.2, 3.2.5.15,
3.2.5.15.2, I4.2.2.3.2P1;
IA-1.C.9.6;Restore /etc/inetd.conf to original configuration;run_command '/bin/cp /etc/inet/inetd.conf-IA-1
/etc/inet/inetd.conf';match;;;
IA-1.C.9.7;Tickle the inetd daemon to return daemon to pre-test configuration;run_command '/bin/kill -HUP
`tests/plugins/print_pid inetd`';match;;;
IA-1.C.9.8;Remove /etc/inetd.conf hold file;run_command_as_user_remove root LOGINENV '/etc/inet/inetd.conf-IA-1';match;;;
IA-1.C.13;Verify no duplicate UID accounts are detected in the password file;test_logins_duplicate_uid 0;match;;3.2.1.1,
3.2.1.2, 3.2.1.2.1, 3.2.1.3;
#
#
SECTION=Sol-IA-2: Password management per system security policy.;
COMMENT=Partial: APM interface and interactive testing of password changing must be performed manually.;
# ================================================================================================================
IA-2.D.1.1;Verify protections on password configuration file;test_protection -r--r--r-- /etc/default/passwd;match;;3.2.1.5.2,
3.2.5.6;
IA-2.D.1.2;Verify ownership on password configuration file;test_owner root /etc/default/passwd;match;;3.2.5.3, 3.2.5.6;
IA-2.D.1.3;Verify group on password configuration file;test_group sys /etc/default/passwd;match;;3.2.5.3, 3.2.5.6;
IA-2.D.4;Verify passwords are required on login for all users;test_parameter "^PASSREQ=YES"
/etc/default/login;found;;3.2.1.1.1, 3.2.1.4.1;
IA-2.D.6.1;Verify accounts are not set-up with null passwords;test_logins_null_password 0;match;;3.2.1.4.1.7.1.1, 3.2.16.5.1,
I4.2.1.2.3;
IA-2.D.6.2;Extract IAaccnt1 status information;extract_account_profile IAaccnt1 "/tmp/IA-2.IAaccnt1";complete;;;
IA-2.D.6.3;Verify user account is locked;test_parameter_count "LK" 1 /tmp/IA-2.IAaccnt1;match;;I4.2.2.3.1P1;
IA-2.D.6.4;Extract daemon account status information;extract_account_profile daemon "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.5;Verify daemon account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.6;Extract bin account status information;extract_account_profile bin "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.7;Verify bin account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.8;Extract sys account status information;extract_account_profile sys "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.9;Verify sys account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.10;Extract adm account status information;extract_account_profile adm "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.11;Verify adm account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.12;Extract lp account status information;extract_account_profile lp "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.13;Verify lp account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.14;Extract uucp account status information;extract_account_profile uucp "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.15;Verify uucp account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.16;Extract nuucp account status information;extract_account_profile nuucp "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.17;Verify nuucp account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.18;Extract listen account status information;extract_account_profile listen "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.19;Verify listen account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.20;Extract nobody account status information;extract_account_profile nobody "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.21;Verify nobody account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.22;Extract nobody4 account status information;extract_account_profile nobody4 "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.23;Verify nobody4 account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.24;Extract noaccess account status information;extract_account_profile noaccess "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.25;Verify noaccess account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.26;Extract SA account status information;extract_account_profile SA "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.27;Verify SA account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.28;Extract COE account status information;extract_account_profile COE "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.29;Verify COE account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
IA-2.D.6.30;Extract SSO account status information;extract_account_profile SSO "/tmp/IA-2.lock_check";complete;;;
IA-2.D.6.31;Verify SSO account is locked;test_parameter_count "LK" 1 /tmp/IA-2.lock_check;match;;I4.2.2.3.1P1;
#
IA-2.D.10;Verify minimum password age limit has been defined;test_parameter "^MINWEEKS="
/etc/default/passwd;found;;3.2.1.4.1.1.4, 3.2.1.4.1.2;
IA-2.D.29;Verify minimum password age limit has been defined to one week;test_parameter "^MINWEEKS=1"
/etc/default/passwd;found;;3.2.1.4.1.1.4.1, 3.2.1.4.1.2;
#
IA-2.E.4.1;Verify minimum password expiration notice warning window has been defined;test_parameter "^WARNWEEKS="
/etc/default/passwd;found;;3.2.1.4.1.1.3;
IA-2.E.4.2;Verify maximum password age limit has been defined;test_parameter "^MAXWEEKS="
/etc/default/passwd;found;;3.2.1.4.1.1.2;
IA-2.E.20;Verify minimum password expiration notice warning window has been defined to one week;test_parameter "^WARNWEEKS=1"
/etc/default/passwd;found;;3.2.1.4.1.1.3.1;
IA-2.E.21;Verify maximum password age limit has been defined to approximately 90 days;test_parameter "^MAXWEEKS=13"
/etc/default/passwd;found;;3.2.1.4.1.1.2.1;
#
IA-2.F.4.1;Verify minimum password age limit has been defined;test_parameter "^MINWEEKS="
```

Figure C-1. Sample HOSTS Input Test File

```
/etc/default/passwd;found;;3.2.1.4.1.1.4;
IA-2.F.4.2;Verify minimum password age limit has been defined to one week;test_parameter "^MINWEEKS=1"
/etc/default/passwd;found;;3.2.1.4.1.1.4.1;
#
IA-2.G.3.1;Extract IAaccnt1 status information;extract_account_profile IAaccnt1 "/tmp/IA-2.IAaccnt1";complete;;;
IA-2.G.3.2;Verify user account is locked;test_parameter_count "LK" 1 /tmp/IA-2.IAaccnt1;match;;;
IA-2.G.3.3;Attempt to change password on another account;run_command_as_user_redirect IAaccnt1 NO "/bin/passwd secman
>/tmp/IA-2.np_passwd 2>&1";unexpected;;3.2.1.4.1.4;
IA-2.G.3.4;Verify the reason for the failure is a lack of permission;test_parameter_count "Permission denied" 2 /tmp/IA-
2.np_passwd;match;;3.2.1.4.1.4;
COMMENT=OPERATOR ACTION:;
COMMENT=Please enter a password, press return, enter the password again and press return a second time. Nothing will be
displayed on the screen.;
IA-2.G.3.5;Verify privileged user is able to change password on another account;run_command '/usr/bin/passwd
IAaccnt1';match;;3.2.1.4.1.4, 3.2.1.6.5, 3.2.16.2.4;
IA-2.G.3.6;Re-extract IAaccnt1 status information;extract_account_profile IAaccnt1 "/tmp/IA-2.IAaccnt1";complete;;;
IA-2.G.3.7;Verify user account is no longer locked;test_parameter_count "PS" 1 /tmp/IA-2.IAaccnt1;match;;3.2.1.4.1.4,
3.2.1.6.5, 3.2.16.2.4;
IA-2.G.15.1;Verify /h/COE/Comp/APM/bin/APM_Assign_Passwords exists;file_exists
/h/COE/Comp/APM/bin/APM_Assign_Passwords;found;;3.2.1.4.1.1.1;
IA-2.G.15.2;Verify protections on /h/COE/Comp/APM/bin/APM_Assign_Passwords;test_protection_minimum -rwxr-x---
/h/COE/Comp/APM/bin/APM_Assign_Passwords;match;;3.2.5.6;
IA-2.G.15.3;Verify owner on /h/COE/Comp/APM/bin/APM_Assign_Passwords;test_owner "COE|root"
/h/COE/Comp/APM/bin/APM_Assign_Passwords;match;;3.2.5.3, 3.2.5.6;
IA-2.G.15.4;Verify group on /h/COE/Comp/APM/bin/APM_Assign_Passwords;test_group "admin"
/h/COE/Comp/APM/bin/APM_Assign_Passwords;match;;3.2.5.3, 3.2.5.6;
IA-2.G.16.1;Verify protections on /h/COE/Comp/APM/bin/APM_EditConfig;test_protection_minimum -rwxr-x---
/h/COE/Comp/APM/bin/APM_EditConfig;match;;3.2.5.6;
IA-2.G.16.2;Verify ownership on /h/COE/Comp/APM/bin/APM_EditConfig;test_owner "COE|root"
/h/COE/Comp/APM/bin/APM_EditConfig;match;;3.2.5.3, 3.2.5.6;
IA-2.G.16.3;Verify group on /h/COE/Comp/APM/bin/APM_EditConfig;test_group "admin"
/h/COE/Comp/APM/bin/APM_EditConfig;match;;3.2.5.3, 3.2.5.6;
#
IA-2.H.5.1;Verify user may be forced to change their password on login;run_command '/usr/bin/passwd -f
IAaccnt1';match;;3.2.1.4.1.5;
IA-2.H.5.2;Re-extract IAaccnt1 status information;extract_account_profile IAaccnt1 "/tmp/IA-2.IAaccnt1";complete;;;
IA-2.H.5.3;Verify user password was expired (set to the epoch date);test_parameter_count "(LK|PS) 010170" 1 /tmp/IA-
2.IAaccnt1;match;;3.2.1.4.1.5;
#
IA-2.M.6.1;Verify minimum password size has been defined;test_parameter "^PASSLENGTH=" /etc/default/passwd;found;;3.2.1.4.1.7,
3.2.1.4.1.7.1;
IA-2.M.6.2;Verify minimum password size has been defined to 8 characters;test_parameter "^PASSLENGTH=8"
/etc/default/passwd;found;;3.2.1.4.1.7, 3.2.1.4.1.7.1, 3.2.1.4.1.7.1.1;
#
#
SECTION=Sol-IA-3: Evaluate consecutive failed login attempts.;
COMMENT=Partial: APM interface and interactive account lock-out testing must be performed manually.;
# ================================================================================================================================
IA-3.P.1.1;Verify /var/adm/loginlog sufficiently protected;test_protection_minimum -rw-------
/var/adm/loginlog;match;;3.2.1.3, 3.2.1.6, 3.2.1.6.3, 3.2.5.6, 3.2.3.3.1;
IA-3.P.1.2;Verify /var/adm/loginlog properly owned;test_owner root /var/adm/loginlog;match;;3.2.5.3, 3.2.5.6;
IA-3.P.1.3;Verify /var/adm/loginlog properly group protected;test_group sys /var/adm/loginlog;match;;3.2.5.3, 3.2.5.6;
IA-3.P.3.1;Verify /etc/default/login sufficiently protected;test_protection_minimum -rw-r--r--
/etc/default/login;match;;3.2.1.6, 3.2.1.6.3, 3.2.5.6;
IA-3.P.3.2;Verify /etc/default/login properly owned;test_owner root /etc/default/login;match;;3.2.5.3, 3.2.5.6;
IA-3.P.3.3;Verify /etc/default/login properly group protected;test_group sys /etc/default/login;match;;3.2.5.3, 3.2.5.6;
IA-3.P.3.4.1;Verify max strikes (retries) is defined;test_parameter "^RETRIES=" /etc/default/login;found;;3.2.1.6.2;
IA-3.P.3.4.2;Verify max strikes (retries) is defined to COE default of 3;test_parameter "^RETRIES=3"
/etc/default/login;found;;3.2.1.6.2.1;
IA-3.P.3.5;Verify syslog on failed login is defined;test_parameter "^SYSLOG=YES" /etc/default/login;found;;3.2.3.3.1;
IA-3.P.3.6.1;Verify wait before displaying banner time is defined;test_parameter "^SLEEPTIME="
/etc/default/login;found;;3.2.1.6, STIG-3.1.3;
IA-3.P.3.6.2;Verify wait before displaying banner time is defined to COE default of 5;test_parameter "^SLEEPTIME=5"
/etc/default/login;found;;3.2.1.6, STIG-3.1.3;
IA-3.P.3.7.1;Verify failed login message record threshold is defined;test_parameter "^SYSLOG_FAILED_LOGINS="
/etc/default/login;found;;3.2.3.3.1, 3.2.1.3;
IA-3.P.3.7.2;Verify failed login message record threshold is defined to COE default of 0;test_parameter
"^SYSLOG_FAILED_LOGINS=0" /etc/default/login;found;;3.2.3.3.1, 3.2.1.3;
IA-3.P.29.1;Verify /h/COE/Comp/PSM/bin/PSM_enable sufficiently protected;test_protection_minimum -rwsr-x---
/h/COE/Comp/PSM/bin/PSM_enable;match;;3.2.1.6.3, 3.2.5.6;
IA-3.P.29.2;Verify /h/COE/Comp/PSM/bin/PSM_enable properly owned;test_owner root
/h/COE/Comp/PSM/bin/PSM_enable;match;;3.2.5.3, 3.2.5.6;
IA-3.P.29.1;Verify /h/COE/Comp/PSM/bin/PSM_enable properly group protected;test_group admin
/h/COE/Comp/PSM/bin/PSM_enable;match;;3.2.5.3, 3.2.5.6;
IA-3.P.29.1;Verify /usr/bin/passwd executables are sufficiently protected;test_protection_minimum -r-sr-sr-x
'/usr/bin/passwd';match;;3.2.1.6, 3.2.1.6.5, 3.2.5.6;
IA-3.P.29.2;Verify /usr/bin/passwd executables are properly owned;test_owner root '/usr/bin/passwd';match;;3.2.5.3, 3.2.5.6;
IA-3.P.29.3;Verify /usr/bin/passwd executables are properly group protected;test_group sys '/usr/bin/passwd';match;;3.2.5.3,
3.2.5.6;
#
#
SECTION=Sol-IA-4: User account management by a trusted user.;
COMMENT=Partial: APM interface testing of account creation/modification/deletion must be performed manually.;
# ================================================================================================================================
IA-4.Q.1.1;Verify /usr/bin/passwd exists;file_exists /usr/bin/passwd;found;;3.2.16.2, 3.2.16.2.4;
```

Figure C-1. Sample HOSTS Input Test File

```
IA-4.Q.1.2;Verify /h/COE/Comp/PSM/bin/PSM_unlock exists;file_exists /h/COE/Comp/PSM/bin/PSM_unlock;found;;3.2.16.2,
3.2.16.2.4;
IA-4.Q.1.3;Verify /usr/bin/admintool exists;file_exists /usr/bin/admintool;found;;3.2.16.1, 3.2.16.1.2, 3.2.16.2, 3.2.16.1.3,
3.2.16.1.5, 3.2.16.2.1, 3.2.16.2.2, 3.2.16.2.3, 3.2.16.2.3.2, 3.2.16.10;
IA-4.Q.1.3;Verify /usr/ucb/vipw exists;file_exists /usr/ucb/vipw;found;;3.2.16.2, 3.2.16.2.1, 3.2.16.2.2, 3.2.16.2.3,
3.2.16.2.3.2;
IA-4.Q.1.4;Verify /usr/sbin/useradd exists;file_exists /usr/sbin/useradd;found;;3.2.16.2, 3.2.16.2.1;
IA-4.Q.1.5;Verify /usr/sbin/userdel exists;file_exists /usr/sbin/userdel;found;;3.2.16.2, 3.2.16.2.2;
IA-4.Q.1.6;Verify /usr/sbin/usermod exists;file_exists /usr/sbin/usermod;found;;3.2.16.1, 3.2.16.1.2, 3.2.16.1.3, 3.2.16.1.5,
3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.2, 3.2.16.1.3, 3.2.16.1.4, 3.2.5.8, 3.2.5.9, 3.2.5.9.1, 3.2.4.3, 3.2.5.2, 3.2.5.4, 3.2.5.7,
3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.1.7;Verify /usr/sbin/groupadd exists;file_exists /usr/sbin/groupadd;found;;3.2.16.1, 3.2.16.1.2, 3.2.16.1.3,
3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.1, 3.2.16.1.4, 3.2.16.1.4.1, 3.2.5.8, 3.2.4.3, 3.2.5.2, 3.2.5.4, 3.2.5.7,
3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.1.8;Verify /usr/sbin/groupdel exists;file_exists /usr/sbin/groupdel;found;;3.2.16.1, 3.2.16.1.2, 3.2.16.1.3,
3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.3, 3.2.16.1.4, 3.2.16.1.4.2, 3.2.16.1.5, 3.2.5.8, 3.2.4.3, 3.2.5.2, 3.2.5.4,
3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.1.9;Verify /usr/sbin/groupmod exists;file_exists /usr/sbin/groupmod;found;;3.2.16.1, 3.2.16.1.2, 3.2.16.1.3,
3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.2, 3.2.16.1.4, 3.2.16.1.4.3, 3.2.5.8, 3.2.5.9, 3.2.5.9.1, 3.2.4.3, 3.2.5.2,
3.2.5.4, 3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.1.10;Verify /usr/sbin/user* executables are sufficiently protected;test_protection_minimum -r-xr-xr-x
'/usr/sbin/user*';match;;3.2.5.6;
IA-4.Q.1.11;Verify /usr/sbin/user* executables are properly owned;test_owner root '/usr/sbin/user*';match;;3.2.5.3, 3.2.5.6;
IA-4.Q.1.12;Verify /usr/sbin/user* executables are properly group protected;test_group sys '/usr/sbin/user*';match;;3.2.5.3,
3.2.5.6;
IA-4.Q.1.13;Verify /usr/sbin/group* executables are sufficiently protected;test_protection_minimum -r-xr-xr-x
'/usr/sbin/group*';match;;3.2.5.6;
IA-4.Q.1.14;Verify /usr/sbin/group* executables are properly owned;test_owner root '/usr/sbin/group*';match;;3.2.5.3, 3.2.5.6;
IA-4.Q.1.15;Verify /usr/sbin/group* executables are properly group protected;test_group sys '/usr/sbin/group*';match;;3.2.5.3,
3.2.5.6;
IA-4.Q.1.16;Verify /usr/bin/admintool executable is sufficiently protected;test_protection_minimum -r-s--x--x
'/usr/bin/admintool';match;;3.2.5.6;
IA-4.Q.1.17;Verify /usr/bin/admintool executable is properly owned;test_owner root '/usr/bin/admintool';match;;3.2.5.3,
3.2.5.6;
IA-4.Q.1.18;Verify /usr/bin/admintool executable is properly group protected;test_group sys
'/usr/bin/admintool';match;;3.2.5.3, 3.2.5.6;
IA-4.Q.1.19;Verify /usr/ucb/vipw executable is sufficiently protected;test_protection_minimum -r-xr-xr-x
'/usr/ucb/vipw';match;;3.2.5.6;
IA-4.Q.1.20;Verify /usr/ucb/vipw executable is properly owned;test_owner root '/usr/ucb/vipw';match;;3.2.5.3, 3.2.5.6;
IA-4.Q.1.21;Verify /usr/ucb/vipw executable is properly group protected;test_group bin '/usr/ucb/vipw';match;;3.2.5.3,
3.2.5.6;
#
IA-4.Q.1.22;Verify /usr/bin/chgrp exists;file_exists /usr/bin/chgrp;found;;3.2.16.1, 3.2.16.1.3, 3.2.16.2.3.2, 3.2.5.4,
3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.1.5;
IA-4.Q.1.23;Verify /usr/bin/chgrp executable is sufficiently protected;test_protection_minimum -r-xr-xr-x
'/usr/bin/chgrp';match;;3.2.5.6;
IA-4.Q.1.24;Verify /usr/bin/chgrp executable is properly owned;test_owner root '/usr/bin/chgrp';match;;3.2.5.3, 3.2.5.6;
IA-4.Q.1.25;Verify /usr/bin/chgrp executable is properly group protected;test_group bin '/usr/bin/chgrp';match;;3.2.5.3,
3.2.5.6;
#
IA-4.Q.6;Create an account with unique name;create_accounts "IAaccnt4" "1237" "admin" "/bin/csh"
"/h/USERS/local/IAaccnt4";created;;3.2.5.1, 3.2.5.2, 3.2.5.7, 3.2.16.2.1;
IA-4.Q.7.1;Verify the new account has been added to the password file;test_parameter_count "^IAaccnt4" 1
/etc/passwd;match;;3.2.16.2.1, 3.2.1.1.1;
IA-4.Q.7.2;Verify the new account has been added to the shadow file;test_parameter_count "^IAaccnt4" 1
/etc/shadow;match;;3.2.16.2.1, 3.2.1.1.1;
IA-4.Q.7.3;Add new user to another group;run_command '/usr/sbin/usermod -G staff IAaccnt4';match;;3.2.16.1, 3.2.16.1.2,
3.2.16.1.3, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.2, 3.2.16.1.4, 3.2.16.1.4.3, 3.2.5.8, 3.2.5.9, 3.2.5.9.1, 3.2.4.3, 3.2.5.2,
3.2.5.4, 3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.7.4;Verify the new account has been added into the group file;test_parameter_count "IAaccnt4" 1
/etc/group;match;;3.2.16.1, 3.2.16.1.2, 3.2.16.1.3, 3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.2, 3.2.16.1.4, 3.2.16.1.4.3,
3.2.5.8, 3.2.4.3, 3.2.5.2, 3.2.5.4, 3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.7.5;Verify /h/USERS/local/IAaccnt4 exists;file_exists /h/USERS/local/IAaccnt4;found;;3.2.15.2, 3.2.5.3, 3.2.5.6;
IA-4.Q.11;Remove created IAaccnt4 account;delete_accounts "IAaccnt4";removed;;3.2.16.2.2;
IA-4.Q.14;Verify the new account has been removed from the password file;test_parameter_count "^IAaccnt4" 0
/etc/passwd;match;;3.2.16.2.2;
IA-4.Q.15;Verify the new account has been removed from the shadow file;test_parameter_count "^IAaccnt4" 0
/etc/shadow;match;;3.2.16.2.2;
IA-4.Q.16;Verify the new account has been removed from the group file;test_parameter_count "IAaccnt4" 0
/etc/group;match;;3.2.16.2.2, 3.2.16.2.3, 3.2.16.1.4;
IA-4.Q.17;Verify /h/USERS/local/IAaccnt4 no longer exists;file_exists /h/USERS/local/IAaccnt4;not found;;3.2.16.2.2,
3.2.16.2.3, 3.2.16.1.4;
IA-4.Q.28.1;Add new group;test_status '/usr/sbin/groupadd -g 1234 grouptst && echo done' 'done';match;;3.2.16.1, 3.2.16.1.2,
3.2.16.1.3, 3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.1, 3.2.16.1.4, 3.2.16.1.4.1, 3.2.5.8, 3.2.4.3, 3.2.5.2, 3.2.5.4,
3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.28.2;Verify the new group has been added to the group file;test_parameter_count "grouptst" 1
/etc/group;match;;3.2.16.1, 3.2.16.1.2, 3.2.16.1.3, 3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.1, 3.2.16.1.4, 3.2.16.1.4.1,
3.2.5.8, 3.2.4.3, 3.2.5.2, 3.2.5.4, 3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.38.1;Modify group name;run_command '/usr/sbin/groupmod -ngrptest grouptst && echo done' 'done';match;;3.2.16.1,
3.2.16.1.2, 3.2.16.1.3, 3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.2, 3.2.16.1.4, 3.2.16.1.4.3, 3.2.5.8, 3.2.4.3, 3.2.5.2,
3.2.5.4, 3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.38.2;Verify the new group has been modified in the group file;test_parameter_count "grptest" 1
/etc/group;match;;3.2.16.1, 3.2.16.1.2, 3.2.16.1.3, 3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.2, 3.2.16.1.4, 3.2.16.1.4.3,
3.2.5.8, 3.2.4.3, 3.2.5.2, 3.2.5.4, 3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.47;Remove new group;test_status '/usr/sbin/groupdel grptest && echo done' 'done';match;;3.2.16.1, 3.2.16.1.2,
```

Figure C-1. Sample HOSTS Input Test File

```
3.2.16.1.3, 3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.3, 3.2.16.1.4, 3.2.16.1.4.2, 3.2.16.1.5, 3.2.5.8, 3.2.4.3, 3.2.5.2,
3.2.5.4, 3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.49;Verify the new group has been removed from the group file;test_parameter_count "grptest" 0
/etc/group;match;;3.2.16.1, 3.2.16.1.2, 3.2.16.1.3, 3.2.16.1.5, 3.2.16.2, 3.2.16.2.3, 3.2.16.2.3.3, 3.2.16.1.4, 3.2.16.1.4.2,
3.2.16.1.5, 3.2.5.8, 3.2.4.3, 3.2.5.2, 3.2.5.4, 3.2.5.7, 3.2.5.11, 3.2.5.11.1, 3.2.16.10;
IA-4.Q.53.1;Verify user account may be locked;run_command '/usr/bin/passwd -l IAaccnt2';match;;3.2.16.2.4;
IA-4.Q.53.2;Extract IAaccnt2 status information;extract_account_profile IAaccnt2 "/tmp/IA-2.IAaccnt2";complete;;;
IA-4.Q.53.3;Verify user account is locked;test_parameter_count "LK" 1 /tmp/IA-2.IAaccnt2;match;;3.2.16.2.4;
COMMENT=OPERATOR ACTION:;
COMMENT=Please enter a password, press return, enter the password again and press return a second time. Nothing will be
displayed on the screen.;
IA-4.Q.58.1;Set user account password to known value thereby unlocking the account;run_command '/usr/bin/passwd
IAaccnt2';match;;3.2.16.2.4;
IA-4.Q.58.2;Extract IAaccnt2 status information;extract_account_profile IAaccnt2 "/tmp/IA-2.IAaccnt2";complete;;;
IA-4.Q.58.3;Verify user account is locked;test_parameter_count "PS" 1 /tmp/IA-2.IAaccnt2;match;;3.2.16.2.4;
IA-4.Q.58.4;Lock account with COE tool;run_command '/h/COE/Comp/PSM/bin/PSM_unlock -l IAaccnt2';match;;3.2.16.2.4;
IA-4.Q.58.5;Extract IAaccnt2 status information;extract_account_profile IAaccnt2 "/tmp/IA-2.IAaccnt2";complete;;;
IA-4.Q.58.6;Verify user account is locked;test_parameter_count "LK" 1 /tmp/IA-2.IAaccnt2;match;;3.2.16.2.4;
IA-4.Q.58.7;Unlock account with COE tool;run_command '/h/COE/Comp/PSM/bin/PSM_unlock IAaccnt2';match;;3.2.16.2.4;
IA-4.Q.58.8;Extract IAaccnt2 status information;extract_account_profile IAaccnt2 "/tmp/IA-2.IAaccnt2";complete;;;
IA-4.Q.58.9;Verify user account is locked;test_parameter_count "LK" 0 /tmp/IA-2.IAaccnt2;match;;3.2.16.2.4;
#
IA-4.Q.58.10;Verify secman account is defined in /etc/passwd;test_parameter "^secman:x:101:36:" /etc/passwd;found;;3.2.16.1.1;
IA-4.Q.58.11;Verify sysadmin account is defined in /etc/passwd;test_parameter "^sysadmin:x:100:1:"
/etc/passwd;found;;3.2.16.1.1;
#
#
SECTION=Test Cleanup;
# Remove created user accounts
# ================================================================================================================================
IA-Cleanup.1;Remove created first account;delete_accounts "IAaccnt1";removed;;;
IA-Cleanup.2;Remove created second account;delete_accounts "IAaccnt2";removed;;;
IA-Cleanup.3;Remove created test files;run_command_as_user_remove root LOGINENV "/tmp/IA-2.np_passwd /tmp/IA-2.IAaccnt1
/tmp/IA-2.IAaccnt2 /tmp/IA-2.lock_check";match;;;
```

Appendix D
Sample HOSTS Plugin File

This appendix provides a example of a HOSTS plugin. Figure D-1 contains the Perl code for the `test_parameter` plugin. This plugin uses the Unix `egrep` command to search one or more files for a string (e.g., any allowed regular expression). If the string is found within the examined file(s), a match is declared.

Both the string to be matched and the list of files to be searched are passed as input parameters to the plugin. The two return values, `found` and `not found`, would be used as the pass/fail criteria for the test entry within the input test file. (The number of return values does not have to be limited to two, but pass/fail must be based on the return value matching one of the possible expected value.)

As an example of using the `found` return value as the pass criterion, one might test for the existence of `localhost` within the `/etc/hosts` file. This plugin would be activated using:

```
test_parameter  "localhost"  "/etc/hosts"
```

The corresponding entry within the input test file would be:

```
SomeTest.001;Verify localhost is defined in /etc/hosts;test_parameter "localhost"
    "/etc/hosts";found;5.4.3.2.1;3.2.1;
```

Here, HOSTS would evaluate the test and compare the return value to the expected value (e.g., `found`). If they match, a pass is declared. Line wrapping has been added for readability.

As an example of using the `not found` return value as the pass criteria, one might test for the existence of commonly used guest accounts within `/etc/passwd` and `/etc/shadow`. This plugin would be activated using:

```
test_parameter  "^(guest|temp|visitor)"  "/etc/password /etc/shadow"
```

The corresponding entry within the input test file would be:

```
SomeTest.002;Verify no guest accounts have been defined;test_parameter
    "(guest|temp|visitor)" "/etc/passwd /etc/shadow";not found;5.4.3.2.1;3.2.1;
```

In this case, HOSTS would again evaluate the test and compare the return value to the expected value (e.g., `not found`). If they match, a pass is declared. Again, line wrapping has been added for readability.

Figure D-1. Sample HOSTS Plugin File

```
#!/usr/bin/perl -w
#
#================================================================================
#================================================================================
#
# Name:         test_parameter
#
#               Host-Oriented Security Test Suite (hosts) Plugin
#
# Description:  This script tests to see if a specified parameter is defined
#               within a specified file.  The parameter is passed as a regular
#               expression.
#
# History:      REV      DATE      PROGRAMMER   DESCRIPTION
#               ===   =========   ============  ==============================
#               1.0   15 Jun 01   Finegan       Original
#
# Usage:        Call:   test_parameter <parameter> <file>
#
#               Where:  <parameter>     The parameter being tested for as a
#                                       regular expression.
#                       <file>          Fully qualified file name within
#                                       which the parameter may/may not be
#                                       found.
#
#               Return: found           Specified parameter was found within
#                                       the examined file.
#                       not found       Specified parameter was not found
#                                       within the examined file.
#
# Corporation:  The MITRE Corporation, 1820 Dolley Madison, McLean, VA 22102
#
#================================================================================
#================================================================================
#
#
# Define system commands used by this plugin
# ============================================================================
$grep = "/bin/egrep";
#
#
# Extract parameter expression and candidate files from the arguments
# ============================================================================
($param_expression, @candidate_files) = @ARGV;
#
#
# Define how the grep command will be used
# ============================================================================
$grep_for_parameter = join(" ","$grep", "\"$param_expression\"",
   @candidate_files);
#
#
# Run system command recording results
# ============================================================================
open (OS_COMMAND,"$grep_for_parameter | ");
   @command_records = <OS_COMMAND>;
```

Figure D-1. Sample HOSTS Plugin File

```
close (OS_COMMAND);
$lines_returned = @command_records;
#
#
# If parameter expression found, return found indication
# ============================================================================
if ( $lines_returned > 0 ) {
   print "found\n@command_records";
}
#
#
# Else return not found indication
# ============================================================================
else {
   print "not found\n";
}
#

#
# Exit
# ============================================================================
exit
```

Appendix E
Available HOSTS Plugin Files

This appendix provides the following information for each of the HOSTS plugin files contained in this distribution:

1. Plugin file name. Included is a description of the plugin's function.
2. Description of the calling sequence (e.g., the application programming language [API]) and the expected return values.

The detailed information is contained within Table E-1.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`active_daemon`
 	
Tests to see if a specified daemon is active. The module filters out other sessions that may contain the same daemon test string (e.g., vi, grep).
	`active_daemon <daemon_name>`
	
	`Where:`
	
	`<daemon_name>  The name of daemon being examined.`
	
	`Return Values:`
	
	`active  Daemon found to be active.`
	`Inactive  Daemon not found to be active.`

`active_port`
 	
Determines if specified ports are active. The port may be specified either by its `/etc/services` name or as an integer.
	`active_port [<port>]`
	
	`Where:`
	
	`<port>  The port number to be tested. If no argument is passed, this module will return the total number of ports in idle mode, listen mode or established mode.`
	
	`Return Values:`
	
	`active  The specified port was found to be active.`
	`inactive  The specified port was not found to be active.`
	`unexpected  A flag other than zero was returned. Typically a sign of an error.`

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`create_accounts`
 	
Exercises the account creation capabilities.
	`create_accounts <username> <uid> <gid> <shell> <home>`
	
	`Where:`
	
	`<username>` Username of the account to be created.
	`<uid>` UID for account to be created.
	`<gid>` GID for account to be created.
	`<shell>` Login shell for account to be created.
	`<home>` Home directory for account to be created.
	
	`Return Values:`
	
	`created` Account has been created.
	`not created` Account has not been created.
	`unexpected` Unable to determine if account has been successfully created.

`delete_accounts`
 	
Tests the account deletion capabilities.
	`delete_accounts <username>`
	
	`Where:`
	
	`<username>` Username of the account to be deleted.
	
	`Return Values:`
	
	`removed` Account has been removed.
	`not removed` Account has not been removed.
	`unexpected` Unable to determine if account has been successfully removed.

`empty_file`
 	
Tests to see if a specified file is an empty file (e.g., has a length of 0).
	`empty_file <file>`
	
	`Where:`
	
	`<file>` Fully qualified file name for which contents are to be checked.
	
	`Return Values:`
	
	`match` The specified file was found and it is empty.
	`no match` The specified file was found and it was not empty.
	`not found` Candidate file was not found.

`extract_account_profile`
 	
Extracts a profile of the specified user account. The profile includes account attributes (e.g., UID, GID), and password status information.
	`extract_account_profile <account> <output>`
	
	`Where:`
	
	`<account>` The name of the account to have profile information extracted.
	`<output>` The fully qualified file name into which the acco8unt profile information will be extracted.
	
	`Return Values:`
	
	`complete` The specified account was found and data was successfully extracted.
	`unexpected` The specified account could not be found or information on the specified account could not be extracted.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`file_exists`
 	
Tests to see if a specified file exists on a system.
	`file_exists <file>`
	
	`Where:`
	
	`<file>` Fully qualified file name for which existence is to be checked.
	
	`Return Values:`
	
	`found` The specified file was found.
	`not found` Candidate file was not found.

`files_differ`
 	
Tests to see if two specified files differ in content from each other. Differences found will be returned.
	`files_differ <file 1> <file 2>`
	
	`Where:`
	
	`<file 1>` Fully qualified file name for first file to be compared.
	`<file 2>` Fully qualified file name for second file to be compared.
	
	`Return Values:`
	
	`match` The contents of both files match.
	`no match` The contents of both files do not match.
	`not found` Candidate file(s) not found.

`files_differ_field`
 	
This script tests to see if two specified files differ (content wise) from each other comparing only a specific field. Differences found will be returned.
	`files_differ_field <field> <file 1> <file 2>`
	
	`Where:`
	
	`<column>` Starting column number to be compared (numeric format)
	`<file 1>` Fully qualified file name for first file to be compared.
	`<file 2>` fully qualified file name for second file to be compared.
	
	`Return Values:`
	
	`match` The contents of both files match.
	`no match` The contents of both files do not match.
	`not found` Candidate file(s) not found.

`find_all_c_shell_scripts`
 	
Tests to see if C shell scripts exist under the specified starting point. It differs from `find_c_shell_scripts` in that it will attempt to examine the contents of files normally tagged as text files to determine whether or not these files may actually be C shell scripts. It will not scan remotely mounted file systems.
	`find_all_c_shell_scripts <start>`
	
	`Where:`
	
	`<start>` Fully qualified directory name from which the scan is to begin.
	
	`Return Values:`
	
	`<list>` List of one or more C shell scripts that were found.
	`none found` No C shell scripts were found.
	`none found - remote` Unable to perform check, starting point is on a remote file system.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values
find_c_shell_scripts Tests to see if C shell scripts exist under the specified starting point. It will not scan remotely mounted file systems.	find_c_shell_scripts <start> Where: <start> Fully qualified directory name from which the scan is to begin. Return Values: <list> List of one or more C shell scripts that were found. none found No C shell scripts were found. none found - remote Unable to perform check, starting point is on a remote file system.
find_file_and_test Parses down local partitions looking for files matching the specified pattern string. If any are found, the secondary test is applied to the detected file(s).	find_file_and_test <pattern> <test> <result> <start> Where: <pattern> ASCII string pattern that is to be matched (e.g., all files with a name that contains string). <test> Secondary test that is to be applied. <result> Expected test result for the secondary test. <start> Fully qualified directory name from which the scan is to begin. Return Values: match Both expected values equal the returned values. match - partial One or more of the tested files did match the test string, but one or more did not. no match The test returned a value that differs from its expected value. no match - remote Unable to perform check, starting point is on a remote file system unexpected Unexpected value returned from issued command(s).
find_misgrouped_files Tests to see if files exist in the specified directories that are not owned buy a GID found within the specified GID range. This is typically used to find files belonging to a non-root group within root directories.	find_misgrouped_files <GID low> <GID high> <start> Where: <GID low> The lowest GID acceptable for files within the scanned directories <GID high> The highest GID acceptable for files within the scanned directories <start> Fully qualified directory name from which the scan is to begin. Return Values: <list> List of one or more misgrouped files or directories found none found No misgrouped files or directories were found none found - remote Unable to perform check, starting point is on a remote file system

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`find_misowned_files`
 	
Tests to see if files exist in the specified directories that are not owned buy a UID found within the specified UID range. This is typically used to find files owned by a non-root account within root directories.
	`find_misowned_files <UID low> <UID high> <start>`
	
	`Where:`
	
	`<UID low>` The lowest UID acceptable for files within the scanned directories
	`<UID high>` The highest UID acceptable for files within the scanned directories
	`<start>` Fully qualified directory name from which the scan is to begin.
	
	`Return Values:`
	
	`<list>` List of one or more misowned files or directories found
	`none found` No misowned files or directories were found
	`none found - remote` Unable to perform check, starting point is on a remote file system

`find_misprotected_files`
 	
Tests to see if files exist in the specified directories that have protections granting group or world greater access than owner or world greater access than group. This is sometimes referred to as uneven protections.
	`find_misprotected_files <start>`
	
	`Where:`
	
	`<start>` Fully qualified directory name from which the scan is to begin.
	
	`Return Values:`
	
	`<list>` List of one or more misprotected files or directories found
	`none found` No misprotected files or directories were found
	`none found - remote` Unable to perform check, starting point is on a remote file system

`find_privileged_files`
 	
Tests to see if privileged files and/or directories exist under the specified starting point. It will not scan remotely mounted file systems.
	`find_privileged_files <start>`
	
	`Where:`
	
	`<start>` Fully qualified directory name from which the scan is to begin.
	
	`Return Values:`
	
	`<list>` List of one or more privileged files or directories that were found.
	`none found` No privileged files or directories were found.
	`none found - remote` Unable to perform check, starting point is on a remote file system.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`find_privileged_scripts`
 	
Tests to see if privileged shell scripts exist under the specified starting point. It will not scan remotely mounted file systems.
	`find_privileged_scripts <start>`
	
	`Where:`
	
	`<start>` Fully qualified directory name from which the scan is to begin.
	
	`Return Values:`
	
	`<list>` List of one or more privileged shell scripts that were found.
	`none found` No privileged shell scripts were found.
	`none found - remote` Unable to perform check, starting point is on a remote file system.

`find_suspicious_file_names`
 	
Tests to see if files and/or directories exist with strange names under the specified starting point. It will not scan remotely mounted file systems. Examples of strange names include names containing spaces, the plus symbol and control charact ers.
	`find_suspicious_file_names <start>`
	
	`Where:`
	
	`<start>` Fully qualified directory name from which the scan is to begin.
	
	`Return Values:`
	
	`<list>` List of one or more files or directories that were found.
	`none found` No files or directories were found.
	`none found - remote` Unable to perform check, starting point is on a remote file system.

`find_unowned_files`
 	
Tests to see if unowned files and/or directories exist under the specified starting point. It will not scan remotely mounted file systems.
	`find_unowned_files <start>`
	
	`Where:`
	
	`<start>` Fully qualified directory name from which the scan is to begin.
	
	`Return Values:`
	
	`<list>` List of one or more unowned files or directories were found
	`none found` No unowned files or. directories were found.
	`none found - remote` Unable to perform check, starting point is on a remote file system.

`find_world_writables`
 	
Tests to see if world-writable files and/or directories exist under the specified starting point. It will not scan remotely mounted file systems.
	`find_world_writables <start>`
	
	`Where:`
	
	`<start>` Fully qualified directory name from which the scan is to begin.
	
	`Return Values:`
	
	`<list>` List of one or more world-writable files or directories that were found
	`none found` No world-writable files or directories were found
	`none found - remote` Unable to perform check, starting point is on a remote file system.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`modify_accounts`
 	
Tests the account modification capabilities.
	`modify_accounts <username> <uid> <gid> <shell> <home>`
	
	`Where:`
	
	`<username>` Username of the account to be modified.
	`<uid>` UID for account to be modified.
	`<gid>` GID for account to be modified.
	`<shell>` Login shell for account to be modified.
	`<home>` Home directory for account to be modified.
	
	`Return Values:`
	
	`modified` Account has been modified
	`not modified` Account has not been modified
	`unexpected` Unable to determine if account has been successfully modified

`multi_part_test`
 	
Runs two specified commands comparing the returned output with the specified expected output. If both results match the expected values, a match is declared.
	`multi_part_test <cmd 1> <expect 1> <cmd 2> <expect 2>`
	
	`Where:`
	
	`<cmd 1>` The first command to be issued for the test.
	`<expect 1>` The expected return value for the 1st test.
	`<cmd 2>` The second command to be issued for the test.
	`<expect 2>` The expected return value for the 2nd test.
	
	`Return Values:`
	
	`match` Both expected values equal the returned values.
	`no match one` The first test returned a value that differs from its expected value.
	`no match two` The second test returned a value that differs from its expected value.
	`no match both` Both of the expected values differ from the returned values.
	`unexpected` Unexpected value returned from issued command(s).

`print_pid`
 	
This Bourne shell script will find and print the process ID for a specified process. The module filters out other processes that may contain the same passed test string (e.g., vi, grep).
	`print_pid <test_string>`
	
	`Where:`
	
	`<test_string>` Test string to examine for. Typically a daemon name (e.g., syslogd).
	
	`Return Values:`
	
	`<pid>` Numeric process ID number.

`run_command`
 	
Executes a command.
	`run_command <command>`
	
	`Where:`
	
	`<command>` The command to be issued for the test.
	
	`Return Values:`
	
	`match` The command executed returning a status flag of zero - no error.
	`unexpected` A flag other than zero was returned.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values
run_command_as_user Uses the su command to assume another user's profile before executing the specified command. This includes the option of assuming the target user's login environment.	run_command_as_user <username> <login> <command> Where: <username> The username (e.g., account) under which the command is to be executed. <login> Flag used to control whether or not the execution shell created, in which the command will be executed, is to be reset to match the login environment belonging to username. Allowed flag values are: LOGINENV Reset environment username's login environment. <else> Do not reset environment to username's login environment. <command> The command to be issued for the test. Return Values: match The command executed returning a status flag of zero - no error. unexpected A flag other than zero was returned.
run_command_as_user_chprot Uses the su command to assume another user's profile before changing the protections on the specified file. This includes the option of assuming the target user's login environment.	run_command_as_user_chprot <username> <login> <protection> <filename> Where: <username> The username (e.g., account) under which the command is to be executed. <login> Flag used to control whether or not the execution shell created, in which the command will be executed, is to be reset to match the login environment belonging to username. Allowed flag values are: LOGINENV Reset environment username's login environment. <else> Do not reset environment to username's login environment. <protection> New protection mask to be assigned to the target file. <filename> The fully qualified pathname for the file to be created. Return Values: match The command executed returning a status flag of zero - no error. unexpected A flag other than zero was returned.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values
run_command_as_user_create Uses the su command to assume another user's profile before creating the specified file. This includes the option of assuming the target user's login environment.	run_command_as_user_create <username> <login> <filename> Where: <username> The username (e.g., account) under which the command is to be executed. <login> Flag used to control whether or not the execution shell created, in which the command will be executed, is to be reset to match the login environment belonging to username. Allowed flag values are: LOGINENV Reset environment username's login environment. <else> Do not reset environment to username's login environment. <filename> The fully qualified pathname for the file to be created. Return Values: match The command executed returning a status flag of zero - no error. unexpected A flag other than zero was returned.
run_command_as_user_redirect Uses the su command to assume another user's profile before executing the specified command. The results of the command are redirected as specified. This includes the option of assuming the target user's login environment.	run_command_as_user_redirect <username> <login> <command> Where: <username> The username (e.g., account) under which the command is to be executed. <login> Flag used to control whether or not the execution shell created, in which the command will be executed, is to be reset to match the login environment belonging to username. Allowed flag values are: LOGINENV Reset environment username's login environment. <else> Do not reset environment to username's login environment. <command> The command to be issued for the test. Return Values: match The command executed returning a status flag of zero - no error. unexpected A flag other than zero was returned.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values
run_command_as_user_remove Uses the su command to assume another user's profile before removing the specified file. This includes the option of assuming the target user's login environment.	run_command_as_user_remove <username> <login> <filename> Where: <username> The username (e.g., account) under which the command is to be executed. <login> Flag used to control whether or not the execution shell created, in which the command will be executed, is to be reset to match the login environment belonging to username. Allowed flag values are: LOGINENV Reset environment username's login environment. <else> Do not reset environment to username's login environment. <filename> The fully qualified pathname for the file to be removed. Return Values: match The command executed returning a status flag of zero - no error. unexpected A flag other than zero was returned.
run_command_as_user_write Uses the su command to assume another user's profile before writing data into the specified file. This includes the option of assuming the target user's login environment.	run_command_as_user_write <username> <login> <string> <filename> Where: <username> The username (e.g., account) under which the command is to be executed. <login> Flag used to control whether or not the execution shell created, in which the command will be executed, is to be reset to match the login environment belonging to username. Allowed flag values are: LOGINENV Reset environment username's login environment. <else> Do not reset environment to username's login environment. <string> String if data that is to be written into the target file. <filename> The fully qualified pathname for the file to be created. Return Values: match The command executed returning a status flag of zero - no error. unexpected A flag other than zero was returned.
run_command_redirect Executes a command allowing the results to be redirected.	run_command_redirect <command> Where: <command> The command to be issued for the test. Return Values: match The command executed returning a status flag of zero - no error. unexpected A flag other than zero was returned.

Table E-1. HOSTS Plugin Files	
Name	**Calling Sequence and Return Values**
os_version This script ascertains which OS is installed on the system. If the OS is as specified, the test passes. If the OS is any other version, the test fails. If enabled, the plugin will issue a "die" thereby aborting test execution.	os_version <required OS> <die flag> Where: <required OS> The OS for which the test was written. <die flag> Boolean flag controlling die logic: TRUE - Die on fail – abort test else - Fail test and continue with test. Return Values: PASS The OS version was determined match the passed requirement. FAIL The OS version differed.
startup_file_exists Tests to see boot time startup files exist with a name that contains the candidate string. It is used to verify candidate startup files that are not run at boot time.	startup_file_exists <pattern> Where: <pattern> Unique pattern from within the name of the candidate startup file. Return Values: found Start-up files with names containing the candidate string were found. This implies the startup routine will run at boot time. not found Start-up files with names containing the candidate string were not found.
test_IFS This Bourne shell script will test IFS behavior under the specified shell. The IFS is the internal field separator definition (e.g., the semicolon normally).	test_IFS <shell> Where: <shell> The name of the shell to be used for performing the test (e.g., sh, ksh). Return Values: WARNING! Security vulnerability detected where the shell being examined does not reset the IFS variable. IFS Safe The shell being examined does reset the IFS variable.
test_dormant_accounts This Bourne shell script will find and print the names of accounts that have either been idle for a period of time in excess of one month or have never been used.	test_dormant_accounts Return Values: CLEAN No inactive accounts were detected. WARNING One or more inactive user accounts were detected. A list of inactive accounts will then be displayed.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values
test_for_checksums Computes a file's 16-bit and cyclic redundancy check (CRC) checksums. These values are then compared against argument passed values.	test_for_checksums <cksum> <sum> <file> Where: <cksum> Expected cyclic redundancy check (CRC) value for file being tested. <sum> Expected 16 bit checksum for the file being tested. <file> Fully qualified file name for the file being checked. Return Values: match CRC and checksum matched for candidate file(s). no match CRC or checksum did not match for candidate file(s).
test_for_string Tests a file to see if a specified ASCII string is detected within the specified file. The file may be any file type (e.g., binary or data). Strings of three or more characters are examined.	test_for_string <ascii string> <file> Where: <ascii string> The ASCII string(s) for which the file is being tested. This argument may be any valid regular string expression. <file> Fully qualified file name for the file being checked. Return Values: found String(s) found in candidate file(s). not found String(s) not found in candidate file(s).
test_for_system_calls Tests to see if system calls with known vulnerabilities are contained within candidate binary file(s). To function, the binary file(s) must not be stripped.	test_for_system_calls <system calls> <candidate> Where: <system calls> System call(s) in a regular expression for which the candidate files are tested. <candidate> Fully qualified name for the candidate file(s). Return Values: found Specified system calls were found within candidate file. not found Specified system calls were not found within candidate file. unspecified Candidate file could not be examined for specified system calls.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`test_for_trojans`
 	
Tests a file to see if a specified ASCII string commonly associated with Trojan horse binaries is detected within the specified file.
	`test_for_trojans <ascii string> <matches> <file>`
	
	`Where:`
	
	`<ascii string>` The ASCII string(s) for which the file is being tested. This argument may be any valid regular string expression.
	`<matches>` Number of times the string is expected to be found (e.g., match) within the binary.
	`<file>` Fully qualified file name for the file being checked.
	
	`Return Values:`
	
	`match` The number of times the string(s) were found in candidate file(s) matched the expected number of times.
	`no match` The number of times the string(s) were found in candidate file(s) did not match the expected number of times the string should be found.

`test_gid_values`
 	
Tests the local group file looking for unexpected GID values less than 20. In addition, it ensures that all GID values identified in `/etc/passwd` have an entry in `/etc/group`. GID values not in compliance will be displayed.
	`test_gid_values`
	
	`Return Values:`
	
	`match` All GID values under 20 are expected and all GID values in /etc/passwd are defined in /etc/group.
	`no match` An unexpected result was encountered. Either a GID definition was found that was under 20 and not expected or a GID is used in /etc/passwd that is not defined in /etc/group.

`test_group`
 	
Tests to see if a specified group is defined for a specified file. The group is passed as a string matching the expected value (e.g., `staff`).
	`test_group <group name> <file>`
	
	`Where:`
	
	`<group name>` The expected group associated with the file being tested. The value is an ASCII string.
	`<file>` Fully qualified file name for which the group is to be checked.
	
	`Return Values:`
	
	`match` The group of the file matched the expected value.
	`no match` The group of the file did not match the expected value.
	`partial match` The group of the file matched one or more files, but no all files.
	`not found` Candidate file was not found.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`test_home_dir_files`
 	
Examines the user's home directory verifying the file attributes on the specified file. The minimum protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`).
 	
Logic has also been added to allow for privileged accounts. If an account has a `UID` of less than 10, matches will be declared if the file(s) is owned by root and has a `GID` value of less than 10.
	`test_home_dir_files <protection> <users> <files>`
	
	`Where:`
	
	`<protection>` The minimal protection setting that is acceptable on the examined file(s). The value is an ASCII string.
	`<users>` The list of user accounts to be examined. To examine all user accounts on the system, enter "ALL".
	`<files>` The file(s) to be examined. These may be directories.
	
	`Return Values:`
	
	`match` The attributes for the specified file were found to belong to the user and had a protection equal to or more stringent than the specified protection.
	`no match` The attributes for the specified file were found not properly owned or to have a protection less stringent than the specified protection.
	`not found` Candidate file was not found.

`test_logins_duplicate_uid`
 	
Tests the local password file looking for duplicate UID definitions (e.g., more than one account with the same UID). The number of accounts with duplicate UID values is then compared to an expected value. Accounts with duplicate values are displayed.
	`test_logins_duplicate_uid <expect count>`
	
	`Where:`
	
	`<expect count>` The number of accounts that are expected to have duplicate UID values. For example, if none of the accounts are expected to share UID values, the expected count would be zero.
	
	`Return Values:`
	
	`match` The number of accounts that share a common UID matches the expected number.
	`no match` The number of accounts that share a common UID did not match the expected number.

`test_logins_duplicate_usernames`
 	
Tests the local password file looking for duplicate usernames (e.g., more than one account with the same account username). The number of accounts with duplicate usernames is then compared to an expected value. Accounts with duplicate usernames are displayed.
	`test_logins_duplicate_usernames <expect count>`
	
	`Where:`
	
	`<expect count>` The number of accounts that are expected to have duplicate usernames. For example, if none of the accounts are expected to share usernames, the expected count would be zero.
	
	`Return Values:`
	
	`match` The number of accounts that share a common username matches the expected number.
	`no match` The number of accounts that share a common username did not match the expected number.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`test_logins_maximum_age_limit`
 	
Tests the local shadow file looking for users with maximum password change thresholds greater than the allowable maximum (e.g., accounts not being forced to change their passwords as often as a given project or application stipulates.) The number of accounts with a threshold above the maximum threshold is then compared to an expected value. Accounts with a threshold above the maximum are displayed.
	`test_logins_maximum_age_limit <max age> <expect count>`
	
	`Where:`
	
	`<max age>` The maximum age a password must exceed before a user is forced to change their password. Specified in days.
	`<expect count>` The number of accounts expected to have long password lifetimes. For example, if none of the accounts are expected to have lifetimes greater than the threshold, the expected count would be zero.
	
	`Return Values:`
	
	`match` The number of accounts that have a maximum age threshold above the specified threshold matches the expected number.
	`no match` The number of accounts that have a maximum age threshold above the specified threshold does not match the expected number.

`test_logins_minimum_age_limit`
 	
Tests the local shadow file looking for users with minimum password change threshold less than the allowable minimum (e.g., accounts allowed to change their passwords more often than a given project or application stipulates). The number of accounts with a threshold below the minimum threshold is then compared to an expected value. Accounts with a threshold below the minimum are displayed.
	`test_logins_minimum_age_limit <min age> <expect count>`
	
	`Where:`
	
	`<min age>` The minimum age a password must exceed before a user is permitted to change their password. Specified in days.
	`<expect count>` The number of accounts that are expected to have a minimum password lifetime below the specified threshold. For example, if none of the accounts are expected to have short lifetimes, the expected count would be zero.
	
	`Return Values:`
	
	`match` The number of accounts that have a minimum age threshold below the specified threshold matches the expected number.
	`no match` The number of accounts that have a minimum age threshold below the specified threshold does not match the expected number.

`test_logins_null_password`
 	
Tests the local password file looking for accounts with null passwords. The number of accounts found to have null passwords is then compared to an expected value. Accounts with null passwords are displayed.
	`test_logins_null_password <expect count>`
	
	`Where:`
	
	`<expect count>` The number of accounts that are expected to have null (no) passwords. For example, if none of the accounts are expected to have null passwords, the expected count would be zero.
	
	`Return Values:`
	
	`match` The number of accounts that have null passwords matches the expected number.
	`no match` The number of accounts that have null passwords did not match the expected number.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`test_logins_system_account`
 	
Tests the local password file looking for system accounts (UID < 100). The number of accounts found to be system accounts is then compared to an expected value. Accounts defined as system accounts are displayed.
	`test_logins_system_account <expect count>`
	
	`Where:`
	
	`<expect count>` The number of accounts that are expected to be system accounts. For example, if none of the accounts are expected to be system accounts, the expected count would be zero.
	
	`Return Values:`
	
	`match` The number of accounts that are system accounts matches the expected number.
	`no match` The number of accounts that are system accounts did not match the expected number.

`test_logins_world_writable_in_path`
 	
Tests non-system accounts from the local passwd file to see if any users have a world-writable directory in their search path (PATH) definition. The number of accounts with world-writable directories within their PATH is compared to an expected value. Accounts with a world-writable directory in their PATH definition are displayed.
	`test_logins_world_writable_in_path <expect count>`
	
	`Where:`
	
	`<expect count>` The number of accounts expected to have world-writable directories in their PATH definition. For example, if none of the accounts are expected to have world-writable directories listed in their PATH definition, the expected count would be zero.
	
	`Return Values:`
	
	`match` The number of accounts that have a world-writable directory within their PATH definition matches the expected number.
	`no match` The number of accounts that have a world-writable directory within their PATH definition does not match the expected number.

`test_multi_strings`
 	
Tests to see if multiple ASCII strings, defined by using an input file, are found within a common test file. Each of the test strings is then defined by a regular expression.
	`test_multi_strings <input file> <test file>`
	
	`Where:`
	
	`<input file>` Fully qualified file name within which the strings(s) are defined. Each string is on a separate line defined by a regular expression.
	`<test file>` Fully qualified file name within which the strings(s) may/may not be found.
	
	`Return Values:`
	
	`match` All tested strings were found within the examined file.
	`partial match` Some of the tested strings were found within the examined file. One or more were not.
	`no match` None of the tested strings were found within the examined file.
	`unexpected` Either the input file or the examined file could not be found.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`test_network_attribute`
 	
Tests to see if a specified network attribute for the specified device is set as expected.
	`test_network_attribute <device> <attribute> <expected>`
	
	`Where:`
	
	`<device>` The network device to examine
	`<attribute>` The attribute to examine
	`<expected>` The expected return value.
	
	`Return Values:`
	
	`match` The expected value equals the actual value.
	`no match` The expected value does not equal the actual value.
	`unexpected` Noting returned from issued command

`test_owner`
 	
Tests to see if a specified owner is defined for a specified file. The owner is passed as a string matching the expected value (e.g., `root`).
	`test_owner <owner name> <file>`
	
	`Where:`
	
	`<owner name>` The expected owner associated with the file being tested. The value is an ASCII string.
	`<file>` Fully qualified file name for which the owner is to be checked.
	
	`Return`
	
	`match` The owner of the file matched the expected value.
	`no match` The owner of the file did not match the expected value.
	`partial match` One or more files had an owner other than the specified owner.
	`not found` Candidate file was not found.

`test_parameter`
 	
Tests to see if a specified parameter is defined within a specified file. The parameter is passed as a regular expression.
	`test_parameter <parameter> <file>`
	
	`Where:`
	
	`<parameter>` The parameter being tested for as a regular expression.
	`<file>` Fully qualified file name within which the parameter may/may not be found.
	
	`Return Values:`
	
	`found` Specified parameter was found within the examined file.
	`not found` Specified parameter was not found within the examined file.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`test_parameter_count`
 	
Tests to see how many times a specified parameter occurs within a specified file. The parameter is passed as a regular expression.
 	
This module assumes that the parameter will only occur once on a given line.
	`test_parameter_count <parameter> <expect count> <file>`
	
	`Where:`
	
	`<parameter>` The parameter being tested for as a regular expression.
	`<expect count>` The number of times the parameter is expected within the file(s) being examined. For example, if the parameter should not be set at all, the expected count would be zero.
	`<file>` Fully qualified file name within which the parameter may/may not be found.
	
	`Return Values:`
	
	`match` Specified parameter was found within the examined file the expected number of times.
	`no match` Specified parameter was not found within the examined file the expected number of times.

`test_parameter_filter_count`
 	
Tests to see how many times a specified parameter occurs within a specified file after applying a filter rule. The parameter and rule are passed as regular expressions.
 	
This module assumes that the parameter will only occur once on a given line.
	`test_parameter_filter_count <filter> <parameter> <expect count> <file>`
	
	`Where:`
	
	`<filter>` The filter to be applied before testing for the existence of a parameter. For example, a filter may remove all comment lines.
	`<parameter>` The parameter being tested for as a regular expression.
	`<expect count>` The number of times the parameter is expected within the file(s) being examined. For example, if the parameter should not be set at all, the expected count would be zero.
	`<file>` Fully qualified file name within which the parameter may/may not be found.
	
	`Return Values:`
	
	`match` Specified parameter was found within the examined file the expected number of times.
	`no match` Specified parameter was not found within the examined file the expected number of times.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`test_parameter_nonmatch_count`
 	
Tests to see how many times lines within the specified parameter occur within a specified file. The parameter is passed as a regular expression.
 	
This module assumes that the parameter will only occur once on a given line.
	`test_parameter_nonmatch_count <parameter> <expect count>`
	
	`Where:`
	
	`<parameter>` The parameter being tested for as a regular expression.
	`<expect count>` The number of lines the parameter is not expected to occur on within the file(s) being examined. For example, if all lines within the file should contain the specified parameter, the expected count would be zero.
	`<file>` Fully qualified file name within which the parameter may/may not be found.
	
	`Return Values:`
	
	`match` Specified parameter was not found on lines within the examined file the expected number of times.
	`no match` Specified parameter was not found on lines with within the examined file the expected number of times.

`test_protection`
 	
Tests to see if a specified protection is set for a specified file. The protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`).
	`test_protection <protection> <file>`
	
	`Where:`
	
	`<protection>` The expected protection associated with the file being tested. The value is an ASCII string.
	`<file>` Fully qualified file name for which the protection is to be checked.
	
	`Return Values:`
	
	`match` The protection of the file matched the expected value.
	`no match` The protection of the file did not match the expected value.
	`not found` Candidate file was not found.

`test_protection_d`
 	
Tests to see if a specified protection is set for a specified file. The protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`). Matching records are displayed.
	`test_protection_d <protection> <file>`
	
	`Where:`
	
	`<protection>` The expected protection associated with the file being tested. The value is an ASCII string.
	`<file>` Fully qualified file name for which the protection is to be checked.
	
	`Return Values:`
	
	`match` The protection of the file matched the expected value.
	`no match` The protection of the file did not match the expected value. In addition, a list of non-matching files will be displayed.
	`not found` Candidate file was not found.

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values

`test_protection_minimum`
 	
Tests to see if a specified file has a protection equal to or more stringent than the specified protection. The minimum protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`).
	`test_protection_minimum <protection> <file>`
	
	`Where:`
	
	`<protection>` The minimal expected protection associated with the file being tested. The value is an ASCII string.
	`<file>` Fully qualified file name for which the protection is to be checked.
	
	`Return Values:`
	
	`match` The protection of the file matched the expected value.
	`no match` The protection of the file did not match the expected value.
	`not found` Candidate file was not found.

`test_protection_minimum_d`
 	
Tests to see if a specified file has a protection equal to or more stringent than the specified protection. The minimum protection is passed as a string matching the expected value (e.g., `-rwxr-xr-x`). Matching records are displayed.
	`test_protection_minimum_d <protection> <file>`
	
	`Where:`
	
	`<protection>` The minimal expected protection associated with the file being tested. The value is an ASCII string.
	`<file>` Fully qualified file name for which the protection is to be checked.
	
	`Return Values:`
	
	`match` The protection of the file matched the expected value.
	`no match` The protection of the file did not match the expected value. In addition, a list of non-matching files will be displayed.
	`not found` Candidate file was not found.

`test_status`
 	
Runs a specified command and then compares the returned output with the specified expected output.
	`test_status <command> <expected>`
	
	`Where:`
	
	`<command>` The command to be issued for the test.
	`<expected>` The expected return value.
	
	`Return Values:`
	
	`match` The expected value equals the returned value.
	`no match` The expected value does not equal the returned value.
	`unexpected` Nothing returned from issued command

Table E-1. HOSTS Plugin Files

Name	Calling Sequence and Return Values
test_status_as_user Uses the su command to assume another user's profile before executing the specified command. It then compares the returned information with the expected value before declaring a match or no match. This includes the option of assuming the target user's login environment.	test_status_as_user <username> <login> <command> <expected> Where: <username> The username (e.g., account) under which the command is to be executed. <login> Flag used to control whether or not the execution shell created, in which the command will be executed, is to be reset to match the login environment belonging to username. Allowed flag values are: LOGINENV Reset environment username's login environment. <else> Do not reset environment to username's login environment. <command> The command to be issued for the test. <expected> The expected return value. Return Values: match The expected value equals the returned value. no match The expected value does not equal the returned value. unexpected Nothing returned from issued command
test_string Tests to see if a specified string is defined within a specified file. The string is passed as a regular expression. This module is identical to test_parameter.	test_string <string> <file> Where: <string> The string being tested for as a regular expression. <file> Fully qualified file name within which the string may/may not be found. Return Values: found Specified string was found within the examined file. not found Specified string was not found within the examined file.
test_users_for_file Examines the user's home directory looking for the specified file.	test_users_for_file <file name> Where: <file name> The command to be issued for the test. Return Values: found The specified file was found in one or more user home directories not found The specified file was not found in any user home directory