82-01-17 The Software Certification Process
Craig A. Schiller

Payoff
This article, the second in a two-part series, examines how to determine the type and degree of certification required for specific products. A checklist is provided to assist in this evaluation. The article also describes the types of testing that should be considered, project management strategies, and how to obtain organizational approval for certification.

Problems Addressed
The certification process for externally developed software begins with a determination of the type and degree of certification required. The types of certification range from none to extensive certification, depending on the nature of potential losses toward which this product may contribute. The purpose of the determination is identifying areas of concern and deciding the nature and extent of certification tasks, including testing. The Appendix, discussed in the following sections, contains a checklist that can be used to determine the type and degree of certification that is required.

General Use Concerns
Part 1 of the checklist addresses some general areas of concern in all products and systems (except operating systems) to determine the need for general certification. Generally, if any of the products or applications to be run under the operating system requires certification, then the operating system also requires certification. A list of the general, externally developed software concerns is contained in the checklist.

Applications in which the general use may cause a loss of the type listed in Part 1, Section A of the Appendix should require as exhaustive a certification as can be afforded. If testing is recommended, then the information security professional should indicate which of the three properties (e.g., integrity, availability, or confidentiality) may contribute to the potential loss. If testing is not recommended, the next item should be addressed. These properties drive the inclusion or exclusion of some types of certification tests. The information security professional should then describe the scenarios that may result in this application contributing significantly to the loss. These scenarios are used to develop test cases.

In answering Part 1, Section B of the Appendix, the security professional should first answer whether general use of the product could cause or significantly contribute to various types of loss. If so, the information security professional should attempt to estimate the range of potential severity and the likely severity. Either the likely severity or the range of potential severity may motivate the information security professional to recommend certification testing. The severity estimates only need to be precise enough for the security professional to decide the extent of certification to be recommended. If certification is recommended, then the security professional should indicate which of the three properties, integrity, availability, and confidentiality may contribute to the potential loss; he or she should also describe the potential scenarios that might result in significant loss.

Part 1, Section C identifies specific vulnerabilities associated with the use of a product. For example, a product may be intended for normal users but require privileged access to execute.
Intended Use Concerns

In Part 2 of the Appendix, the intended use of the product is addressed. Often it is the intended use of the software product, rather than the general use, that is the motivation for certification of an externally developed software package. For example, a database package may not need to be certified on the basis of general use or potential loss concerns; however, management may require certification of the same database package if the package is used to support applications supporting data protected by privacy laws. The description of the intended use should be in terms of the operational uses of the package and system. The information security professional should translate these terms into the information security concerns of integrity, availability, and confidentiality. In Part 2, Section A, the preceding potential-loss-oriented questions are asked about each intended use of the product, and additional concerns are addressed. (The use of the checklist in Part 2 follows the processes already described for Part 1.) In Part 2, Section B, the listed potential losses require a judgement about the potential severity of loss. Section C identifies specific vulnerabilities that the intended use of a product may cause. For example, in general, a product may require only normal user access. However, to use the product with system data may require that the product be run with privileged access. Section D covers data-oriented concerns. Data is divided into three types: critical, sensitive, and confidential, and the requirements for each are evaluated. Critical data is data that has an availability requirement if this data is not available when needed, loss can occur. Sensitive data is data that has an integrity requirement if sensitive data is not from the right source or is corrupted, loss can occur. Confidential data is data that must be protected from unauthorized and accidental disclosure to prevent loss.

An organization should maintain one set of Part 1 information about the general use of the product and one set of Part 2 information for each important intended use of the product. The data should be available for the use of system developers in decisions regarding product purchase and in selecting externally developed software for use in systems and other products.

Each organization should develop a method for using the checklist. Three levels of certification are recommended: documentation, acceptance, and extensive. The higher levels include the assurance measures of the levels that precede them.

The extensive level would be used for products with any extreme severity loss concerns. Extensive certification should include certification testing similar to the level of testing that would occur for internally developed applications with potentially severe losses. A management decision may be needed to determine how much can be spent to avoid these unacceptable losses. Extensive certification may include reverse engineering requirements that may be used to improve testing effectiveness by permitting functional and security requirements testing. It should include purchasing the source code (though not necessarily for modification) if available. If the source code is available, then glass box testing can be used to increase the testing effectiveness.

The acceptance level certification would be used for products with no extreme severity loss concerns and with potentially moderate severity losses. Tests should be performed, but the effort to make additional testing possible is probably not cost effective. Testing at this level should include feature confirmation tests, acceptance testing, and simulation scenario testing.

The documentation level certification would be used for all other systems-related products. The documentation level uses such general tests as virus testing, vulnerability testing, code reviews, demonstrations, and such mechanisms as digital signatures and cryptographic checksums (when they are provided by the developer). In other cases, the assurances come from documented reviews of the product, the availability of technical support lines, and document-based assurances collected about the systems developer and product.
Gathering Identification and Assurance Information

The first category of information to collect regarding systems products concerns vendor and product identification. For the vendor, the collected information should include the name of the company, its address, points of contact, phone numbers for technical support, sales, and other information relevant to the type of systems. For the product, identifying information should include the name of the product, current version, points of contact, and phone numbers for product-specific technical support, maintenance support, and license support.

Information regarding the number of years in related businesses, the number of employees, other related products, and the vendor's reputation may be used to evaluate organizational process assurance. Although it is not specifically about organizational processes, this information helps the information security professional judge the stability of the developer's business. This information should be supplemented with information regarding the results of International Standards Organization 9000 certifications and Software Engineering Institute evaluations and assessments.

Product-specific information may be gathered to build an opinion about product assurances. Data regarding the success and maturity of the product should be collected. This may include the product age, the number of copies sold, the frequency of updates for major versions and for short-term software error fixes, and the reputation of the product. It may be useful to locate opinions of user groups, user references, independent product reviews, and the results of evaluations against standards. The information security professional should obtain the current list of known deficiencies from the manufacturer and known vulnerabilities from advisories such as those released by the Computer Emergency Response Team (CERT), the Computer Incident Advisory Council (CIAC), and the NASA Incident Response Committee (NASIRC). ISO 9000 certification data and SEI evaluation and assessment results may provide evidence of the developer's product assurance measures. In the product-specific information section, it should be noted whether source code is available and under what conditions and restrictions it is available. The primary and secondary languages used to develop the product and the platforms on which it is expected to run should be indicated. The information security professional should document all interoperability standards supported.

Information about requirements for Certified Information Systems Security Professional certification for key positions in the product developer's organization should be gathered. For products with potentially severe losses, a statement should be obtained from the product developer describing roles, responsibilities, clearances required, internal controls, guarantees of separation of duty, and key position qualifications and credentials. Information about financial stability discussed under financial assurances can also provide support for this opinion.

A description of the physical environment of the developer may only be necessary if the product has an availability need for ensured developer emergency response (e.g., two-hour response to emergency deficiency report). The description of the physical environment helps build confidence that the developer has adequate protection from physical environment threats that might interfere with emergency response. Logical assurances should be collected, including restrictions (e.g., minimum memory required, maximum records supported, restrictions against export) and limitations (e.g., supports only text files, specific device requirements). The logical assurances should include information regarding devices tested, devices not tested but expected to execute, and compatibility issues.

The process of certifying externally developed software is an extension of risk and vulnerability assurances. Data is collected about the assets and about potential losses and severity. This information is used to make decisions about the nature and extent of testing and about recommended safeguards and accepted risks.
The availability of interface specifications and Application Program Interface (APIs) should be documented. Both interface specifications and APIs can make black book testing possible.

**Determining the Nature and Extent of Testing**

Determining the nature and extent of the tests to execute when certifying externally developed software is important to judge the integrity of the product. Types of tests are addressed in the following sections.

**Code Reviews**

If the source code is available, a code review is always recommended. Code reviews can reveal malicious code; they may also result in test recommendations and recommendations for reengineering comments, code, or documentation. The code review provides some assurance of integrity of form and function.

**Digital Signature or Cryptographic Checksum Confirmation Checking**

Using a digital signature from the developer of the product can ensure that the product came from the developer and that it has not changed since the developer signed the product. This assurance allows security practitioners to consider using the developer's tests and quality measures in place of conducting their own evaluations. If the practitioner has confidence in the developer's quality controls and testing process and can confirm that the product came from the developer without corruption, fewer tests may be necessary to certify the product. The cryptographic checksum can ensure that the product has not been changed since the checksum was calculated. It cannot provide irrefutable evidence that the product originated from the developer. However, the information security professional can take procedural actions to confirm that the product came from the developer. The digital signature and cryptographic checksum provide an assurance of the integrity of form. By evaluating the tests and quality controls of the developer, the tests can provide inherited assurance of integrity of function. The digital signature and cryptographic checksum cannot provide assurance of integrity of fit.

**Virus Testing**

At this time, testing virus signatures is only cost effective for microcomputer platforms. Not enough virus signatures are known for UNIX, VAX, or MVS systems for signature checking to be effective. All platforms benefit from performing the assurance testing on a platform that is isolated from the rest of the organization's computing resources. By isolating the externally developed software or data, the chance of discovering or triggering the virus during the certification test process is increased; this reduces the potential for damage to systems.

**Vulnerability Testing**

Vulnerability testing can be divided into two types. System vulnerability testing (e.g., COPS, the Tiger scripts from Texas A&M University, some of the Vanguard products, and SPI from the Air Force Information Warfare Center) and product-specific vulnerability testing, such as manufacturer-suggested tests of bugs found in Computer Emergency Response Team advisories and locally developed tests of bug reports form the manufacturer or user groups.

**Feature Confirmation Testing**

Without feature confirmation testing, an organization relies on marketing hype and product claims when making decisions regarding product use. Some assurance may be derived from independent reviews of the product found in periodicals or publications. Products
with the potential for severe losses should perform independent feature tests. To make feature confirmation testing cost-effective, the security professional can attempt to identify and test those features that may contribute to the losses described in the above surveys. Feature confirmation testing provides assurance of integrity of function, and it can be expanded to ensure integrity of fit. If the product has availability functions (such as backup), then feature confirmation testing may provide some assurance of availability. The same is true of confidentiality.

**Functional Requirement Testing**

Functional requirement testing is a major component of traditional large-system testing. Normally an application must have clearly defined, testable requirements in order for functional requirements testing to be effective. In the case of externally developed software, a software analyst may be able to reverse engineer functional requirements from the application. Alternatively, functional requirements may be derived from the applications that interface with the product. In practice, the system that the product supports may have requirements already allocated to the product. In the case of systems products, the degree of assurance is diminished by the amount of uncertainty introduced by reverse engineering the requirements. When it is performed, this form of testing provides directly applicable assurance of integrity of function. When the product includes availability or confidentiality functions, then this form of testing also provides availability and confidentiality assurance. This form of testing is essential in the case of potential severe loss, and it is useful for other cases of loss when cost effective.

**Glass Box Testing**

Glass box testing is sometimes called path testing. At a minimum, path testing requires testing of enough paths to ensure that every instruction in the routine has been exercised at least once and that every decision (i.e., branch or case statement) has been taken in each possible direction at least once.

It should be noted that traditional path testing does not require that every path be taken during every state in which a module can exist. To accomplish complete coverage of traditional path testing, its proponents say that it should be performed during unit testing, when the size of modules to be tested are at their smallest. Another form of path testing performs tests on transaction paths (rather than code paths) during system test. For many software products, path testing is not possible. Path testing can only be performed on software when the source code is available or when transaction paths can be accurately diagrammed. For products that have potentially severe losses because of availability or integrity, the source code should be acquired from the developer. Acquiring the source code permits better testing (e.g., path testing) to take place, but it may also complicate later deliveries of code if the source is modified. Path testing provides integrity of function assurance.

**Black Box Testing**

In black box testing, the code and processing within the module under test is ignored. Instead, the tests are built based on inputs and expected outputs. Black box testing can be used during unit testing, in integration testing, and in system testing. To use black box testing on systems-related products, the output should be predicted, given known inputs to the product. Black box testing provides integrity of function assurance.

**Security Safeguards Testing**

In security safeguard testing, the testers are trying to ensure that the safeguards that are delivered with the product meet their stated objectives and that the product does not negate or damage existing security safeguards. This focus is necessary because some software development organizations do not include the testing of security requirements in their functional requirements tests. As such, security safeguard testing includes many other
types of testing, although some may have a distinctive security appearance (e.g.,
penetration testing and covert channel analysis). The nature of this test depends on the
nature of the product. The same is true for the forms of assurance provided by these tests.

**Acceptance Testing**
Acceptance testing may come in two varieties, operations acceptance testing and system
acceptance testing. System acceptance is performed by a developer with the customer as a
witness. The customer and the developer have an agreement about the conditions of
acceptance. The system acceptance test demonstrates that the conditions of acceptance have
been met. In the case of operations acceptance, the testing is performed by the operations
community after it has accepted delivery but before it has approved the system for
operational use. This testing is designed by the operations community and provides a user
perspective test. The tests are usually created to determine the system's ability to perform
user tasks. Acceptance testing may provide some integrity of function assurance, but its
main contribution is integrity of fit assurance. Although it is not as exhaustive as other
forms of assurance, the acceptance test is intended to demonstrate the acceptability of the
functions of a system that are most important to the customer or operations community.
Depending on the nature of the product, operations acceptance may also provide availability
or confidentiality assurance. In addition this form of testing may reveal some differences in
expectations of the operations community, the specifiers, the designers, and the coders.
This form of test is useful for most software, especially software without formal
specifications.

**Simulation Scenario Testing**
Simulation scenario testing is used in traditional software development to test the system
under load conditions and to test the system performance in response to specific operational
threats. The tests provide integrity of fit assurances and availability assurances. The
information gathered about potential loss scenarios should be the basis for developing
simulation scenarios to test products. Although it is not as exhaustive as other testing
forms, it does test those specific known scenarios that could result in losses. This is
another useful form of testing for most software, because formal specifications are not
necessary.

**Fault- and Failure-Oriented Testing**
Fault- and failure-oriented testing is used to supplement other traditional forms of testing.
Although most other tests are designed to prove that the system works as intended, the
fault- and failure-oriented tests attempt to prove that the system responds to failures and
faults in an expected manner. It is especially important to determine whether the product
compromises the system's security on fault or failure. Fault- and failure-oriented testing
provide assurance of integrity of fit and may also provide assurance of availability and
confidentiality.

**Test Results**
Testing may result in several different outcomes. On the basis of testing results, an
organization may decide to use a different product than that originally selected. The product
may be surrounded with a shell to make up for deficiencies or to impose restrictions and
monitoring. Changes may be requested from the original developers. If the source code is
available, the organization may rewrite the code to comply with the organization's
development standards, including commentary, prologues, and structure. If future versions
of the product are expected, then the organization should develop a process for applying the
local changes to each subsequent version. It may even be more effective to run a
comparison of the original software from the developer against the new version from the
developer, identifying the changes from the developer and then applying those changes against the organization’s version. The developer may be interested in receiving copies of the local changes. After the actions resulting from testing have been applied and approved (i.e., certified), the identification of the product should be changed to indicate that this is the certified version of the product. Even if the product itself was not changed, it is useful to indicate a change in the state of the product from one with uncertain confidence to one in which risks have been identified and accepted and for which use has been certified.

Managing the Effort Centrally

Initially, the security professional may want to prove the concept of certifying externally developed software to management using a pilot project. However, this process can be made more cost effective if it is managed centrally rather than by project. If the pilot project is run with this goal in mind, then the transition to a centrally managed certification process is easier.

Centrally managing the certification of externally developed software requires an organization to gather information about the general use of the product. The general use information would then be supplemented by information regarding the specific or anticipated use of the product in the organization or system. Future projects need only update the general data, provide new specific use, or locate data(in the central repository) collected by others in the organization that match their own intended use. Gathering this data also highlights the need for standardization of some products and the potential benefit of pursuing a site license for others.

The data collected about systems products and vendors should be easily accessible. As the data is gathered it should be submitted electronically or manually to the organization maintaining the data base. In this way, the inputs can be reviewed and converted to a common scale, when necessary, and duplicate or unauthorized sets of information can be detected and removed.

Selling the Concept to Project Managers

To improve the chances of success, it is recommended that a project manager be found to serve as a champion for the project. This project manager should be respected by other project managers and should already have a favorable opinion of certification of internally developed software. The discussions for and against the practice may not end after the practitioner leaves the discussion. Someone may need to argue for the concept. The presentation to this manager should be informal, but it should contain the same basic elements that the formal presentation to management may contain. The formal presentation should cover the following areas:

- Characterizing the problem and its potential for loss.
- Demonstrating that the problem is local and that important applications are potential victims.
- Describing the plan briefly.
- Listing the benefits.
- Proposing a coordinated pilot program.
- Presenting a schedule for implementation, assuming a successful pilot.
Discussions about costs may derail the presentation before benefits are discussed. The security practitioner should ensure that cost information shows that the cost of the activities are considerably less than the potential for loss. If the cost of activities is not considerably less than the potential for loss, then the selection of activities should be reviewed with a more conservative perspective. The security practitioner should include in calculations the savings that result from finding these systems-related errors before they affect operations.

To sell this concept to project managers, local incidents involving Commercial Off-The-Shelf products should be referenced. These incidents may be found by looking through the organization's list of system deficiency reports or in the security incident data base. In the security data base, occurrences of viruses that were embedded in shrink-wrapped, packaged COTS products may be listed. This helps to combat the shrink-wrap syndrome. In the system deficiency reports data base, the number of reports attributable to products should be counted. If a package is clearly preferable, that product package should be brought to the presentation. The security practitioner may want to refer to well-publicized incidents to illustrate potential threats.

Another aspect of making the problem personal is illustrating the important applications that depend on systems products. The security practitioner can illustrate the impact by listing the important applications by name and the number of products that support each application or the percentage of important applications that rely on systems products. He or she can also describe some of the more credible and damaging loss scenarios.

The description of the plan should be given at a high level. The idea is to sell the concept and not debate its details. The security practitioner should spend more time on the benefits than on the plan. The primary benefits of the concept are higher quality systems and products. Other benefits may include greater standardization of commonly used products and a better understanding of (and response to) the recognized potential loss scenarios.

A high-level description of the pilot program should be presented. The characteristics that make this program the ideal candidate for the pilot should be explained. Project managers may raise objections to the choice of the pilot; they should be encouraged to share their points of view.

The schedule for implementation should begin with a task to collect information about systems and product use in the organization. The organization's systems and products should be ranked for implementation. Using an estimated average time to complete a system project, a schedule for completion for all the organization's systems and products should be obtained. It can be assumed that the schedule will accelerate when approximately one-third of systems products have been certified for a similar use.

A summary of the presentation and why the issue should be addressed should be included. The presentation content and structure should be tailored to the organization.

**Recommended Course of Action**

As more and more organizations turn to externally developed software in an effort to reduce organizational costs, the challenge is to limit the huge potential for loss represented by software developed outside of organizational control. By using the checklist and implementation steps contained in this article, the certification of externally developed software can be ensured.

The author would appreciate hearing about experiences in implementing this or related efforts regarding externally developed software. The author may be contacted at the following address and phone number:

Craig A. Schiller
16018 Fathom Lane
Houston, 77062
Author Biographies

Craig A. Schiller

Craig A. Schiller is a senior analyst for PARANET, Inc. In addition, he is the co-founder of an automated information system security engineering team for NASA's Johnson Space Center, NASA's technology for information security conference, and the Texas Gulf Coast chapter of the ISSA.
CHECKLIST FOR DETERMINING THE TYPE AND DEGREE OF CERTIFICATION REQUIRED FOR EXTERNALLY DEVELOPED SOFTWARE

APPENDIX

Part 1. General Use

1A. Extreme Severity Losses

1. Can general use of this product cause or contribute to loss of life (e.g., use of a software package for monitoring vital patient signs from standard sensors)?
   
   NO       YES
   
   Integrity ☐ ☐
   (Check yes if a loss of integrity could cause or contribute to loss of life, otherwise check no.)

   Availability ☐ ☐
   (Check yes if a loss of availability could cause or contribute to loss of life, otherwise check no.)

   Confidentiality ☐ ☐
   (Check yes if a loss of confidentiality could cause or contribute to loss of life, otherwise check no.)

   Scenario Descriptions: ____________________________________________
   (Describe the information, processes, and circumstances that could lead to loss of life. This information is used to develop testing scenarios and to decide what other tests to perform.)

2. Can general use of this product cause or contribute significantly to the loss of the enterprise (e.g., use of an accounts receivables package or a software package that is an essential component of the mission of the organization)?

   NO       YES
   
   Integrity ☐ ☐

   Availability ☐ ☐

   Confidentiality ☐ ☐

   Scenario descriptions: ____________________________________________

1B. Other Potential Losses

1. Can general use of this product cause or contribute significantly to financial losses or increased liability (e.g., use of a networked transaction package that encourages customers to send their credit card numbers over the internet in unencrypted form)?

   Range of Severity:
   Likely Severity:
3. Can general use of this product cause a loss of market share (e.g.,
   use of a software package that provides limited support for de facto
   standards in the industry)?

Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>□</td>
</tr>
<tr>
<td>Availability</td>
<td>□</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>□</td>
</tr>
</tbody>
</table>

Scenario Descriptions: ____________________________________________

3. Can general use of this product cause a loss of reputation (e.g., use
   of a software package that uses unspecified or noncertifiable
   supporting data to make decisions or produce products in a
   primary business area)?

Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>□</td>
</tr>
<tr>
<td>Availability</td>
<td>□</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>□</td>
</tr>
</tbody>
</table>

Scenario Descriptions: ____________________________________________

4. Can general use of this product cause a loss of corporate desired or
   required confidentiality (e.g., use of a mail software package)?

Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>□</td>
</tr>
<tr>
<td>Availability</td>
<td>□</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>□</td>
</tr>
</tbody>
</table>

Scenario Descriptions: ____________________________________________

5. Can general use of this product cause or contribute significantly to
   a loss of quality or integrity (e.g., widespread use of software
   packages of unknown origin or unknown quality)?

Range of Severity:
Likely Severity:
### 10. Specific Vulnerabilities

1. Does use of this product require privileged access?
   1a. If yes, does this product provide a means to execute other applications and system commands without exiting the product and possibly retaining the product's privileged access (e.g., sendmail with debug turned on. Many software packages allow users to exit to the operating system temporarily)?
   1b. Does the product permit users to retain privileged access on termination or in the event of application failure (i.e., this is a condition that should be checked on any package that requires or uses privileged access. Malicious programs may do this on purpose, well behaved programs do not)?

### Part 2. Intended Use

#### 2A. Extreme Security Issues

1. Can the intended use of this product cause or contribute to loss of life (e.g., use of a software package for monitoring vital patient signs from standard sensors)?

<table>
<thead>
<tr>
<th></th>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Availability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Confidentiality</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Check yes if a loss of integrity could cause or contribute to loss of life, otherwise check no.)

<table>
<thead>
<tr>
<th></th>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Check yes if a loss of availability could cause or contribute to loss of life, otherwise check no.)
Confidentiality □ □

(Check yes if a loss of confidentiality could cause or contribute to loss of life, otherwise check no.)

Scenario Descriptions: ______________________________________________________

(Describe the information, processes, and circumstances that could lead to loss of life. This information can be used to develop testing scenarios and to decide what other tests to perform.)

2. Can the intended use of this product cause or contribute significantly to the loss of the enterprise (e.g., use of an accounts receivables package or a software package that is an essential component of the mission of the organization)?

<table>
<thead>
<tr>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity □ □</td>
<td></td>
</tr>
<tr>
<td>Availability □ □</td>
<td></td>
</tr>
<tr>
<td>Confidentiality □ □</td>
<td></td>
</tr>
</tbody>
</table>

Scenario Descriptions: ______________________________________________________

2D. Other Potential Losses

1. Can the intended use of this product cause or contribute significantly to financial losses or increased liability (e.g., use of a networked transaction package that encourages customers to send their credit card numbers over the internet in unencrypted form)?

Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity □ □</td>
<td></td>
</tr>
<tr>
<td>Availability □ □</td>
<td></td>
</tr>
<tr>
<td>Confidentiality □ □</td>
<td></td>
</tr>
</tbody>
</table>

Scenario Descriptions: ______________________________________________________

2. Can the intended use of this product cause a loss of market share (e.g., use of a software package that provides limited support for de facto standards in the industry)?

Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity □ □</td>
<td></td>
</tr>
<tr>
<td>Availability □ □</td>
<td></td>
</tr>
<tr>
<td>Confidentiality □ □</td>
<td></td>
</tr>
</tbody>
</table>

Scenario Descriptions: ______________________________________________________

3. Can the intended use of this product cause a loss of reputation (e.g., use of a software package that uses unspecified or unverifiable supporting data to make decisions or produce products in a primary business area)?
Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th></th>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Availability</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>□</td>
<td>□</td>
</tr>
</tbody>
</table>

Scenario Descriptions: ____________________________________________________________

4. Can the intended use of this product cause or required confidentiality (e.g., use of a mail software package)?

Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th></th>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Availability</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>□</td>
<td>□</td>
</tr>
</tbody>
</table>

Scenario Descriptions: ____________________________________________________________

5. Can the intended use of this product cause or contribute significantly to a loss of quality or integrity (e.g., widespread use of software packages of unknown origin or unknown quality)?

Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th></th>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Availability</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>□</td>
<td>□</td>
</tr>
</tbody>
</table>

Scenario Descriptions: ____________________________________________________________

6. Can the intended use of this product cause or contribute significantly to a loss of availability (e.g., for an organization with heavy network use, dependency on communications or network software without source code or maintenance guarantees)?

Range of Severity:
Likely Severity:

<table>
<thead>
<tr>
<th></th>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Availability</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>□</td>
<td>□</td>
</tr>
</tbody>
</table>

Scenario Descriptions: ____________________________________________________________