for requirements pertaining to installation and especially environmental controls. Generic papers on formulating computer security acceptance criteria and developing security standards are [NEUG82] and [KON81]. Further background material is contained in two NBS Special Publications on audit and evaluation of computer security ([NBS77] and [NBS80]). No single source provides all the questions or answers for a particular situation, but they do serve as useful judgmental aids in the evaluation process. Note that the judgments of acceptability made here and below are technical judgments and do not substitute for the overall decision made by the Accrediting Official.

2.3.2 Security Function Evaluation

2.3.2.1 With Defined Security Requirements—Given well-defined security requirements, function evaluation becomes the most important task in basic evaluation. It determines whether security functions (control techniques)[7] such as authentication, authorization, monitoring, security management, and security labeling [DoD83] satisfy security requirements. The primary method is simply to use the stated requirements as a checklist to follow in assessing whether they are satisfied. For example, where called for in requirements: Is individual accountability provided? Are subjects and objects identified and given security labels? Is an execute-only mode of access provided? Are all file accesses recorded? Are functions partitioned so as to provide separation of duties? Does a contingency plan exist and has it been tested [FIPS87]?

In some cases requirements specify only the need for a generic function such as authentication. In other cases the requirements call for use of a specific mechanism, such as a particular password technique. In both situations, function evaluation identifies the defined security function and examines it for acceptability.

2.3.2.2 Without Defined Security Requirements—Situations arise in which a reliable requirements baseline does not exist and it is not possible or appropriate to formulate one. These situations call for a more elaborate method for function evaluation. Most of the guidance sources discussed above under requirements evaluation are helpful in these situations. Several (e.g., [CIC75, GAO81-2, IBM80]) are structured in such a way that they might be termed as "methods" for doing this. Without a reliable requirements baseline to work from, however, it is difficult to assess control acceptability. Some controls are more important than others. Some are redundant or complementary. Some are effective while others may also be efficient. Some look effective but are not. Most are only effective if properly situated. Different controls have different purposes and are of differing quality.

One suitable "method" for those situations in which requirements are not well-defined is that in [MAI76], as summarized in [NBS83]. It examines how effectively controls counter specific threats and thereby reduce the resultant exposures. It also emphasizes the differing purposes and reliability of controls (e.g., computerized controls are more reliable) and incorporates analysis of control quality and placement. It emphasizes analysis of key controls. The emphasis on threats and exposures (though not on assets) makes the method similar to risk analysis. This is appropriate since, in lieu of well-defined requirements, a baseline is still needed against which to assess controls. Whereas risk analysis uses expected loss as a baseline, however, [MAI76] uses reduction of exposures.

2.3.2.3 Level of Detail—An important concern for function evaluation is the appropriate level of detail. The recommendation is that basic evaluations be complete (for all applicable control features) down through the functional level, where "functional level" is the logical level represented by functions as defined in (or appropriate for definition in) the Functional Requirements Document. This notion applies to both controls within the computer and physical/administrative controls external to it (although the latter might not actually be defined in a Functional Requirements Document).

---

7. At the functional level, application controls would be described ideally in terms of control techniques or standards. The actual control mechanisms selected would appear at the implementation level. However, in practice, these distinctions are often blurred.

This function evaluation approach is suggested in full realization of the difficulty sometimes confronted in determining which functions to include in a Functional Requirements Document. It is also recognized that many applications do not have such documents associated with them. Furthermore, where such documents exist, they are often incomplete, forcing an evaluator to examine operating procedures, specifications, and other documents, in search of functional control techniques that should have been identified in the Functional Requirements Documents. Nevertheless, the functional level (1) is the level best suited to serve as a "security specification" in compliance with OMB A-71 TM1 (as noted in [FIPS73, p. 28]); (2) is a legitimate, commonly-used level (e.g., see [FIPS38]); and (3) can, when done with care, represent a *complete* picture of security functions and services, with respect to the environment surrounding the application. Completeness is necessary to ensure that major problem areas are not overlooked. The functional level does not include evaluation of individual mechanisms used to implement the security functions. This is not a problem, however, because even though implementation mechanisms can certainly influence security, they represent a level of detail not needed in a basic evaluation.

2.3.2.4 Security Requirements Documents—At this point it is useful to discuss security requirements documents in more depth. Typically the user's initial statement of requirements are contained in the Project Request [FIPS64]. The *Functional Requirements Document* is produced during the Definition Phase of development (see Appendix F). It identifies application modules at the functional level and includes inputs, outputs, processing requirements, and system performance requirements. Controls identified are also in terms of application modules and needs. Examples of such functions include authentication (e.g., passwords), authorization (e.g., subject/object definition and capabilities), and security monitoring as well as proper operation, performance, and (ideally) penetration resistance of these functions. FIPS PUB 73 provides guidance in preparing a "security specification" at this level [FIPS73, pp 29-30].

In contrast, controls at the *System/Subsystem Specification* level are the specific mechanisms required in providing the functions defined in the Functional Requirements Document (i.e., the "how it works" as opposed to the "what it does"). Examples include internal password encryption and software-module checksums. Program Specification controls typically include control counts, balancing, and checks for format, sequence, completeness, and validity. Some of these are also introduced at the code level along with typical code-level controls such as checks for input/output device errors.

In many cases, a *Data Requirements Document* [FIPS38] is produced during the Definition Phase, along with the Functional Requirements Document. Information in the Data Requirements Document is also assessed during function evaluation. This information might reveal such things as unacceptable flow, backup, manipulation, or aggregation of data, where these were not detected during requirements evaluation as discussed in the preceding section. While this examination of the Data Requirements Document is important, primary attention is usually focused on the Functional Requirements Document because it defines required security functions of the application. The Data Requirements Document is more concerned with the data to be processed by the functions. This is important, but usually not as important as whether the functions provide adequate security. For evaluations where the Data Requirements Document plays a major role, this task name can be changed from functional evaluation to functional and data requirements evaluation.

2.3.3 Control Existence Determination

The fact that functions are described in a document or discussed in an interview does not prove that they have been implemented. Basic evaluations require assurance that security function controls exist. The existence of most physical and administrative controls can be determined via visual inspection. For controls internal to the computer, testing is needed. Such testing does not gather significant evidence towards determining how well controls work since that is beyond the scope of a basic evaluation. The intent is simply to verify that the functions exist. On the other hand, quality must be kept in mind in the event there are fundamental shortcomings that call into question the overall effectiveness of the functions. A particularly vulnerable area here is the susceptibility of procedural controls to human errors.

Tests for control existence determination are straightforward. In many cases, a short operational demonstration suffices as shown in Appendix H. For example, the existence of a password function can be determined by attempting to use the application and verifying that a valid password is required. The existence of a grant access function can be determined by verifying that access is not allowed unless explicitly granted (e.g., by the file owner). Black box (external) testing is generally sufficient for control existence determination.

2.3.4 Methodology Review

Control existence determination provides assurance that controls exist. It says nothing about their quality. Even though this is a high-level overview-type evaluation, it is still desirable to gain some assurance that controls are acceptably implemented. The best way to do this without becoming immersed in testing or detailed analysis is to examine the methodology used to develop the application. This step applies regardless of whether the application is currently under development or has long been operational.

Methodology review contributes to a confidence judgment on the extent to which controls are reliably implemented and on the susceptibility of the application to flaws. This review is important since an unreliable development process can create flaws in the product. If review findings suggest that the implementation cannot be relied upon, detailed evaluation is typically required in order to find specific flaws. Specific flaws are far preferable as certification evidence than a simple judgment of low confidence.

Appendix F shows how security-relevant products and reviews are integrated into the development process. More extensive guidance is found in [FIPS73], which is also concerned specifically with the security of sensitive Federal government applications. Many other sources also provide guidance in proper development and reviews [FIPS101] [NBS81] [NBS82-3]. Software evaluation methods can embody and support effective development practices in addition to providing analytic support [NBS82-2]. One such methodology, software quality metrics [NBS83], might eventually be useful in automating portions of the methodology review. The areas of concern in reviewing a development methodology for certification are summarized below. Several of the areas also apply to security products obtained from vendors.

1. *Documentation.* Is there current, complete, and acceptable-quality documentation? This applies to both development and operational documentation.

2. *Objectives.* Was security explicitly stated and treated as an objective, with an appropriate amount of emphasis for the situation? Were security requirements defined?

3. *Project Control.* Was development well controlled? Were independent reviews and testing performed and did they consider security? Was an effective change control program used?

4. *Tools and Techniques.* Were structured design techniques used (e.g., modularization, formal specifications)? Were established programming practices and standards used (e.g., high order languages, structured walk-throughs)?

5. *Resources.* How experienced in security were the people who developed the application? What were the sensitivity levels or clearances associated with their positions?

2.4 Detailed Evaluation

In many cases a basic evaluation does not provide sufficient evidence for certification. Examples are cases where (1) basic evaluation reveals problems that require further analysis (2) the application has a high degree of sensitivity, or (3) primary security safeguards are embodied in detailed internal functions that are not visible or suitable for examination at the basic evaluation level. These situations require detailed evaluations to obtain additional evidence and increased confidence in evaluation judgments.

Detailed evaluations involve analysis of the *quality* of security safeguards. Primary tasks are examinations of the application from three points of view:

1. Functional Operation (Do controls function properly?)

2. Performance (Do controls satisfy performance criteria?)

3. Penetration Resistance (How readily can controls be broken or circumvented?)

These points of view are discussed at length below in Section 2.4.1. They apply to the evaluation of controls at a deeper level than appropriate for basic evaluation. Whereas the tasks in a basic evaluation are necessary for all certifications, those in detailed evaluation are performed as needed. Detailed evaluation consists of a collection of approaches. Selection of which to use depends primarily on the threats and exposures of concern, rather than on the general characteristics or overall sensitivity of the application. To illustrate, if the primary concern is to protect secrets from an external penetrator, penetration resistance is stressed. Agencies providing a critical service might stress system availability (a performance attribute) rather than functional operational or penetration resistance. An accounts-receivable application might place emphasis on functional operation. Ideally each of these "points of view" has a corresponding set of requirements or acceptance criteria against which to perform the evaluation [NEUG82].

If several points of view are to be employed, it may not be necessary to complete analysis in one area before beginning the next. In many cases, however, these points of view are not mutually exclusive and form a hierarchy that needs to be done sequentially (i.e., functional operation, performance, and penetration resistance—in that order). In all cases, each can be pursued to varying depths of thoroughness, depending on the perceived security problems. The utility of the three points of view is in organizing detailed evaluation work.

The final topic covered in this section is *detailed focusing*. Unlike basic evaluations, which need to be complete for all security safeguards down through the functional level, detailed evaluations can rarely be complete. There are simply too many controls and combinations of controls to examine every one in detail, except in extreme cases. Detailed evaluations need to be focused. Decisions of where to focus detailed evaluation attention can be among the most important decisions associated with an evaluation. Two strategies for such focusing are discussed below in Section 2.4.2.

2.4.1 Three Points of View

2.4.1.1 Functional Operation—Functional operation is the point of view most often emphasized in detailed evaluation since it assesses protection against human errors and casual attempts to misuse the application. Evaluations of functional operation assess whether controls acceptably perform their required functions. Although testing is the primary technique used in evaluating functional operation, other validation and verification techniques [NBS81] [FIPS101] must also be used, particularly to provide adequate analysis and review in early phases of the application life cycle. To the extent possible, certification requirements for testing are satisfied by the testing and verification performed routinely during development and operation. It is not practical for certification to duplicate these activities. On the other hand, it is desirable for certification needs to influence them. Where routine testing and verification does not provide sufficient assurance for certification, additional testing, focusing on security control function operation, must be added to satisfy certification needs. Tests for functional operation examine areas such as the following.

1. Control operation (e.g., do controls work?).

2. Parameter checking (e.g., are invalid or improbable parameters detected and properly handled?).

3. Common error conditions (e.g., are invalid or out-of-sequence commands detected and properly handled?).

4. Control monitoring (e.g., are security events such as errors and file accesses properly recorded; are performance measurements of characteristics such as resource utilization and response time properly recorded?)

5. Control management (e.g., do procedures for changing security tables work?).

To illustrate this testing, consider several of the tests needed to examine control operation of a password function:

1. Test whether access without a password is disallowed.

2. Test whether valid passwords are accepted and invalid passwords are rejected.

3. Test the interface between the password function and the access authorization function by testing whether access is properly allowed or disallowed. For example, verify that valid passwords allow proper access and do not allow improper access, and that invalid passwords result in proper access restriction.

4. Test whether the system responds correctly to multiple invalid passwords.

5. Test whether system-initiated reauthentication functions correctly.

Note that these tests are illustrative. Actual tests depend on the detailed characteristics of the specific function involved, and cannot be fully derived from a generic list such as this.

Functional operation includes the application's resistance to external errors. Therefore the test areas of primary interest include those interfaces across which errors might propagate:

1. man-man (e.g., operator messages)

2. man-system (e.g., commands, procedures)

3. system-system (e.g., intersystem dialogue)

4. process-system (e.g., calls)

5. process-process (e.g., interprocess calls)

Most test tools and methods are of use here, since functional operation is the application characteristic most often tested. Testing can be either external ("black box" or acceptance testing) or internal ("white box," program testing, integration testing) depending upon the interfaces of concern. Testing can be performed by the evaluation team (see Section 1.3.4), by an agency test and evaluation group, by the developer, by the user, or by combinations of these groups. As noted above, to the extent possible, certification personnel rely on the evidence from normal development testing for certification evidence. One promising approach to internal testing for certification is the establishment of test "measures of coverage" criteria and the use of automated tools to measure actual test coverage. This is discussed in [NBS83], together with other aspects of security testing, and in [FIPS101].

When performed independently of the development team, internal testing can present major logistic problems. It can require stub and call routines, test data collection instrumentation, test data itself, and many other forms of support software. It can also require a full software development capability, tailored to the specific operating system and the particular application. The ideal

solution is use of the facilities on which the application was originally developed. If this is not possible, careful planning is needed if major difficulties with internal testing are to be avoided.

Several "through the computer" audit techniques are applicable to functional operation testing. For example, the Test Deck method and Base Case System Evaluation, both of which are common forms of testing, are clearly applicable. Integrated Test Facility or Parallel Simulation techniques might also be of use in operational applications. Where financial controls are of concern, EDP audit experience can be particularly useful.

Some audit techniques are applicable to integrity issues rather than function operation. Techniques used to monitor production activity such as Transaction Selection or use of a System Control Audit Review File (SCARF) are applicable to operational audits or security monitoring. Data reliability assessment techniques (e.g., those contained in some Generalized Audit Software that foot and balance files) play an additional role in certification. As noted in Section 1.5.5, certification is, however, primarily focused on examining the procedures, not verifying the data ("substantive" testing). All of the audit techniques mentioned here are described in [IIA77-1].

Besides testing, there are other security evaluation tools and techniques that can be of use in examining functional operation. For example, software tools for program analysis [NBS83] [GAO81-2, p. 255] can be helpful in documentation analysis. Matrices as in [MAI76] can suggest ideas for test cases and scenarios. Checklists have utility in providing quick training as well as suggesting ideas for tests. This value will increase as more varied checklists become available to meet particular needs. For example, it can be useful, for purposes of reference and to ensure completeness, to have checklists of assets, exposures, policies, policy alternatives and issues, environmental characteristics, threats, threat and asset characteristics, factors influencing threat frequency, controls, control interactions, flaw categories, penetration approaches, tests, and so forth.

Formal verification is a technique that may be used during a detailed evaluation. Formal verification offers the hope of being able to mathematically "prove" that a functional design abides by a few simple security rules, and that lower levels of abstraction are consistent with the proven higher-level design. Formal verification is still primarily a research area, and is not widely used outside of some specialized DoD projects. Nevertheless, formal techniques are being used to develop and to verify the functional operation of weapons control, space-vehicle control, and other extremely critical applications. Such techniques might soon play a wider role. More research is needed, however, before formal verification can play a major role in a typical evaluation.

2.4.1.2 Performance—There is much more to the quality of safeguards than proper functional operation. A number of qualitative factors are listed under the general heading of performance, which is the second area of concern in detailed evaluation. These are availability, survivability, accuracy, response time, and throughput. They can be applied either to individual controls or to entire applications. Each is illustrated with an example.

1. *Availability.* What proportion of time is the application or control available to perform critical or full services? Availability incorporates many aspects of reliability, redundancy, and maintainability. It is often more important than accuracy. It is especially relevant to applications with denial of service exposures as primary concerns (e.g., air traffic control, automatic funds disbursement, production control). Security controls usually require higher availability than other portions of an application.

2. *Survivability.* How well does the application or control withstand major failures or natural disasters? "Withstand" includes the support of emergency operations during the failure, backup operations afterwards, and recovery actions to return to normal operation [FIPS87]. Major failures are those more severe than the minor or transient failures associated with availability. Survivability and availability overlap where failures are irreparable, as in space systems.

3. *Accuracy.* How accurate is the application or control? Accuracy encompasses the number, frequency, and significance of errors. Controls for which accuracy measures are especially

applicable are identity verification techniques (e.g., using signature, voice) and communication line error handling techniques [FIPS83]. Research in software quality metrics is applicable here.

4. *Response Time.* Are response times acceptable? Slow control response time can entice users to bypass the control. Examples of controls for which response time is critical are passwords (especially in distributed networks) and identity verification techniques. Response time can also be critical for control management, as in the dynamic modification of security tables. It is useful in evaluating response time to assess the impact of varying levels of degradation.

5. *Throughput.* Does the application or control support required usage capacities? Capacity includes the peak and average loading of such things as users and service requests. This can involve the analysis of performance ratios such as total users versus response time.

Testing is the best way to evaluate performance, with specific tests needed for each of the above factors that are of concern. A useful technique here is "stress" testing. This can involve using large numbers of users and requests, using large amounts of background activity, or employing maximal resources to attain conditions of operational stress. Functional operation might also be examined under these conditions, since stress loading often interferes with normal processing.

Stress testing is also used in a more directed fashion by attempting to exhaust quota limits for specific resources such as buffers, queues, tables, and ports. These resources might be external or internal to the application and might support application functions such as jobs, transactions, and sessions. This directed stress testing is especially useful in evaluating protection against denial of service threats.

2.4.1.3 Penetration Resistance—The final area of concern in detailed evaluation is penetration resistance. The task here is to assess resistance against the breaking or circumventing of controls, where resistance is the extent to which the application and controls must block or delay attacks. Cryptanalysis is an example of a technique for breaking a particular control, encryption. Creating and using a fraudulent log-on utility to discover passwords is an example of control circumvention. The nature of the evaluation activity here differs widely depending on whether the penetrators of concern are users, operators, application programmers, system programmers, managers, or external personnel. In addition, the notion of penetration resistance applies not only to attacks against data, but also to attacks against physical assets and performance.

Assessment of penetration resistance can be the most technically complex of the detailed evaluation categories. It is best done to establish confidence in security safeguards. It can also be done to find and correct flaws, although recent history has shown the inadequacy of "find and fix" as an approach for achieving security. In both cases it:

1. provides an assessment of an application's penetration resistance;

2. helps to determine the difficulties involved in actually exploiting flaws; and

3. provides a clear demonstration of flaw exploitability (since it might not be clear from analysis whether, say, an asynchronous timing flaw can be exploited).

It should not be inferred that this Guideline is recommending penetration testing as a standard technique. It is presented here as an optional subtask. Nevertheless, penetration resistance evaluation is different in kind from other forms of evaluation and can play an important role in certification.

The objective of penetration-resistance evaluation is to identify externally exploitable flaws in internal security functions and the interfaces to them. Following are illustrative areas for this detailed examination (taken primarily from [IBM76, p. 106]):

1. complex interfaces

2. change control process

3. limits and prohibitions

4. error handling

5. side effects

6. dependencies

7. design modifications/extensions

8. control of security descriptors

9. execution chain of security services

10. access to residual information

There are several approaches to structure software penetration resistance evaluation. These involve (1) searching for flaws that fall into certain categories or patterns [HOL74, LIN75, NEU78, WEBB76]; or (2) hypothesizing generic flaws and then determining if they exist [LIN75, WEI73]. Although these methods apply to the evaluation of software, similar approaches are available to evaluate hardware [AKE80] and physical and administrative controls.

When employed to assess complex objects such as large software operating systems, penetration-resistance evaluation can typically employ a team of two or three people for from two to four months. Beyond this time frame, there is a point of diminishing returns, since the object of the effort is not to find all flaws but to provide an assessment of the application's penetration resistance.

2.4.2 Detailed Focusing Strategies

It is rarely feasible or desirable, even in a detailed evaluation, to examine everything. Two strategies are presented for focusing on small portions of the security picture when evaluating from some or all of the three points of view discussed above. One is based on security relevant components and the other on situational analysis.

2.4.2.1 Security Components—This focusing strategy is based on four components relevant to ADP security: assets, exposures, threats, and controls. All of the components will have already been considered in the basic evaluation or in a risk analysis. The current activity involves a detailed view. It can use basic evaluation or risk analysis data where suitable, and extensions of such data, as needed, for the analysis reports.

The list of sample analysis reports discussed below for each component could be expanded. It illustrates that a variety of reports might be needed. The questions of how many and which types depends upon evaluation findings.

1. *Assets.* Assets are the tangible and intangible resources of an entity. The evaluation issue here is: What should be protected? It might be useful to examine assets (data, files, physical resources) in detail along with their relevant attributes (amount, value, use, characteristics). Most-likely targets can be identifed in this way. A variety of specific tasks might be needed. For example, an asset value analysis determines how the value differs among users and potential attackers; an asset exploitation analysis examines different ways to use an asset for illicit gain (e.g., as "insider" stock information).

2. *Threats.* Threats are possible events with the potential to cause loss or harm. "What are assets being protected against?" is the evaluation issue. In examining threats, it is important to distinguish among accidental, intentional, and natural threats. Intentional threats

can be the most complex. An example of an analysis task for intentional threats is to identify perpetrator classes (programmers, operators, users) based on knowledge, skills, and access privileges. The Relative Impact Measure (RIM) approach to security evaluation can be used for this purpose [NIE80]. Perpetrator motivation, resources, opportunity, and organization are all considered in such a process. An extensive list of generic threats is in [FIPS65, Appendix A].

Another useful analysis examines the factors affecting threat frequency. Threat frequency depends on such factors as (1) threat magnitudes, (2) assets and whether their loss is full or partial, (3) relevant exposures, (4) existing controls, and (5) expected gain on the part of the perpetrator.

The nature of the threats can influence evaluation methods used. For example, a standard evaluation technique is to review samples of source code to determine compliance with established programming practices and to look for security flaws. If the threat is a malicious developer, however, and the intent is to find "malicious" software, the assembled object code is reviewed rather than the source code or specifications, since the malicious steps will not be documented at the higher levels.

3. *Exposures.* Exposures are forms of possible loss or harm. Here the evaluation issue is: What might happen to assets if a threat (internal failure, human error, attack, natural disaster) is realized? Examples of exposures are disclosure violations, erroneous decisions, and fraud. [NBS83] discusses different exposure categories. An example of an exposure analysis is the examination of the impact of a particular exposure (e.g., greatly increased response time for a service, caused by the malicious actions of a competitor or disgruntled user). Much exposure analysis focuses on identifying areas where exposures are greatest. The question of which exposure types represent the areas of greatest loss or harm can have a major influence on detailed evaluation activities. For example, if integrity or accuracy is the primary concern, evaluation emphasis focuses on the basic application processing; if disclosure is the primary concern, evaluation emphasis falls on those functions and interfaces associated with disclosure protection.

4. *Controls.* Controls are measures that protect against loss or harm. The evaluation issue here is: How effective are security safeguards in reducing exposures? Evaluation tasks here often focus on controls embodied in specific application functions and procedures. Examples of evaluation tasks include control analysis (to examine a particular control in depth and determine its vulnerabilities and severity); work-factor analysis (to determine actual difficulties in exploiting control weaknesses); and countermeasure tradeoff analysis (to examine alternative ways to implement a control—this is often necessary in order to recommend corrective actions).

2.4.2.2 Situational Analysis—One forbidding and constraining aspect of computer security evaluation is the complexity of an application and its protective safeguards. This limits not only the percentage of the application that can be examined but also the degree of understanding attainable for those portions that are examined. These limitations represent an important and fundamental problem of security evaluation: How does one make a confident judgment based on incomplete information and partial understanding? A solution to this dilemma is the use of situational analysis. Two forms of situational analysis are discussed: the analysis of attack scenarios and the analysis of transaction flows. Both are used to complement the high-level "completeness" of a basic evaluation with detailed, well-understood examples and can focus on particular aspects of the application that are of concern (functional operation, performance, and/or penetration resistance).

An attack scenario is a synopsis of a projected course of events associated with the realization of a threat. It encompasses the four security components discussed above—threat, control, asset, and exposure—interwoven with the specific functions, procedures, and products of the application. An example of an attack scenario is a step-by-step description of a penetration, describing penetrator planning and activities, the vulnerability exploited, the asset involved, and the resulting exposure.

A transaction flow is a sequence of events involved in the processing of a transaction, where a transaction is typically an event or task of significance to and visible to the user. Transaction flow analysis is commonly used in EDP auditing [AAC78, IIA77-1] and is discussed in [NBS83]. If the application as a whole contains only a small set of transactions, transaction flow analysis might be a sufficient vehicle in itself for the detailed evaluation. A basic evaluation is still needed, however.

The idea underlying situational analysis is to focus attention on a manageable set of individual situations that can be carefully examined and thoroughly understood. This makes the resulting analysis more meaningful for several reasons.

1. It places threats, controls, assets, and exposures in context with respect both to each other and to application functions. This allows the evaluation to properly consider interdependencies, such as those among controls, and presents a balanced, realistic picture. If a detailed evaluation decomposes security components into constituent parts, a situational analysis pieces these together again into a coherent whole.

2. It emphasizes the objectives being served by control(s), and allows safeguards to be evaluated based on these objectives.

The increased understanding that can result from use of situational analysis, as well as its illustrative value, make it an important tool for use in conducting and presenting detailed evaluations.

2.5 Report of Findings

This section is concerned with the security evaluation report that is prepared for the Accrediting Official. The security evaluation report is the primary product of certification. It contains technical security recommendations for the application and is the main basis for the accreditation decision.

2.5.1 Integrating the Report

Figure 2-4 shows an example of how evaluation findings might be integrated into the security evaluation report. The evaluation work is partitioned into three areas, (1) application software and administrative and procedural safeguards, (2) physical security, and (3) operating systems and hardware. (Section 2.1.2.3 includes discussion of partitioning.) Evaluation needs in the operating systems and hardware area are satisfied externally, as might be the case if using a product evaluation from the DoD Computer Security Center [DoD83]. Most of the internal work is in the area of application software and administrative and procedural safeguards. Here there could be detailed evaluations of several partition areas that might have problems or high sensitivity. The detailed findings are combined with basic evaluation findings, and all of the findings are integrated into the security evaluation report. It is preferable to integrate findings from different evaluation areas into one final report rather than to deliver several security evaluation reports to the Accreditor, since the safeguards in each area can have complex interrelationships that require a technical interpretation.

2.5.2 Transmitting the Report

The security evaluation report is prepared under the direction of the Application Certification Manager, signed, dated, and delivered to the Accrediting Official(s). It might also be reviewed and approved by the overall agency Certification Program Manager to ensure compliance with agency standards. Typically there is a formal transmittal letter to the Accrediting Official(s) that

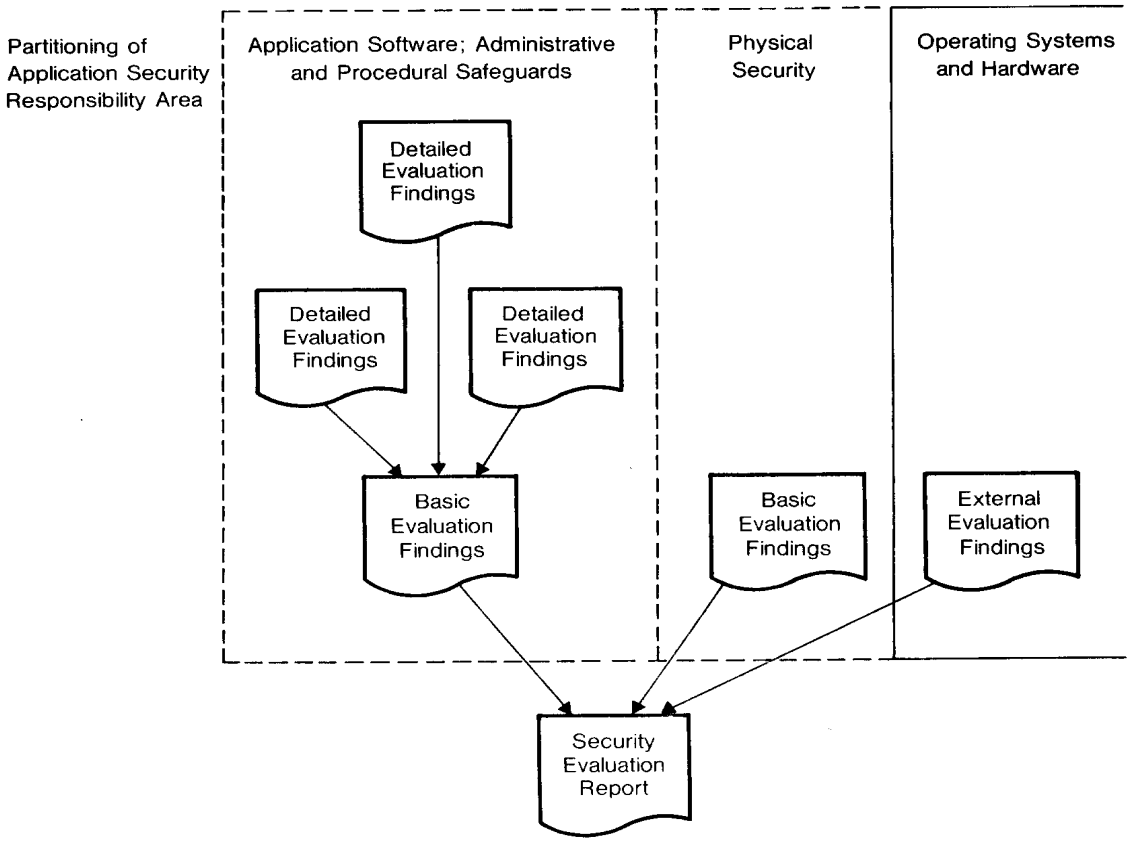


Figure 2-4. *Sample documentation flow for certification findings*

describes the contents of the report and recommends signing of the accreditation statement. Figure 2-5 shows a sample transmittal letter. It includes an official certification statement in order to comply with [OMB78].

2.5.3 Sample Outline of Report

A sample outline of the security evaluation report is shown in Figure 2–6. Each section of the outline is briefly described below.

1. *Introduction and Summary.* This section briefly describes the application and summarizes evaluation findings and recommendations.

2. *Background.* This section provides contextual information for the Accrediting Official. One important item is the security standards or policies that were applied. Another is a list of the general functional characteristics of the application that generically influence its certifiability (e.g., the presence or absence of user programming). Application boundaries are defined, along with security assumptions about areas outside the boundaries.

3. *Major Findings.* The first portion of this section summarizes the controls that are in place and their general roles in protecting assets against threats and preventing exposures. This is important in maintaining perspective, and emphasizes those areas where safeguards are acceptable.

Subject: Certification of _____ [1] _____.
Reference computer security policies _____ [2] _____. This Certification has been performed because _____ [3] _____.

Attached are the findings from security certification evaluation of _____ [1] _____. The security evaluation report summarizes findings and presents recommendations. Attached to the report is a proposed accreditation statement for your review and signature.

Based on the report and my judgment, I hereby certify (with the exceptions or clarifications noted below) [4] "that _____ [1] _____ meets the documented and approved security specifications, meets all applicable Federal policies, regulations, and standards, and that the results of [testing] demonstrate that the security provisions are adequate." [5]

(exceptions or clarifications)

In addition, weighing the remaining residual risks against operational requirements, I recommend that you authorize (continued) operation of _____ [1] _____ (under the following restrictions):

(restrictions)

(I further recommend that you authorize initiation of the following corrective actions.)

(corrective actions)

_____
Signature and Date

[1] Name of the application being certified.

[2] OMB A-71, TM1 and other applicable policies.

[3] Reasons include the following: (1) initial development has been completed, (2) changes have been made, (3) requirements have changed, (4) a required threshold of time has been reached, (5) a major violation has occurred, and (6) audit or evaluation findings question a previous certification.

[4] Parentheses indicate portions of the letter that are not required in some situations.

[5] Quotation from OMB A-71, TM1. The quotation marks are explanatory and, along with the editorial brackets, are not included in the actual letter.

Figure 2-5. *Sample transmittal letter for security evaluation report*

The second portion summarizes major vulnerabilities. Vulnerabilities described in the report are divided into two categories: proposed residual vulnerabilities and proposed vulnerabilities requiring correction. This format serves as both a summary of findings and a recommendation of which vulnerabilities to accept and which to correct. Authority to approve the recommendations resides with the Accrediting Official.

4. *Recommended Corrective Actions.* Here corrective actions, together with anticipated costs and impacts, are recommended and prioritized. Responsibility for making the corrections might be proposed. Also criteria must be established for evaluating the corrections. This section must be sufficiently complete to give the Accrediting Official a clear understanding of the implications of either accepting or correcting vulnerabilities.

Since sensitive applications are typically important to agency operations, most flaws will not be severe enough to remove an operational application from service although some

1. INTRODUCTION AND SUMMARY

2. BACKGROUND

3. MAJOR FINDINGS

   3.1 General Control Posture

   3.2 Vulnerabilities

4. RECOMMENDED CORRECTIVE ACTIONS

5. CERTIFICATION PROCESS

Attachment A Proposed Accreditation Statement

Attachment B (etc.) Detailed Evaluation Report(s)

Figure 2-6. *Sample outline for a security evaluation report*

restrictions may need to be implemented immediately. It is likely that a serious flaw will be severe enough to delay implementation of a change or an application under development.

Other than removing an application from service or delaying its implementation, there are many intermediate accreditation alternatives available. The most common is to withhold accreditation pending completion of corrections. Many types of operational restrictions are also possible. Examples follow.

a. Adding procedural security controls. Restricting use of the application to sites that have compensating controls.

b. Restricting the application to process only nonsensitive or minimally-sensitive data.

c. Removing especially vulnerable application functions or components. In a network environment a particularly weak node might be excluded from the network.

d. Restricting users to only those with approved access to all data being processed or to those with a sufficient "clearance" based on an investigation.

e. Restricting use of the application to non-critical situations where errors or failures are less severe.

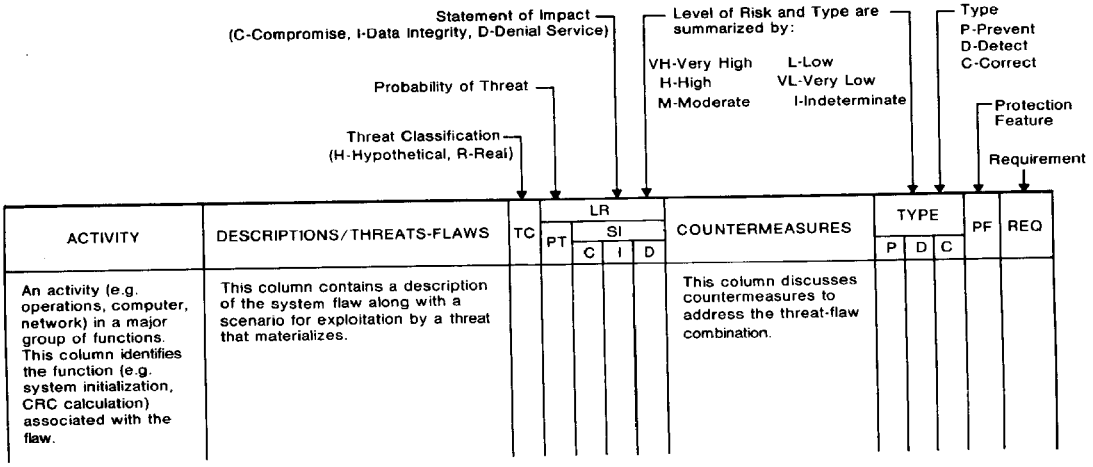
f. Removing dial-up access (thus relying more on physical security).

g. Granting conditional accreditation for a "shakedown" period before full trust is granted.

5. *Certification Process.* This section summarizes the work performed in the certification process. Its purpose is to enable the Accrediting Official to determine the confidence that can be placed in the findings. It might also be useful to include the Application Certification Plan as an attachment to the report.

Attachment A. Proposed Accreditation Statement. This is a critical part of the report. It summarizes recommended actions and is prepared for the Accrediting Official's signature. A sample statement is shown and discussed in Section 2.6. Judgments and recommendations embodied in the statement are subject to approval by the Accrediting Official.

Attachment B. Evaluation Report(s). These describe the full set of findings, not just major ones. It can be useful, especially if separate evaluation teams are participating, to use standard forms to present basic and detailed findings. An example of such a standard form is shown in Figure 2-7. Most columns are self-explanatory. The threat classification column permits distinction between flaws that exist and those which are suspected but for which no positive evidence can be found (e.g., malicious software, unknown operating system loopholes). Columns are available to reference applicable protection features and requirements.



Adapted from work done by System Development Corporation for the Defense Communications Agency.

Figure 2-7. *Sample vulnerability chart*

2.5.4 Characteristics Of The Report

Since the Accrediting Official is a high-level official, usually with a busy schedule, the security evaluation report is kept brief. The report must be accurate, meaningful, and constructive, as follows:

1. *Accurate.* All judgments must be supported. Quantitative ratings are to be avoided unless founded on demonstrably accurate data. Ratings that have a large uncertainty are often interpreted the same as accurate ones and this leads to a high potential for misunderstanding. Where ratings of some form are used, they must explicitly reflect assumptions, conditions, and variances.

2. *Meaningful.* The content and form must be understandable to the Accrediting Official. For example, it is common to separate exposures into categories (e.g., disclosure, modification, denial of service, and destruction). This breakdown might not be meaningful to a high-level manager such as the Accreditor. It might be preferable instead to orient the presentation around exposures such as fraud, competitive disadvantage, agency embarrassment, statutory sanctions, and so forth.

3. *Constructive.* Positive evidence must be summarized. Most applications are doing much more right than they are wrong. The evidence as a whole must be kept in balance. Security evaluations often report only those things that are wrong. That does not present a fair picture of all the available evidence. Similarly, recommendations must be realistic. It is not realistic to suggest that a critical application that has been running for years be shut down because of a single flaw. A more constructive suggestion is to adopt added precautionary procedures, and to begin planning on upgrades or a new version of the application.

The report is a very sensitive document; it represents a vulnerability, since it is subject to loss or abuse. Some agencies might need charters (as have been approved for criminal justice or national security data) to protect this information from Freedom of Information Act inquiries. In some cases it might be desirable to destroy the reports after they have been used (agency legal staff should be consulted here).

2.5.5 Coordinating the Report

For the report to be most effective, it typically needs to be coordinated through appropriate agency offices. (In some cases it is not desirable to coordinate the *entire* report since this might unnecessarily promulgate vulnerability information.) Offices responsible for corrective actions as well as others who might be affected are included. This increases the likelihood that recommendations will be ultimately accepted and implemented. It is possible that this coordination will result in changes to the report. This should not be viewed as threatening the report's objectivity but as helping to ensure its validity and eventual acceptance. Similar coordination is usually performed by auditors in reporting audit findings. Procedures should be established for airing and, if possible, resolving disagreements.

2.6 Accreditation

The Accrediting Official is responsible for evaluating the certification evidence, deciding on the acceptability of application security safeguards, approving corrective actions, signing the accreditation statement, and ensuring that corrective actions are acomplished. The products around which these actions are focused are the security evaluation report and the accreditation statement.

2.6.1 Using the Security Evaluation Report

The Security Evaluation Report contains evidence to support not only the evaluation findings, but also the evaluation process itself. Evaluation findings are used to assess the application while the evidence on the quality of the evaluation process can be used to assess the quality of that process. For each of the following issues the Certification Program Manager and/or Application Certification Manager might examine both the sufficiency of what was done and where improvements can be made in the evaluation activity. Some questions they might ask are:

1. Auditability. Is there a record of the evaluation that allows the work and the process to be assessed?

2. Data. How accurate are the data that were obtained?

3. Tools. Were tools effective and efficient?

4. Techniques. Were the methods used effective and efficient?

The Accrediting Official should also be able to use the report as a guide for formulating questions.

The security evaluation report is the primary evidence for the accreditation decision although there might be other evidence, such as written or verbal reports from other agency offices. The task for the Accrediting Official is to evaluate the report and its findings and recommendations. Figure 2-8 contains questions that might be asked by the Accrediting Official in assessing the report.

If the Accrediting Official decides to add security safeguards, he must approve required corrective actions and allocate sufficient resources to make the corrections. The Accreditor must also ensure that all parties understand the corrections and their roles in making them. The Accreditor might assume or delegate responsibility for follow-up examinations. It must be stressed that the identification and implementation of corrective actions are important and difficult management activities.