Chapter 7

Verify Effectiveness of Threat Control Measures

This chapter describes the fourth component of an effective information security/IA program — verifying the effectiveness of threat control measures. The following activities are performed while verifying the effectiveness of threat control measures:

- IA verification techniques are selected and employed.
- Residual risk exposure is determined and its acceptability evaluated.
- Ongoing vulnerabilities, threats, and survivability are monitored.

Outputs from several previous components serve as inputs to these activities.

The majority of contemporary information security books and standards do not mention the topic of verification at all*. This is rather surprising; why should a user, customer, or system owner have any confidence that a system which has not been verified is secure? Perhaps there is a correlation between this fact and the continual reporting of information security breaches on the evening news. In contrast, computer safety and reliability books and standards include extensive discussions of verification activities.[288]

It is important to understand that the effectiveness of the threat control measures — not generic system functionality — is being verified. This component does not involve general-purpose system validation and verification. As Schneier[411] succinctly states:

> *…[security] flaws cannot be found through normal beta testing. Security has nothing to do with functionality. A cryptography product can function normally and be completely insecure.*

* As a historical note, the *Orange Book* specified security testing requirements, by evaluation class, the qualifications of people performing the testing, the duration of the testing, and high-level testing criteria.[135,141]

In Chapter 6, specific threat control measures were implemented in response to particular vulnerabilities and threats. The threat control measures included IA design techniques/features, operational procedures, contingency plans, and physical security practices. This component confirms that these measures do (or do not) in fact eliminate or mitigate the vulnerabilities and threats against which they were deployed. This component also demonstrates whether or not the specified IA integrity level was achieved.

7.1 Select/Employ IA Verification Techniques

A combination of static and dynamic techniques are employed to verify the effectiveness of threat control measures. Exhibit 1 lists 18 proven IA verification techniques. A description of each technique is provided in Annex B, which discusses the purpose, benefits, and limitations of each technique and provides pointers to references for further information.

In addition, several IA analysis and accident/incident investigation techniques can be used to verify the effectiveness of threat control measures, including cause consequence analysis, common cause failure analysis, event tree analysis, HAZOP studies, Petri nets, software and system FMECA, software and system FTA, sneak circuit analysis, barrier analysis, and damage mode effects analysis. Exhibit 2 lists the IA verification role played by each of these techniques.

The effectiveness of threat control measures is verified through a three-step process:

1. Verify that appropriate IA design techniques/features were selected.
2. Verify that IA design techniques/features were implemented correctly.
3. Verify the robustness and resiliency of the threat control measures.

Each step must be successfully completed before continuing to the next step; there is no point in continuing if the preceding step has failed.

The first step is to verify that appropriate IA design techniques/features were selected to eliminate or mitigate specific vulnerabilities/threats. Consult Chapter 6, Exhibit 24 when performing this exercise. Inappropriate techniques/features will be ineffective and may give a false sense of security. Common mistakes include assuming firewalls perform authentication functions or using encryption to enhance data integrity. Mismatches between vulnerabilities/threats and the IA design techniques/features intended to control them are highlighted for correction. The set of IA design techniques/features should be complementary, not redundant; redundant techniques should be highlighted for resolution.

The second step is to verify that the IA design techniques/features were implemented correctly and that the corresponding operational procedures and contingency plans are accurate and complete. Several items are examined, including:

Exhibit 1 IA Verification Techniques

IA Verification Techniques	*C/R*	*Type*	*Life-Cycle Phase in which Technique is Used*		
			Concept	*Development*	*Operations*
Boundary value analysis	C3	All		x	x
Cleanroom	C3	All		x	
Control flow analysis[a]	C3	All		x	x
Data or information flow analysis[a]	C3	All		x	x
Equivalence class partitioning	C3	All		x	x
Formal proofs of correctness	C3	SA, SE	x	x	x
Interface testing	C3	All		x	x
Performance testing	C3	All		x	x
Probabilistic or statistical testing	C3	All		x	x
Regression testing	C3	All		x	x
Reliability estimation modeling	C3	RE		x	x
(IA) requirements traceability	C3	All	x	x	x
Review IA integrity case[a]	C3	All	x	x	x
Root cause analysis[a]	C3	All		x	x
Safety/security audits, reviews, and inspections	C3	SA, SE		x	x
Stress testing	C3	All		x	x
Testability analysis, fault injection, failure assertion	C3	All		x	x
Usability testing	C3	All		x	x

[a] These techniques can also be used during accident/incident investigations.

Source: Adapted from Herrmann, D., *Software Safety and Reliability: Techniques, Approaches and Standards of Key Industrial Sectors*, IEEE Computer Society Press, 1999.

Legend for Exhibit 1

Column	*Code*	*Meaning*
Type	SA	Technique primarily supports safety engineering
	SE	Technique primarily supports security engineering
	RE	Technique primarily supports reliability engineering
	All	Technique supports a combination of safety, security, and reliability engineering
C/R	Cx	Groups of complementary techniques
	Rx	Groups of redundant techniques; only one of the redundant techniques should be used

- Does the threat control measure execute within the correct sequence of events? Is it implemented in the correct execution/attack point in the system?
- Does the threat control measure interact correctly with other IA design techniques/features, especially with regard to defense in depth?
- Have system integration issues been handled correctly; for example, interfaces, parameter initialization and processing, default values and settings, etc.?

Exhibit 2 Verification Role of IA Techniques

Technique	*IA Verification Role*
I. Verification Techniques	
Boundary value analysis	Identify software errors that occur in IA-critical and IA-related functions and entities when processing at or beyond specified parameter limits, whether inputs or outputs.
Cleanroom	Prevent defects from being introduced or remaining undetected in IA-critical and IA-related functions and entities through an evaluation of the completeness, consistency, correctness, and unambiguousness of requirements, design, and implementation.
Control flow analysis	Uncover poor and incorrect program logic structures that could compromise IA integrity.
Data or information flow analysis	Uncover incorrect and unauthorized data transformations and operations that could compromise IA integrity.
Equivalence class partitioning	Identify the minimum set of test cases and test data that will adequately test each input domain.
Formal proofs of correctness	Prove that the requirements, design, and implementation of IA-critical and IA-related functions and entities are correct, complete, unambiguous, and consistent.
Interface testing	Verify that interface requirements are correct and that interfaces have been implemented correctly.
Performance testing	Verify whether or not a system will meet stated performance requirements and that these requirements are correct.
Probabilistic or statistical testing	Provide quantitative assessment of operational IA integrity; verify design integrity against operational profiles.
Regression testing	Verify that changes or enhancements have been implemented correctly and that they do not introduce new errors or affect IA integrity.
Reliability estimation modeling	Estimate software reliability for the present or some future time.
(IA) requirements traceability	Verify that (1) all safety, reliability, and security requirements derived from IA goals are correct; (2) all safety, reliability, and security requirements have been implemented correctly in the end product; and (3) no additional unspecified or unintended capabilities have been introduced.
Review IA integrity case	Determine if the claims made about IA integrity are justified by the supporting arguments and evidence.
Root cause analysis	Identify the underlying cause(s), event(s), conditions, or actions that individually or in combination led to an accident/incident; determine why the defect was not detected earlier.
Safety/security audits, reviews, and inspections	Uncover errors and mistakes throughout the life of the system that could affect IA integrity.
Stress testing	Determine (1) maximum peak loading conditions under which a system will continue to perform as specified and IA integrity will be maintained, and (2) system overload/saturation conditions that could lead to a system compromise or failure.

Exhibit 2 Verification Role of IA Techniques (continued)

Technique	*IA Verification Role*
Testability analysis, fault injection, failure assertion	Verify IA integrity by determining if a system design can be verified and is maintainable, and that it detects and responds correctly to erroneous data, conditions, and states.
Usability testing	Determine if a system performs in the operational environment in a manner acceptable to and understandable by administrators and end users; verify that the design does not contribute to induced or invited errors that could lead to a system compromise or failure.
II. Analysis Techniques	
Cause consequence analysis	Identify inappropriate, ineffective, and missing threat control measures; verify that all accidental and intentional failure modes have a corresponding threat control measure.
Common cause failure analysis	Verify that fault tolerant design components are immune to CCFs.
Event tree analysis	Identify inappropriate, ineffective, and missing threat control measures.
HAZOP study	Verify that all accidental and intentional, physical and cyber, hazards associated with the operation of a system have been eliminated or mitigated.
Petri nets	Verify that deadlock, race, and nondeterministic conditions that could cause a system compromise or failure do not exist.
Software, system FMECA	Examine the effect of accidental and intentional, random and systematic failures on system behavior in general and IA integrity in particular.
Software, system FTA	Identify potential root cause(s) of undesired system events (accidental and intentional) to verify the effectiveness of mitigating design features and operational procedures.
Sneak circuit analysis	Verify that all hidden, unintended, and unauthorized hardware and software logical paths or control sequences that could inhibit desired system functions, initiate undesired system events, or cause incorrect timing and sequencing have been removed.
III. Accident/Incident Investigation Techniques	
Barrier analysis	Ascertain which defensive layers failed or were missing or inadequate during an accident/incident.
Damage mode effects analysis	Postulate which specific threat mechanisms caused an accident/incident from an analysis of the damage modes.

- Is the threat control measure implemented in the correct TCP/IP or ISO OSI layer(s)?
- Have concerns about COTS products been dealt with correctly? Consult Chapter 5, Exhibit 10 when evaluating this.

Verifying that intrusion detection profiles or access control rules have been implemented correctly are examples. Correct implementation in relation to

the operational environment, not just the design, is evaluated.[362] Several IA verification techniques can be used during this step, such as boundary value analysis, cleanroom, equivalence class partitioning, formal proofs of correctness, interface testing, performance testing, probabilistic testing, IA requirements traceability, fault injection, and usability testing. Safety/security audits and usability testing and analysis can be used to verify operational procedures and contingency plans.

The first two steps parallel normal verification activities somewhat, although the focus is on safety and security — and not system functionality. The third step, verifying the robustness and resiliency of threat control measures, is when IA verification activities diverge from the norm. Instead of proving that a system functions correctly, the intent is to see how a system's threat control measures can be broken, bypassed, or disabled. This is where the "twisted mindset that can figure out how to get around rules, break systems, and subvert a designer's intentions," described by Schneier,[411] comes into play. A variety of verification techniques can be used to verify the robustness and resiliency of threat control measures: control flow analysis, data flow analysis, interface testing, performance testing, reviewing IA integrity cases, root cause analysis, safety and security audits, stress testing, and fault injection.

From the outside, safety and security testing may appear to be random. In fact, it is quite methodical. An attack, especially an organized attack, follows the same process; however, it is not encumbered or biased by knowledge of the system design and development. Consequently, it is highly recommended that safety and security testing be conducted by an independent team. Most national and international standards require this independence.[18,24,31,38,53,57,60,63–69,124–127,129,130,143]

As mentioned, safety and security testing attempts to discover if and how a system's threat control measures can be defeated, accidentally or intentionally. For example, by:

- Taking advantage of errors in the system design, operational procedures, or physical security practices
- Inducing transient faults, through an unusual combination or sequence of events, that can be exploited for malicious purposes
- Fooling people or processes into doing or permitting something they normally would not
- Co-opting unintended or unauthorized functionality for devious purposes

Design errors, such as timing errors or inconsistencies, not accounting for all possible logic states, incorrect data or control flow, unused or unreachable code, inadequate or inconsistent authentication parameters, and inadequate or conflicting access control rules, can be exploited as easily as loopholes in operational procedures or physical security practices. The design error or operational security loophole may be accidental, but the exploitation is intentional. An attack that takes advantage of design errors is difficult to detect and often remains undetected until it is too late; hence, the importance of this type of verification.

Transient faults are the nemesis of any verification activity. How does one test against an unforeseen temporary state? To illustrate, the first vertical launch of the Space Shuttle in 1980 was delayed two days due to a transient fault. Three computers controlled the main engine in a triple parallel redundant design with 100 percent voting/agreement. Following a transient fault, the three computers did not agree, causing the launch to be halted. It was reported that the transient fault had been experienced once before in the lab, but engineers had been unable to duplicate the condition and as a result verify its resolution.

Safety and security testing verifies system behavior during transient faults; it attempts to uncover all abnormal conditions and events for which the system is not protected. Transient faults can be induced through an unusual unanticipated combination or sequence of events (commands, responses, input, etc.), a sudden change or degradation of the operational environment (power drop or spike, increase in temperature or humidity, temporary saturation of a system entity, etc.), or a temporary loss of synchronization among system entities. Transient faults can be exploited to compromise a system or render it inoperable. As a case in point, during the simulation and testing of nuclear missile software, it was discovered that the targeting coordinates would reset to latitude 0°/longitude 0° if a particular transient fault was introduced at a specific interval during the launch control sequence. Several IA design techniques can be employed to mitigate transient faults, including accounting for all possible logic states, diversity, block recovery, defensive programming, and error detection/correction.

Fooling people into doing or permitting things they normally would not has been around as long as the human race. Remember the biblical story of Sarah pretending to be Abraham's sister so that the Pharaoh would not kill him? Attempting to fool computer processes began with the computer age. Distributed processing, LANs, WANs, and the Internet have made it easier and more widespread. Masquerading, IP-spoofing, "man-in-the-middle," password guessing, and replay attacks are common examples. Some of these attacks require sophistication; others do not. Most systems will be subjected to these generic types of attacks; therefore, all systems should be tested to ascertain their ability to withstand them. At the same time, operational procedures and physical security practices should be evaluated for vulnerabilities to masquerading, spoofing, and replay attacks.

Unintended, unauthorized functionality can serve as a conduit for attacks and intruders. These hidden logic paths, referred to as sneak circuits, permit a user or process to inhibit desired functionality, initiate undesired functionality, and bypass normal safety and security controls. A sneak circuit might give an intruder root access to a server, bypass access control rules, or prevent an audit trail of malicious activity from being recorded. Several companies currently have accounting, timekeeping, and other systems that employees access from a telephone keypad, usually by entering their employee number and password. (For the moment, the inherent insecurity of these systems will be ignored because transactions are conducted over open telephone lines, subject to eavesdropping, replay, masquerading, etc.) Most of these systems

have a simplistic user interface, no data integrity or privacy protections, and no immunity from misuse or abuse. One would be surprised how some of these systems respond when a * or # is entered in the middle of a transaction.

Sneak circuits can be designed into a system accidentally or intentionally. Safety and security testing checks for the presence of sneak circuits through a combination of static and dynamic analyses. As Dima, Wack, and Wakid[251] state:

> *Security testing has to do more than just determine if the system conforms to some specification or standard. It must also test the implementation — in other words, it must pinpoint if any of the system's functions are unintended or unauthorized.*

Creative "what-if …" testing is an essential part of verifying the robustness and resiliency of threat control measures. This is the time to explore system behavior in response to "I wonder what would happen if …" test scenarios. The intent is to discover how threat control measures can be broken, bypassed, or disabled. Do not worry about crashing the system; after all, it is preferable to crash the system by finding an ineffective threat control measure rather than to let it remain in the system for an attacker to find. Transaction paths and critical threat zones are analyzed to identify potential attack points and develop "what-if …" test scenarios. Accidental and intentional vulnerabilities, accidental and intentional actions are exercised in "what-if …" test scenarios.

"What-if …" test scenarios are unique to each system and correlate to the required integrity level. Two high-level examples follow. They are intended to (1) illustrate the thrust of safety and security testing, and (2) stimulate ideas about the types of situations and circumstances to test for in "what-if …" test scenarios. Note that these examples are by no means exhaustive. Exhibits 3 through 5 depict "what-if …" test scenarios for the three hypothetical systems discussed in this book: a radiation therapy system, an ATC system, and an online banking system. Exhibit 6 provides a test scenario checklist for three common threat control measures: access control, audit trail/security alarm, and defense in depth.

7.2 Determine Residual Risk Exposure

The initial risk exposure determined the type and extent of threat control measures and IA integrity level required. Now, after the threat control measures have been implemented, an assessment is made of whether or not the residual risk exposure is acceptable and consistent with the target risk exposure. Several questions are pursued in this regard:

1. Did the threat control measures reduce the likelihood and severity of potential hazards as planned?
2. Has the initial risk exposure been reduced to ALARP?

Exhibit 3 Sample High-Level Test Scenarios for Verifying the Effectiveness of Threat Control Measures: The Radiation Therapy System

I. Radiation Therapy System

- How does the system respond to a suboptimal operational environment: (a) heat? (b) humidity? (c) dust? (d) vibration? (e) noise? (f) power faults? (g) EMI/RFI/EMC?

1. Patient records database
   - Can the patient records database be accessed without authorization: (a) from the local clinic LAN? (b) from the remote billing system? (c) from the remote research database? (d) from …?
   - Can information in the patient records database be copied, deleted in whole or in part, modified, or added to without authorization?
   - Does the system record an audit trail of unauthorized access to or use of system resources? Is a security alarm generated?
   - Does the system alert valid operators that unauthorized changes have been made to treatment profiles before a therapy session can begin?
   - How easy is it to fake user authentication parameters?
   - How does one know that: (a) the correct patient's treatment profile is retrieved before a therapy session? (b) the current treatment capture is stored under the correct patient's name? (c) a query against past treatments retrieves the correct patient's records?
   - Can patient records be overwritten accidentally or intentionally?
   - Can the patient records database be accidentally or intentionally saturated, from internal or external sources, so that no records can be retrieved?
2. Treatment planning system
   - Can information about the tumor characteristics, treatment algorithm, or treatment plan be accessed without authorization: (a) from the local clinic LAN? (b) from the remote billing system? (c) from the remote research database? (d) from …?
   - Can the tumor characteristics information, treatment algorithm, or treatment plan be copied, deleted in whole or in part, modified, or added to without authorization?
   - Does the system record an audit trail of unauthorized access to or use of system resources? Is a security alarm generated?
   - Does the system alert valid operators that unauthorized changes have been made to the treatment plan, tumor characteristics, or treatment algorithm before a therapy session can begin?
   - How easy is it to fake user authentication parameters?
   - Does the system automatically check for illegal combinations of beam type, duration, dosage, number of targets (if fractionated therapy), etc.?
   - Can the software that performs these checks be surreptitiously altered? Does the system detect and alert valid operators of these unauthorized modifications?
   - How does one know that a treatment plan is stored under the correct patient's name?
   - Can a treatment plan be overwritten accidentally or intentionally?
   - Can the treatment planning system be accidentally or intentionally saturated, from internal or external sources, so that no plans can be retrieved?
3. Radiation delivery unit
   - Does the radiation unit in fact deliver the correct dosage to targets specified in the treatment profile?

Exhibit 3 Sample High-Level Test Scenarios for Verifying the Effectiveness of Threat Control Measures: The Radiation Therapy System (continued)

- Does the radiation unit deliver radiation: (a) other than as specified: different beam type, dosage, etc.? (b) when not specified? Can it be surreptitiously be made to do (a) or (b)?
- Can the firmware controlling the radiation unit be surreptitiously modified? Does the system alert valid operators of unauthorized changes before beginning a therapy session?
- Can the electrical, electronic, or mechanical components be induced to fail: (a) at certain times? (b) in certain modes? Does the system detect and alert valid system operators of these failures? Are the events that lead to these failures detectable? Preventable?
- What is the minimum safe interval between therapy sessions so that treatment profiles for one patient are not accidentally carried forward to the next?
- What ensures that default or uninitialized treatment parameters are not used?

4. People
   - Are the operators, calibration, and maintenance staff cognoscente of system safety and security features and procedures?
   - Are they proficient at using these features and procedures?
   - Are safety and security procedures followed?
   - Do operators, trainers, calibration and maintenance staff know how to report and respond to: a) an anomalous situation? b) a suspected safety or security compromise? c) warnings and alarms?
   - Have they been trained about how and when to invoke contingency plans?

3. Is the residual risk exposure acceptable within known operational constraints?
4. Has the specified IA integrity level been demonstrated?
5. Are there opportunities to improve or optimize IA design techniques/features, operational procedures, contingency plans, or physical security practices?

Residual risk exposure is evaluated for all applicable scenarios:

- Different operational modes/states, profiles, environments, and missions
- Normal and abnormal conditions and events
- Independent, dependent, and simultaneous hazards
- Random and systematic failures
- Accidental and malicious intentional failures
- Physical and cyber hazards

At the same time, the analysis verifies that the threat control measures did not introduce any new hazards. It is highly recommended that an internal assessment be supplemented by an independent assessment of the adequacy, appropriateness, and effectiveness of threat control measures.

Using the results from the static and dynamic analyses, threat control measures are mapped to vulnerabilities/threats to develop a threat control

Exhibit 4 Sample High-Level Test Scenarios for Verifying the Effectiveness of Threat Control Measures: The ATC System

II. ATC System

1. Pilot, aircraft flight control/navigation system
   - Can pilots be fooled into thinking that they are talking with real air traffic controllers when, in fact, they are not?
   - Can instrumentation readings be accidentally or intentionally altered or corrupted? Is an alarm generated or any evidence provided to alert the pilot of this situation?
   - Can flight control software be accidentally or intentionally altered or corrupted? Is an alarm generated or any evidence provided to alert the pilot or maintenance crew of this situation?
   - Can the aircraft location signal, sent to the ATC radar, be: (a) frequency modulated without authorization? (b) jammed? (c) intercepted? (d) retransmitted, such that the signal is erroneously repeated at a later time, deleted and replaced by a bogus signal, transmitted in the wrong sequence, delayed, modified, or corrupted?
   - Can the aircraft location signal transmitter be disabled accidentally or intentionally, such that it is not transmitting but the controls indicate that it is functioning normally?
2. Radar
   - Can the radar be fooled into "thinking" that a bogus signal is in fact coming from a real aircraft?
   - Can the radar receiver be disabled accidentally or intentionally, such that it is not receiving aircraft location signals but the controls indicate it is functioning normally?
   - Can the radar transmitter be disabled accidentally or intentionally, such that it is not transmitting information to the ATC system but the controls indicate it is functioning normally?
   - Can the signal from the radar to the ATC system be: (a) frequency modulated without authorization? (b) jammed? (c) intercepted? (d) retransmitted, such that the signal is erroneously repeated at a later time, deleted and replaced by a bogus signal, transmitted in the wrong sequence, delayed, modified, or corrupted?
   - Can radar system software be accidentally or intentionally altered or corrupted? Is an alarm generated or any evidence provided to alert the operators or maintenance crew of this situation?
3. ATC system
   - Can air traffic controllers be fooled into thinking they are talking to real pilots when, in fact, they are not?
   - Can the air traffic control system be fooled into "thinking" that a bogus signal is in fact coming from a real radar system?
   - Can the ATC system receiver be disabled accidentally or intentionally, such that it is not receiving signals from the radar but the controls indicate it is functioning normally?
   - Can the ATC system receiver detect whether or not signals from the radar have been repeated, resequenced, delayed, modified, or corrupted?
   - Can the air traffic controller terminal be accidentally or intentionally corrupted, so that the screen freezes temporarily or permanently, duplicate data is displayed, some data points are deleted, the screen goes blank temporarily, bogus data points can be inserted, screen refreshes with new data points are delayed, information is displayed on the wrong air traffic controller's terminal, or the terminal becomes inoperable?

Exhibit 4 Sample High-Level Test Scenarios for Verifying the Effectiveness of Threat Control Measures: The ATC System (continued)

- Can the ATC DBMS be accidentally or intentionally corrupted, so that bogus data can be added to the database, legitimate data can be deleted from the database, data can be modified or duplicated, old data can overwrite current data, pointers or indices used to access the data are scrambled, or the data is unintelligible or unavailable?
- Can communication between the ATC DBMS and air traffic controller terminals be accidentally or intentionally corrupted, so that information is sent to the wrong controller's terminal, information is sent to controllers' terminals too early, too late, or in the wrong sequence, information is withheld or not sent to the controllers' terminals, wrong information is sent to a controller's terminal, or the communication link between the ATC DBMS and controllers' terminals is inoperable?
- How does the system respond to a suboptimal operational environment: (a) heat? (b) humidity? (c) dust? (d) vibration? (e) noise? (f) power faults? (g) EMI/RFI/EMC?

effectiveness assessment, as shown in Exhibit 7. First, the specific vulnerability/threat is identified, along with the severity, likelihood, and TCP/IP or ISO OSI layers in which it occurs. Next, the IA design techniques and features deployed to control this specific vulnerability/threat are identified. The TCP/IP or ISO OSI layer(s) in which the techniques are effective is (are) listed. The phase(s) in the threat control chronology in which the techniques are effective is (are) indicated: anticipate/prevent, detect/characterize, and respond/recover. The EAL and the demonstrated integrity level of the system are cited.

Given this information, the threat control effectiveness assessment seeks to uncover any mismatches or gaps in controlling this vulnerability/threat. Seven key factors are investigated as part of this assessment:

1. The appropriateness of this set of techniques for eliminating or mitigating this vulnerability/threat (Chapter 6, Exhibit 24 is reviewed)
2. The effectiveness of this set of techniques against all operational modes/states and profiles in which this vulnerability/threat occurs (the system operation characterization is reviewed)
3. Whether or not this set of techniques covers all layers in the TCP/IP or ISO OSI reference model in which the vulnerability/threat occurs (Chapter 6, Exhibits 12 and 13 are reviewed)
4. Whether or not this set of techniques covers all phases of the threat control chronology (Chapter 6, Exhibit 23 is reviewed)
5. Whether or not the EAL is appropriate and the static and dynamic analyses results positive
6. Whether or not the demonstrated IA integrity level corresponds to the required IA integrity level
7. Whether or not this set of techniques provides adequate defense in depth

Exhibit 5 Sample High-Level Test Scenarios for Verifying the Effectiveness of Threat Control Measures: The Online Banking System

III. Online Banking System

1. Home PC/user
   - Can a home PC user fake authentication parameters to the online banking system?
   - Can a home PC user bypass access control rules when acting as a legitimate user or masquerading as another user?
   - Can a home PC user intentionally saturate the online banking system so that it becomes unstable and exhibits unpredictable behavior?
   - Can a home PC user accidentally or intentionally provide erroneous input that could lead to a system compromise or failure?
   - Can account transactions be accidentally or intentionally erased, inhibited, modified, or initiated without authorization?
   - Can the transaction audit trail be accidentally or intentionally altered, corrupted, or erased?
   - Can data files be accessed without executing the appropriate application software?
   - Are data files containing authentication parameters protected?
   - Can account data be accessed without authorization: (a) from other financial institutions? (b) from other internal bank systems? (c) from …?
   - How does the system respond to a suboptimal operational environment: (a) heat? (b) humidity? (c) dust? (d) vibration? (e) noise? (f) power faults? (g) EMI/RFI/EMC?
   - Can account transactions be initiated without authorization: (a) from other financial institutions? (b) from other internal bank systems? (c) from …?
2. Online banking system
   - Can account data be accidentally or intentionally overwritten, modified, deleted, copied, read, printed, or added to without authorization?
   - Can the online banking system be fooled into "thinking" it is interacting with a legitimate user or other financial system when, in fact, it is not?
   - Can hidden accounts exist in the system?
   - Can funds be surreptitiously moved from account to account but appear to be in the correct account for audit purposes?
   - Can transaction records be accidentally or intentionally linked to the wrong account number?
   - Can transient fault conditions be induced so that access controls are bypassed and transactions are not recorded?
   - How does the system respond to a suboptimal operational environment: (a) heat? (b) humidity? (c) dust? (d) vibration? (e) noise? (f) power faults? (g) EMI/RFI/EMC?
3. Communications link between home PC user and online banking system
   - How secure are the communications protocols?
   - Can sessions be hijacked or listened to?
   - Can authentication parameters or transaction data be intercepted?
   - Can this information be used to initiate a fake session later?
   - Can a home PC user be prevented from accessing the online banking system?
   - Can transaction data be modified between the home PC user and the online banking system?
   - Can an intruder listen to online banking transactions to learn about a person's financial status and habits?

Exhibit 6 Checklist for Verifying the Effectiveness of Three Threat Control Measures

1. Access control
   - Have all access control rules been validated?
   - Have all access control rules been implemented correctly?
   - Are there any unintended data or control flows?
   - Are all inferred access control rules acceptable?
   - Are the access control rules consistent and complete?
   - Are the access control features implemented in the correct TCP/IP or ISO OSI layers? In the correct execution sequence?
   - What happens when an access control feature is saturated?
   - What happens when the process invoked immediately *preceding* access control is disabled or fails?
   - What happens when the process invoked immediately *after* access control is disabled or fails?
   - What happens if a transient fault is introduced during access control mediation?
   - What happens if the authentication function fails, is bypassed, or disabled?
   - What happens if the authentication parameters passed to the access control function are corrupted?
   - Can the table defining access control rights and privileges be accidentally or intentionally overwritten, copied, modified, deleted in whole or in part, or added to without authorization?
2. Audit trail, security alarm
   - Does the audit trail record all necessary activity? Are the activities recorded at meaningful intervals?
   - Does the audit trail record unnecessary events?
   - Can the audit trail be accidentally or intentionally overwritten, modified, deleted in whole or in part, copied, or added to without authorization?
   - Can the audit trail function be intentionally bypassed, disabled, or induced to fail?
   - What happens if the audit trail function becomes saturated?
   - Can events recorded in the audit trail be faked to cover malicious activity and prevent an alarm from being triggered?
   - Are alarms triggered in a timely manner?
   - Are the alarm false-positive and false-negative rates acceptable?
   - Can the parameters for triggering an alarm be accidentally or intentionally overwritten, modified, copied, deleted in whole or in part, or added to without authorization?
   - What happens if there is no human-initiated response to an alarm?
   - What happens if there is no computer-initiated response to an alarm?
   - Is it possible to block events from being recorded in the audit trail?
   - Is it possible to intercept an alarm before it is distributed?
3. Defense in depth
   - Are the defensive layers complementary or redundant?
   - Do the techniques cover all phases of the threat control chronology?
   - Do the techniques cover all applicable layers in the TCP/IP or ISO OSI reference models?
   - Are the defensive layers subject to CCFs?
   - Are there dependencies between the techniques, for example, the input to one technique is dependent on the output of another technique?
   - If the technique(s) responsible for anticipate/prevent fails is bypassed or disabled, will the detect/characterize and respond/recover techniques function correctly?

Exhibit 6 Checklist for Verifying the Effectiveness of Three Threat Control Measures (continued)

- If the detect/characterize technique(s) fails is bypassed or disabled, will the respond/recover technique(s) function correctly?
- Are there any unnecessary time delays after the failure of one defensive layer and before the next defensive layer becomes effective or is activated?
- Does the failure of one defensive layer reveal any information or create additional vulnerabilities for the other defensive layers?
- Can all the defensive layers be simultaneously saturated or attacked?
- Do the defensive layers include operational procedure provisions, contingency plan provisions, and physical security practices?

Answers to these inquiries highlight missing, inappropriate, and ineffective threat control measures for individual vulnerabilities/threats. Information about individual vulnerabilities/threats is combined to produce a threat control effectiveness summary, as shown in Exhibit 8. Responses to the seven key inquiries are summarized by vulnerability/threat severity. The resulting one-page digest quantitatively illustrates the acceptability of the residual risk or, conversely, the need for further risk reduction activities.

The information needed to evaluate these seven factors is found in the IA integrity case. An IA integrity case is a systematic means of gathering, organizing, analyzing, and reporting the data needed by internal, contractual, regulatory, and certification authorities to confirm that a system has met the specified IA goals and integrity level and is fit for use in the intended operational environment. Many national and international standards require a system safety or system reliability case as part of the certification and approval process.[31,38,57,63–65,124,129,130] This book expands that concept to the broader realm of IA.

Presenting information in a logical, complete, and concise manner is the hallmark of a well-founded IA integrity case. Exhibit 9 depicts the structure of an IA integrity case. Section 1 states the IA goals for the system, the justification for those goals, and the required IA integrity level. This information was developed in activities 1.1 (Chapter 4) and 3.1 (Chapter 6). Section 2 states assumptions that have been made about the development environment, operational environment, operational profiles, and operational mission of the system. Claims are made about the relevance of previous experience with similar systems or technology and the design, development, and verification techniques and processes used. Relevant evaluations of COTS products by independent laboratories are cited, such as an EAL.

Section 3 (Exhibit 9) contains the evidence needed to substantiate the conclusions and recommendations in Section 5. This evidence represents the results of all the analyses conducted to date. To be credible, the evidence must be complete and current — several of these analyses are updated frequently. Additional backup or supporting information can be included in Annex A. The evidence incorporates the following:

Exhibit 7 Threat Control Effectiveness Assessment

System/Entity:

Date:

I. Vulnerability/Threat Identification[1]

No.	*Description*	*Severity*	*Likelihood*	*Present in layer(s)*	
				TCP/IP	*ISO OSI*
1					

II. Threat Control Measures[2]

IA Design Technique/ Feature	*Effective in Layer(s)*		*Threat Control Chronology Effectiveness*			*EAL*	*IA Integrity Level*
	TCP/IP	*ISO OSI*	*A/P*	*D/C*	*R/R*		
1a							
1b							
1c							

Key: A/P - anticipate/prevent; D/C - detect/characterize; R/R - respond/recover.

II. Assessment[3]

a. Is this set of techniques appropriate for eliminating or mitigating this vulnerability/threat?
b. Is this set of techniques effective against all operational modes/states and profiles in which this vulnerability/threat occurs?
c. Does this set of techniques cover all layers in which the vulnerability/threat occurs?
d. Does this set of techniques cover all phases of the threat control chronology?
e. For each technique/feature: (a) is the EAL appropriate? (b) are the static and dynamic analysis results positive?
f. Is the demonstrated IA integrity level of this set of techniques consistent with the required IA integrity level?
g. Does this set of techniques provide adequate defense in depth?
h. Are there any mismatches or gaps in controlling this vulnerability/threat?

[1] A separate template is prepared for each vulnerability/threat pair. The information in Part I comes from the system vulnerability and threat characterizations.

[2] All threat control measures that eliminate or mitigate the vulnerability/threat cited in Part I are listed.

[3] Except for h, all answers should be an unequivocal yes or no; if no, a rationale should be provided.

- System vulnerability characterization
- System threat characterization
- Critical threat zones
- IA design techniques/features implemented
- Demonstrated IA integrity level
- Threat control effectiveness assessment
- Residual risk exposure

Exhibit 8 Threat Control Effectiveness Summary

Assessment Criteria	Vulnerability/Threat Severity: Catastrophic #	Catastrophic %	Critical #	Critical %	Marginal #	Marginal %	Insignificant #	Insignificant %
1. TCP or ISO OSI layers:								
a. Covered								
b. Not covered								
2. Operational modes/states, operational profiles:								
a. Covered								
b. Not covered								
3. Phases of threat control chronology:								
a. Covered								
b. Not covered								
4. EAL, static and dynamic analysis results:								
a. Appropriate								
b. Inappropriate								
5. Demonstrated IA integrity level:								
a. Appropriate								
b. Inappropriate								
6. Defense in depth:								
a. Adequate								
b. Inadequate								
7. Threat control gaps or mismatches:								
a. None remaining								
b. Some remaining								
Total vulnerabilities/threats		100%		100%		100%		100%

This information was developed in activities 2.2, 2.3, and 2.5 (Chapter 5), 3.5 (Chapter 6), and 4.1 and 4.2 (Chapter 7).

A chronological list of issues and their resolution is kept in Section 4. Section 5 contains the conclusions and recommendations from various stakeholders that the assumptions, claims, and evidence presented prove (or do not prove) that the IA goals and integrity level have been (or will be) achieved and maintained. A Certification Authority may concur, nonconcur, or request more information. Section 6 contains a chronology of reviews/approvals for the system.

An IA integrity case is a living document.[289] The case commences as soon as the IA goals are defined. Assumptions, claims, and evidence are added to the case throughout the system's development and operational phases. The IA integrity case is reviewed at regular milestones to verify that a system is on track for attaining or maintaining its IA goals and integrity level. An IA integrity case should be reviewed/revalidated whenever any of the claims,

Exhibit 9 Structure of an IA Integrity Case

System:_____

as of:_____

last review/approval:_____

1. IA goals
   a. IA goals for this system
   b. Justification for the IA goals
   c. IA integrity level required for this system
2. Assumptions and claims
   a. Assumptions about development environment, operational environment, operational profiles, operational mission
   b. Claims about previous experience with similar systems and technology
   c. Claims about design, development, and verification techniques and processes used
   d. Evaluations of COTS products by independent laboratories, such as an EAL
3. (Current) evidence
   a. System vulnerability characterization
   b. System threat characterization
   c. Critical threat zones
   d. IA design techniques/features implemented
   e. Demonstrated IA integrity level
   f. Threat control effectiveness assessment and summary
   g. Residual risk exposure
4. Outstanding issues
5. Conclusions and recommendations
   a. System developer
   b. System owner
   c. Regulatory authority (if applicable)
   d. Certification Authority
6. Approval, certification history

Annex A Backup, supporting information
   a. System definition
   b. System operational characterization
   c. System entity control analysis
   d. Transaction paths
   e. Operational procedures
   f. Contingency plans
   g. Static and dynamic analysis results
   h. Real-world experience with end users

assumptions, or evidence has changed, or following a system enhancement, correction, modification, reconfiguration, failure, or compromise. An IA integrity case should be reviewed as part of an accident/incident investigation to ascertain which claim(s), assumption(s), or evidence was (were) false.

7.3 Monitor Ongoing Risk Exposure, Responses, and Survivability

Verification of the effectiveness of threat control measures does not end once a system is fielded. In fact, some would argue, with merit, that is when the real verification of threat control effectiveness takes place. The ability to maintain a specified integrity level in the real-world operational environment, despite accidental and malicious intentional actions, is what information security/IA is all about.

The effectiveness of threat control measures during the in-service phase of a system is often appraised as a function of survivability. Survivability is defined as the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents.[336] As Linger[336] observes:

> *Survivability depends on three key capabilities: resistance, recognition, and recovery. Resistance is the capability of a system to repel attacks. Recognition is the capability to detect attacks as they occur and to evaluate the extent of damage and compromise. Recovery, the hallmark of survivability, is the ability to maintain essential services and assets during an attack, limit the extent of damage, and restore full services following an attack.*

Accordingly, a survivability assessment covers the full threat control chronology: anticipate/prevent, detect/characterize, respond/recover. This reinforces the need for defense in depth — a threat control measure designed to anticipate/prevent an attack may be disabled or bypassed, leaving the detect/characterize and respond/recover threat control measures to protect the system. Fault tolerance, fault containment, maintainability, and the ability to efficiently transition to degraded mode operations, a fail safe/secure or fail operational state when needed are characteristics of robust survivability. Contingency plans and operational procedures are also evaluated in regard to their contribution to survivability. Wylie et al.[444] point out that data as well as systems must be designed to be survivable. The importance of maintainability in relation to survivability is often overlooked. However, as Jackson[304] notes:

> *Maintainability is the systematic assessment of the effectiveness of maintenance strategies and can have considerable influence on system safety [and security].*

The resilience of all IA design techniques/features must be continually assessed. As McSteen and Pesante[350] point out:

> *… the security of an organization's online information and information systems will depend on, among other things, that organization's ability to stay current on ever-changing attack methods and the [inherent] security vulnerabilities in the technology they are using.*

Survivability should be reassessed at regular intervals. Changes or additions to the operational mission, profile, or environment and/or system enhancements, modifications, or reconfiguration should trigger an update to the system vulnerability/threat characterizations and a reevaluation of the threat control effectiveness assessment. It is highly recommended that internal survivability assessments be supplemented by those performed by independent teams.

Human and organizational factors are also considered when assessing survivability. One way to do this is to see how an organization's practices compare to the common accident/incident precursors and the proactive responses cited in Chapter 6, Exhibit 1. To ensure objectivity, an independent assessment is recommended. In addition, contingency plans and operational procedures should be examined in light of human-factor engineering concerns, especially proclivities toward induced or invited errors. Accidental induced or invited errors can lead to a system failure or create an opening for a malicious intentional act that leads to a system compromise. Real-world feedback from system administrators and end users about problems they have experienced, features they do not like, and recommendations they have for improving the plans and procedures must be incorporated.

7.4 Summary

The fourth component of an effective information security/IA program is verifying the effectiveness of threat control measures. Three activities are performed when verifying the effectiveness of threat control measures (as shown in Exhibit 10):

- IA verification techniques are selected and employed.
- Residual risk exposure is determined and its acceptability is evaluated.
- Ongoing vulnerabilities, threats, and survivability are monitored.

After threat control measures have been implemented, it is essential to verify that they do in fact eliminate or mitigate the vulnerabilities and threats against which they were deployed. Likewise, the IA integrity level achieved must be demonstrated. Without actual verification, there is no factual basis upon which to claim that a system is safe, secure, or reliable. To ensure objectivity, internal verification activities should be supplemented by those performed by independent teams.

A combination of static and dynamic analysis techniques are used to verify the effectiveness of threat control measures throughout the life of a system. A three-step process is followed:

1. Verify that appropriate IA design techniques/features were selected.
2. Verify that IA design techniques/features were implemented correctly.
3. Verify the robustness and resiliency of the threat control measures.

The third step is critical. It attempts to discover how threat control measures can accidentally or intentionally be broken, bypassed, or disabled. Creative

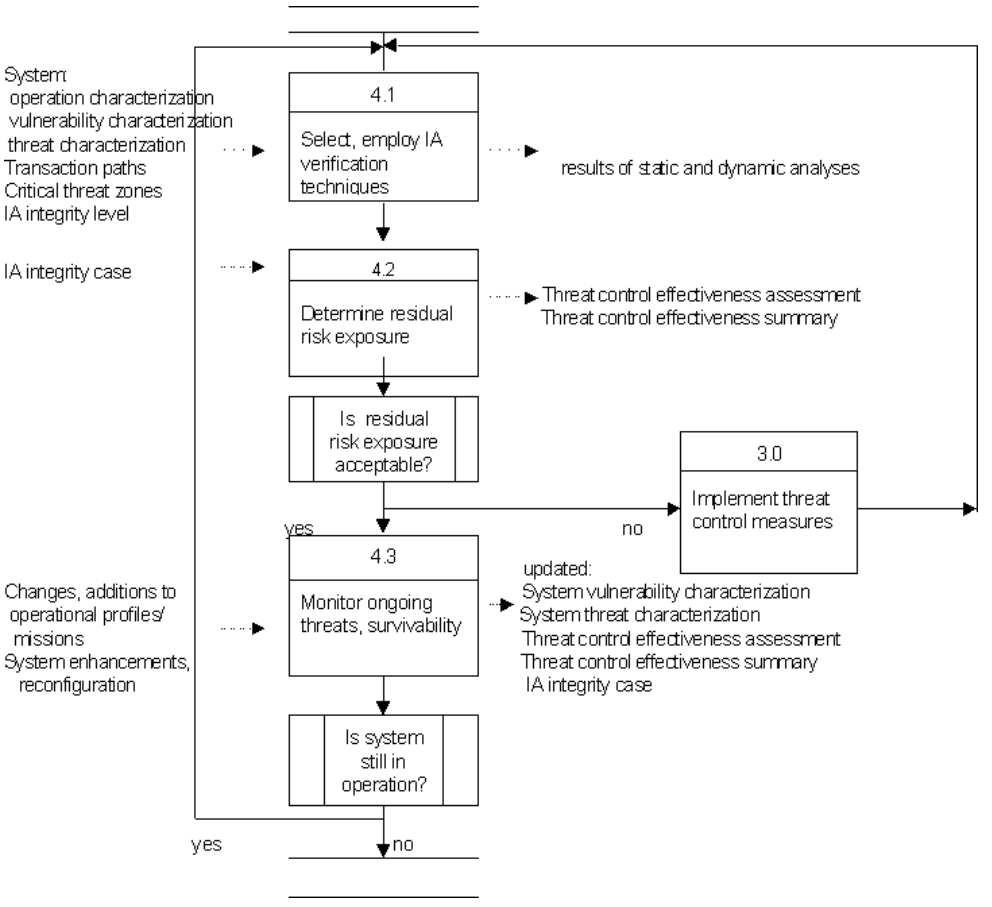

Exhibit 10 Summary of Activities Involved in Verifying the Effectiveness of Threat Control Measures

"what-if …" test scenarios are developed from an analysis of transaction paths and critical threat zones. Particular attention is paid to transient faults and sneak circuits.

Results of the static and dynamic analyses are mapped to vulnerabilities/threats to develop a threat control effectiveness assessment. Missing, inappropriate, ineffective, and redundant threat control measures are identified, along with gaps in covering all phases of the threat control chronology and applicable layers in the TCP/IP or ISO OSI reference models. Residual risk exposure is compared to the target to determine acceptability or the need for further risk reduction. An IA integrity case is a systematic means of collecting, organizing, analyzing, and reporting the information needed to evaluate residual risk.

Accidental and intentional vulnerabilities and threats are monitored throughout the life of a system. The effectiveness of threat control measures during the in-service phase is often assessed as a function of survivability. Human and organizational factors, such as opportunities for induced or invited errors, are evaluated as part of a survivability assessment.

Next, Chapter 8 explains how to and why one should conduct an accident/incident investigation.

7.5 Discussion Problems

1. When is the effectiveness of threat control measures verified?
2. What is the same, what is different between generic verification activities and verifying the effectiveness of threat control measures?
3. How is the effectiveness of COTS threat control products verified?
4. Why is the robustness and resiliency of threat control measures verified?
5. What role do the following items play in IA verification activities, if any: (a) transient faults, (b) sneak circuits, (c) transaction paths, and (d) critical threat zones?
6. For what scenario is residual risk exposure evaluated?
7. What does a threat control effectiveness assessment measure?
8. What can you learn about IA integrity from a threat control effectiveness summary?
9. Should IA verification techniques be selected to: (a) test an IA design technique/feature, or (b) test a known or suspected vulnerability/threat?
10. What is the purpose of including assumptions and claims in an IA integrity case?
11. How is residual risk exposure determined? How is the acceptability of residual risk evaluated?
12. What is involved in assessing survivability, and when is it assessed?
13. What effect, if any, do the following items have on the effectiveness of threat control measures: (a) human factors, (b) operational procedures, (c) contingency plans, (d) physical security practices, and (e) temperature?
14. How could a risk acceptability model, based on that discussed in Chapter 5.5, be used to determine the acceptability of a threat control effectiveness summary?
15. Develop some "what-if …" test scenarios for an intelligent transportation system. Identify which IA verification techniques would be used in each instance.