

Contents

Introduction	3
Risk Assessment and Management Approach	5
Supply Chain Integrity Approach: Standards Correlation or Business Process Model	7
Standards Correlation Approach.....	8
Phase 1: Planning	8
Phase 2: Discovery	8
Phase 3: Assessment	9
Define control categories within relevant standards.....	9
Rate the effectiveness of the selected control activities	9
Selecting control activities.....	10
Phase 4: Development.....	11
Phase 5: Validation	13
Phase 6: Implementation.....	13
Business Process Model Approach.....	14
Phase 1: Planning	14
Phase 2: Discovery	14
Phase 3: Assessment	17
Analyze attack scenarios and identify existing controls.....	17
Rate and prioritize risk	18
Phase 4: Development.....	20
Phase 5: Validation	20
Phase 6: Implement.....	20
Controls for Managing Supply Chain Risk	21
Identity and access management controls.....	21
The Security Development Lifecycle	21
Operational Security Assurance.....	22
Software integrity policies and procedures.....	23
Anti-counterfeit measures	23


Introduction

As the integrity and trustworthiness of information and communications technology (ICT) systems become increasingly critical to a nation's well-being, governments worldwide are increasingly concerned about the threat to their ICT systems through the global supply chain for ICT products. These concerns stem from the fear that an adversary might tamper with products during their development, manufacture, or delivery.

For Microsoft the protection of the technology supply chain is not new, because of the importance of protecting the company's intellectual property. Microsoft has invested significant efforts in protecting these assets, including controlling access to Microsoft software while it is being developed, scanning releases for malware, and applying a digital signature to binary files within a product so that users can verify that the software they have received actually comes from Microsoft and has not been modified. Microsoft also seeks to minimize unintentional vulnerabilities—and make it harder to introduce intentional vulnerabilities disguised as unintentional vulnerabilities—through the [Security Development Lifecycle](#). Collectively these efforts, along with proof of identity and anti-counterfeit measures, play an important role in mitigating software integrity risks and providing a more secure computing experience for all users.

Microsoft also shares the company's approach¹ with industry-led organizations such as the [Software Assurance Forum for Excellence in Code](#) (SAFECode) the [Open Group Trusted Technology Forum](#), and international standards organizations, and works closely with the government, enterprises, and consumer customers around the world to assess, manage, and respond to risks. Microsoft's broad experience working with both governments and industry allows the company to understand their different priorities and come to a set of recommendations, which provides a framework to benefit both. Microsoft's experience also helped identify risk as a core focus area from assessment to management, to understand the holistic lifecycle of the threat and how best to develop strategies and solutions to neutralize it.

¹ Any solution to technology supply chain challenges should be risk-based, transparent, flexible, and be based on global standards. See the Microsoft white paper on *CyberSupply Chain Risk Management*, published July 14, 2011.



The goal of this paper is to describe Microsoft's framework for incorporating software integrity risk-management practices in both the product development process and online services operations. This paper first presents an overview of the company's approach to providing risk-based protection for the integrity of Microsoft's software during development and distribution. It then presents the details of Microsoft's approach to assessing the risks to the supply chain and determining where to apply security controls. Finally, the paper summarizes some of the specific controls that Microsoft relies on to protect the integrity of Microsoft software.

Microsoft's methodology is designed to improve software security and trustworthiness across the people, processes, and technologies that make up a modern ICT supply chain. Applying it produces distinct benefits, such as increased accountability and transparency, helping to lower risk for both the organization and the software user. The paper may therefore be of interest to those in government, as well as to customers. It outlines the steps that Microsoft takes to develop and implement a risk-based approach to managing software integrity risks. It also summarizes specific security and software integrity measures that help reduce the risk of attacks on the product supply chain.

Risk Assessment and Management Approach

Overall, Microsoft focuses on risks using an all-hazards approach to risk assessment. With this approach as a backdrop, Microsoft assesses the entire organization to identify what is most important to the company and its stakeholders. This begins by identifying the most important processes and then moves to identifying their key dependencies, which include systems and data.

At the corporate level, Microsoft's Operational Enterprise Risk Management (OERM) strategy aligns with ISO 31000: 2009, *Risk management—principles and guidelines*. An OERM risk-assessment process has three primary components that inform how risk is managed programmatically across the company:

- **Risk identification.** This process is led by OERM, but subject matter experts in the business units responsible for the risk drive the execution and identification activities. This enables significant internal oversight and coordination across the company.
- **Risk analysis.** Microsoft OERM teams also review the risk-assessment process for data quality and then map identified risks to common risk descriptions, drivers, and domains. Throughout this process, there are a number of quantitative and qualitative analyses that are conducted to evaluate risks, including evaluating cross-category risk exposures within each of the four pillars and setting priorities.
- **Risk evaluation.** Through the risk-evaluation process, OERM teams evaluate common theme areas and domains for changes to its existing enterprise risks.

The goal of risk management is not to eliminate all risk but rather to mitigate, transfer, or accept risk through organizational, technical, and programmatic efforts that are supported and sustained through company-wide risk-management offices and programs. Microsoft Enterprise Risk Management (ERM) provides insight into the company's most significant short- and long-term risks, ensuring accountability and management of these risks, and facilitating a global and programmatic approach to risk management. Microsoft organizes its risk-management efforts around prevention, detection, response, and recovery. Three of these four areas (prevention, response, and recovery) have mature capabilities that are anchored in international standards and amplified through best practices. For detection, where there are no globally accepted supporting standards, the company has built several best practices to address the rapidly evolving threat landscape.

Prevent. A large part of risk management is focused on preventing events from happening, containing events from expanding, and/or decreasing the negative impact when events occur. Microsoft's ERM program helps risk owners determine where and what preventive controls to invest in and tracks the effectiveness of those controls.

Detect. In the current threat environment, detection may be the most critical of the four risk-management areas. Talented and patient adversaries will delete logs, change data, and take whatever actions are necessary to gain and retain access to a network. Detecting when an attacker has gained access to a network, system, or asset requires incredibly skilled forensic investigators equipped with cutting-edge tools and resources. Microsoft takes a multi-layered approach to detecting cyberincidents, with responsibility spread among the business units across the company. Data is collected from systems and devices by using common industry tools and standards and through well-known Microsoft products or security organizations, as well as through Microsoft's own internal processes and technologies. That data is then analyzed by the teams that administer the environments to detect isolated incidents, and by a centralized group that looks for attacks against multiple business groups or advanced attacks by determined adversaries. Microsoft's privacy practices, the applicable privacy statements, and relevant regulatory or contractual requirements provide a framework to help ensure that the data is always appropriately handled.

Respond. Many companies are faced with two different types of response: to defend the enterprise itself and to mitigate an impact on customers. Microsoft strongly encourages cybersecurity policy frameworks to use the Incident Command System (ICS) as a foundation. ICS has an established history of success in the United States; its features include:

- Allowing for the integration of facilities, equipment, personnel, procedures, and communications operating within a common organizational structure.
- Enabling a coordinated response among various jurisdictions and functional agencies, both public and private.
- Establishing common processes for planning and managing resources.

Recover. An organization's ability to recover from a cybersecurity incident largely depends on its overall capabilities for reliability and resiliency. For Microsoft and others in the ICT sector, this necessitates investments in processes and technology to improve reliability, a continued focus on every customer's experience, and active partnerships with a wide variety of software and hardware companies.

Supply Chain Integrity Approach: Standards Correlation or Business Process Model

Microsoft's supply chain assessment follows a principled framework and is focused on both the development and operation of software and online services. As new risks are discovered, the process may require changes to the way Microsoft manages and trains people, processes and technology to mitigate these risks in software development and operational environments.

Microsoft uses either a Standards Correlation or Business Process Modeling approach to perform a risk assessment. Each approach follows the same six phases, from planning to implementation:



Standards Correlation is Microsoft's preferred approach, when relevant mature standards exist that can also mitigate software integrity threats. It tends to be less resource intensive because it identifies pre-defined standard controls. It is particularly effective in analyzing operations environments for online services as operational security already includes many relevant standards.

Microsoft uses the Business Process Model if there are no relevant and mature standards, or if applicable standards oversimplify or misrepresent the situation. For example, this model is particularly effective for software development, because few relevant standards exist. It produces a detailed, documented representation of the process flow to identify threats and process weaknesses as well as potential process improvements.

Standards Correlation Approach

In the Standards Correlation approach, Microsoft first identifies the broad classes of software integrity threats, such as an insider intentionally tampering with a product or service. Following from that, the company associates these threats with mature, relevant standards. The next task is to identify a subset of control activities within the appropriate standards that are most effective in mitigating threats to software integrity. For example, to offer software products or services to the United States government, Microsoft complies with standards such as the Federal Information Security Management Act (FISMA) and International Standards Organization (ISO) standard 27002. Some of the control activities specified by these standards, such as physical and logical access controls, can also be effective in mitigating software integrity risks.

Phase 1: Planning

In this phase, Microsoft defines the objectives, scope, and approach for the assessment, including roles, responsibilities, milestones, and deliverables. The resulting plan includes a justification for using the Standards Correlation approach and a description of how meeting the assessment's objectives will support the goals for software integrity. It specifies the appropriate level of management and resource commitment from the teams that are essential to the success of the assessment, including the team performing the assessment and the team that owns the underlying process being evaluated.

Phase 2: Discovery

The goals of this phase are to identify broad classes of threats to software integrity and to understand the detailed control activities that are likely to mitigate those classes of threats.

Broad classes of software integrity threats in an operational environment include (but are not limited to):

- Malicious software inserted into a production environment by a staff member.
- Malicious configuration changes to a production environment—for example, where a staff member makes a malicious configuration change to the production environment that reduces its security, such as by altering network traffic controls designed to protect against attack or malware.

The Standards Correlation approach requires the identification of the relevant standards and related control activities to be evaluated. These may include standards relating to regulatory compliance, certification, or other industry-accepted guidance. For example, standards are derived from government concerns about operational controls to protect the integrity of services, and from industry certifications that may or may not be required by customers. Standards that may be used in a software integrity assessment include:

- [Guide for Assessing the Security Controls in Federal Information Systems and Organizations](#) (FISMA)
- [ISO 27001 or 27002](#)
- [Government of Canada Operational Security Standard](#)

The Discovery Phase ends with a list of broad classes of software integrity threats, relevant standards, and internal policies and procedures.

Phase 3: Assessment

The goals of this phase are to identify control categories, choose control activities relevant to software integrity threats, assign the selected control activities to the appropriate control categories, rate their effectiveness, and prioritize them based on their effectiveness ratings.

Define control categories within relevant standards

With broad classes of software integrity threats in mind, Microsoft reviews the standards documentation and all control activities described within it. A subset of control activities relevant to addressing threats to software integrity is then selected. As this subset may include hundreds of individual control activities, the team may also need to identify a handful of control categories to group similar individual control activities. This grouping eases the assessment of the individual control activities, which can later be included in discrete control practices. For example, in online services operations, control categories could include change control, separation of duties, monitoring, logical access control, and physical access control.

Rate the effectiveness of the selected control activities

Microsoft uses a risk-rating framework to ensure that risk is communicated to appropriate decision-makers in a consistent format. To rate the effectiveness of control activity, the team captures all of the relevant control activities in a spreadsheet, with each control activity listed in a separate row and each software integrity threat in a separate column (Figure 1 below).

Subject matter experts (SMEs) with knowledge of the underlying processes and an understanding of the selected controls rate each control activity for its effectiveness at mitigating the threat to software integrity on a scale from 1 through 5, with 1 representing the least effectiveness and 5 representing the most. It is useful to have more than one SME review and rate the selected controls, and then compare effectiveness ratings to resolve major differences, if any, so that the outcome is objective. To determine which control activities are the most effective against the broadest classes of software integrity threats, an average of the control activity effectiveness ratings for each threat is calculated.

Threat →	Control Activity ↓	Malicious software inserted into production environment	Malicious change to production environment	Theft of customer data - Physical	Control Effectiveness Score (Higher is better) (5 MAX)
1	ISO27002: Security perimeters (barriers such as walls, card controlled entry gates or manned reception desks) should be used to protect areas that contain information and information processing facilities.	2.0	2.0	4.0	2.7
2	ISO27002: Secure areas should be protected by appropriate entry controls to ensure that only authorized personnel are allowed access.	3.0	3.0	4.0	3.3
3	ISO27002: Physical security for offices, rooms, and facilities should be designed and applied.	3.0	3.0	4.0	3.3
4	ISO27002: Physical protection against damage from fire, flood, earthquake, explosion, civil unrest, and other	1.0	1.0	1.0	1.0
5	forms of natural or man-made disaster should be designed and applied.	3.0	3.0	3.0	3.0
6	ISO27002: Physical protection and guidelines for working in secure areas should be designed and applied.	3.0	3.0	3.0	3.0

Figure 1: Standard Control Activity Effectiveness Rating Example (Physical Access Control Category)

Selecting control activities

Based on the overall view of threats, the team agrees on an average effectiveness rating to be included in a final list of selected control activities. For example, based on the overall view of threats and risks, the team may choose control activities that have an effectiveness rating of 2.5 or higher and prioritize accordingly. At the end of this assessment phase, the team creates a standards correlation spreadsheet that includes:

- A list of mature, relevant standards
- Control categories with detailed control activities relevant to threats to software integrity
- Effectiveness ratings 1 to 5 for all of the selected control activities
- Control activities ranked in order of effectiveness at mitigating software integrity risks with a rating of 2.5 or higher

Phase 4: Development

The goal of this phase is to document the control activities selected in the Assessment phase in the form of software integrity control practices. These include the overall objective, related requirements and guidance on how to comply with those requirements. For example, in the operations, each control category could become its own control practice. The control practices represent a set of ideal requirements that would be considered to be effective at mitigating broad classes of software integrity threats. An example of such a class is unauthorized physical access to systems and media with the aim of modifying access control policies.

The outcome of the Development phase is a list of software integrity control practices that expresses an ideal state that would be effective in mitigating broad classes of threats to software integrity. However, until the Validation phase is complete, the status of the proposed software integrity control practices or their commercial viability is unknown.

SI-#:	Operations: Physical Access Control
Objective	Ensure that access to physical information systems and facilities used in operations is limited to only authorized personnel
Requirements	<ul style="list-style-type: none"> Physical Access Control procedures to prevent unauthorized access Monitoring and response Location of information systems to prevent unauthorized access and environmental hazards
Procedures	<ul style="list-style-type: none"> Enforcing authorization for all physical access points (including designated entry/exit points) to the facility where the information system resides (excluding those areas within the facility officially designated as publicly accessible) Verifying individual access authorization before granting access to the facility Controlling entry to the facility using physical access devices (e.g., keys, locks, combinations, card readers) and/or guards Controlling access to areas officially designated as publicly accessible in accordance with the organization's assessment of risk Securing keys, combinations, and other physical access devices via process and physical protections, such as locked cabinets with a documented access procedure Defining, documenting, approving, and enforcing physical access restrictions associated with changes to the information system Defining the information system components to be protected from unauthorized physical access and using lockable physical casings to protect them Developing a system capable of detecting and preventing physical tampering or alteration of hardware components within the system Safeguards such as guards, alarms, and monitors that are active at all times at every physical access point to the facility That no equipment, information or software should be taken off-site without prior authorization
Monitoring and Response	<ul style="list-style-type: none"> Usage of alarms and surveillance equipment monitoring physical intrusion Automated mechanisms to recognize potential intrusions and initiate responses
Location	<ul style="list-style-type: none"> Positioning information system components within the facility to minimize potential damage from physical and environmental hazards Positioning information system components within the facility to minimize the opportunity for unauthorized access.

Figure 2: Control Practice Example: Physical Access Controls

Phase 5: Validation

The goal of this phase is to identify, based on a cost-benefit analysis, which control requirements a particular product group currently meets, and which the group needs to implement to manage risks to software integrity.

To meet this goal, the assessment team compares the software integrity control practices against the internal policies and procedures identified in the Discovery phase. Each control practice requirement the group currently meets becomes part of a baseline of software integrity policies and procedures. For those requirements that the group does not meet, the assessment team performs a cost-benefit analysis of implementing the unmet control practice requirement and the risk it mitigates. Factors they consider include the extent to which the product group partially implements the software integrity control practices and the level of effort required to meet the new requirements. Whatever new control practice requirements the group decides it should meet become part of the proposed software integrity policies and procedures. At Microsoft, these proposed policies and procedures are then presented to a broader audience for feedback, including groups responsible for security, compliance, risk management, and engineering, as well as the appropriate decision-makers who can approve the policy and corresponding procedures.

The Validation phase results in a proposed set of software integrity policies and procedures ready for implementation. Unlike control practice requirements, which represent an ideal state for an organization's software integrity efforts, the proposed policies and procedures reflect what product teams actually do and are committed to doing within a specified timeframe to manage software integrity risks.

Phase 6: Implementation

The Implementation phase begins with the approval of the new software integrity policies and procedures set in the Validation phase, and communication of these policies and procedures to all of the stakeholders within Microsoft who are affected. In addition, the following actions also take place as part of implementation:

- Publishing the approved policies and procedures in the appropriate organizational portals (for example, where other policies are published)
- Appropriate training for policies and procedures and any related tools
- Corresponding management of policies and procedures compliance tracking

Business Process Model Approach

In the Business Process Model approach, Microsoft first creates a graphical representation of the software development activities and the workflow that defines the product group's actual process. This graphical representation can follow a standard notation methodology, such as Business Process Modeling Notation, which uses flowcharts and other standardized graphical notations that are easily understandable by technical and non-technical audiences alike. Microsoft has found that using the Business Process Model has made it easier to analyze software integrity attack scenarios to define areas of risk, and develop or strengthen corresponding controls to mitigate those risks.

Phase 1: Planning

This phase defines the objectives, scope, and approach for the Assessment, including roles and responsibilities, milestones, and deliverables. The plan includes a description of how meeting the Assessment's objectives will support Microsoft's goals for software integrity. Completion of the planning phase will produce a documented plan with the appropriate level of management and resource commitment from the teams essential to the success of the Assessment, including the team performing the Assessment and the team that owns the underlying process being evaluated.

Phase 2: Discovery

The goal of this phase is to identify, understand, and document the business process related to developing and operating software products and services. A well-established method for representing these activities is a Business Process Diagram, a network of graphical objects that shows engineering activity and the flow controls that define their order of performance.

To build the diagram, the team collects all relevant information about the current state of every relevant business process, and uses Business Process Modeling Notation, which provides a standardized way to show business process activities (illustrated in Figure 3).

No	Name	Executed By	Inputs	Description	Outputs
5.	Receive Bug Notification	Product Dev. Team	Email	Product Development Team receives and reviews the bug notification including details about the bug found.	Human Response
6.	Mark Bug as In Progress	Product Dev. Team	Human Response	The Product Development Team marks the bug as in progress in the Bug Database.	Bug Status Change
7.	Notify Test Team of Bug Status	Bug Database	Bug Status Change	The Bug Database application sends a notification mail to the Test Team that informs them of the bug status change.	Email
8.	Receive Notification of Bug Status	Test Team	Email	The Test Team receives the notification email, which informs them that the Product Development Team has started fixing the bug.	N/A
9.	Request Source Code Check-Out	Product Dev. Team	Human Response	A developer on the Product Development Team makes the request to the Source Repository to check out the product source code relating to the bug filed.	Application Activity
10.	Lock Source Code for Check-Out	Source Repository	Application Activity	The Source Repository application locks the source code so that only the individual who has the source code checked out can edit it.	Application Activity
11.	Transfer Source Code to Dev. Machine	Source Repository	Application Activity	The Source Repository application transfers the checked-out source code to the developer's machine.	Source Code
12.	Receive Source Code	Product Dev. Team	Source Code	The developer on the Product Development Team receives the source code on his machine.	Human Response
13.	Develop Bug Fix	Product Dev. Team	Human Response	The developer on the Product Dev. Team develops a resolution to the bug filed.	Source Code
14.	Check-In Code	Product Dev. Team	Source Code	The developer checks in the updated code meant to fix the bug that has been filed.	Human Response
15.	Unlock Source after Check-In	Source Repository	Human Response	The Source Repository application receives the source code check-in and unlocks the code.	Application Activity
16.	Mark Bug as Fixed	Product Dev. Team	Human Response	The developer who fixed the bug marks the bug as fixed in the Bug Database application.	Application Activity

No	Name	Executed By	Inputs	Description	Outputs
17.	Notify Test Team of Bug Status	Bug Database	Application Activity	The Bug Database application sends a notification mail to the Test Team that informs them of the bug status change.	Email
18.	Notification of Bug Status	Test Team	Email	The Test Team receives the email that the bug status has been changed to "Fixed."	Human Response
19.	Re-test Product for Bug Resolution	Test Team	Human Response	The Test Team retests a new build of the product to determine if the bug has been fixed.	Test Output
20.	Mark Bug as Open	Test Team	Human Response	If the tests showed that the bug was not fixed, the Test Team will mark the bug as open which returns the process to Step 5. Otherwise the process ends and the bug is verified as fixed.	Application Activity

Figure 4: Process Narrative Example

At the end of the Discovery phase, the team has a Business Process Model that includes a set of diagrams and narratives that reflect the current state of business processes.

Phase 3: Assessment

The goals of this phase are to analyze classes of potential threats to software integrity and weaknesses (attack scenarios) within the Business Process Model, identify the product team's existing controls, and prioritize the resulting risk from high to low. The risk-rating framework evaluates the likelihood of the attack (threat) scenario occurring, its potential impact, and the effectiveness of existing controls. This framework provides a consistent format for communicating risk to process owners and executive decision-makers.

Analyze attack scenarios and identify existing controls

The first step is to identify and analyze attack scenarios against weaknesses in the software development or delivery process. For example, an actor could intentionally insert malware into a product during the development or delivery of software products, through either a process or a technology weakness. Another example could involve a network administrator intentionally replacing part of the service with code that is malicious in nature during the deployment of an update to an online service. Another possible threat could result from a developer with full access to the production hardware who tries to exploit weakness within the production environment to reduce the security of the production system.

An understanding of the skills, resources, and motivations of different types of potential actors is helpful to identify attack scenarios. Once the attack scenarios are defined, the team identifies controls currently in place to mitigate them. Examples of types of controls include preventive and forensic controls.

- **Preventive controls** are designed to stop violation of policies and procedures before any damage can be done—for example, limiting source code access to only approved groups of individuals, or scanning all software releases for known malware before the software is shipped.
- **Forensic controls** are used to identify a violation of policies or procedures after the violation has happened. Common forensic controls include logging of individual activity, verifying the identity of individuals who develop or deploy software or services, and ensuring that changes to software are traceable to an individual. Forensic controls have a benefit beyond holding attackers responsible for their misdeeds: they also serve as a deterrent by making it clear to potential attackers that their actions are likely to be detected.

Rate and prioritize risk

Each control identified is rated for its effectiveness in mitigating the attack scenario on a scale from 1 to 5, with 1 representing the lowest effectiveness and 5 representing the highest. At Microsoft, SMEs with knowledge of the company's underlying processes and an understanding of the existing controls assist in rating their effectiveness.

Once the software integrity attack scenarios are identified, along with control effectiveness ratings, the team can assign a risk rating to each scenario by determining its impact and likelihood, including any offsets based on control effectiveness.

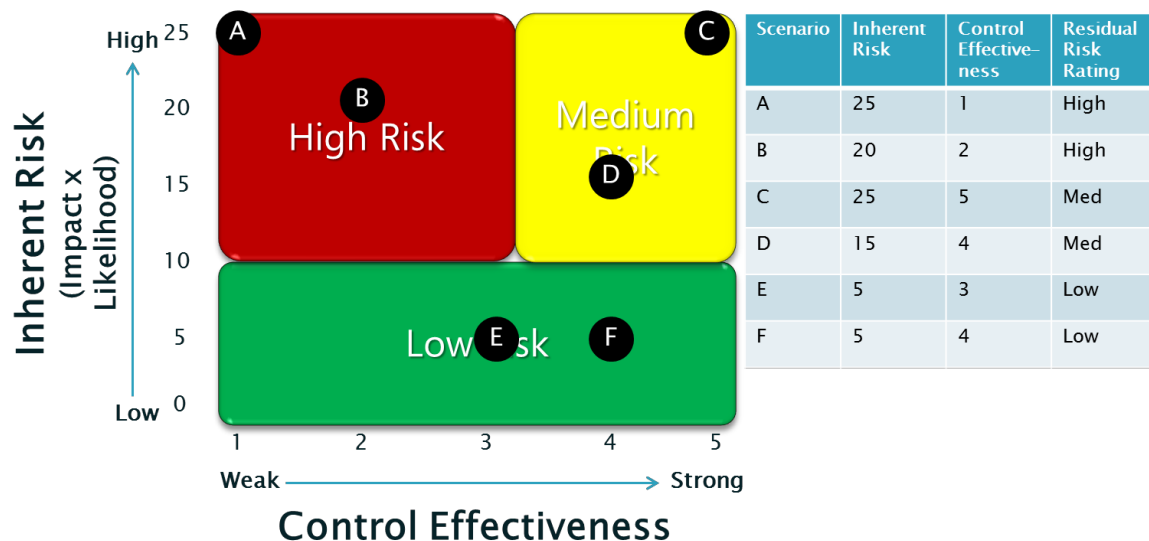


Figure 5: Risk Rating Example (Sorted by Residual Risk Rating from High to Low)

- **The impact** of an attack scenario is a reflection of potential loss of revenue or reputation to Microsoft or its customers from a legal or compliance perspective if the scenario occurred. Impact is rated on a scale from 1 through 5, with 1 being minimal and 5 being critical.
- **The likelihood** of the attack scenario is a reflection of the depth and breadth of knowledge needed to exploit a weakness in the business process. Each attack scenario is rated for its probability of occurring within a defined timeframe on a scale of 1 through 5, with 1 being slightly probable and 5 having a high probability. For example, if collusion is needed to execute an attack, this attack scenario would be rated as less likely than an attack which can be conducted by individual acting alone.
- **Inherent risk** is the likelihood of an attack scenario occurring assuming there are no controls in place. Inherent risk is calculated by multiplying the likelihood rating by the impact rating and can be represented as a number with a value between 0 and 25 (see Figure 5 above).
- **Residual risk** is the likelihood of an attack scenario occurring after accounting for existing controls and their effectiveness ratings. Residual risk is an evaluation of both the inherent risk and the effectiveness of controls, which are represented as high, medium, or low. Risks are sorted by residual risk, which is the risk that remains after control effectiveness has been taken into account.

The Assessment phase culminates in a list of software integrity attack scenarios, existing controls with effectiveness ratings, and risks prioritized from low to high. Figure 6 provides an example.

Step	Name	Attack Scenarios	Controls in Place	Consequences	Impact	Likelihood	Control Effectiveness	Residual Risk Rating
11 through 20	Bug Fix Example	Developer checks in malware when performing a fix for a known bug,	Testing is performed after the bug fix is created to ensure that the bug is properly fixed.	If attack scenario succeeds, malware could be included in a software release.	5	4	2	High

Figure 6: Risk Rating Spreadsheet Example

Phase 4: Development

The goal of this phase is to develop control practices designed to address the risks identified in the Assessment phase as requiring mitigation. These control practices should include the overall objective, related requirements, and guidance on how to comply with the requirements. Software integrity control practices include identity and access management, antimalware and virus scanning, and code signing of binary files.

The Development phase results in a list of software integrity control practices that express an ideal state that would be effective in mitigating attacks on the integrity of software. The formats for describing the control practices are similar to those used in the Standards Correlation approach (illustrated in Figure 2). In general, the two approaches to analysis should produce roughly similar sets of control practices, with two potential exceptions:

- Compliance with a standard may dictate the inclusion or implementation of a control that the Business Process Model approach would not require.
- The Business Process Model approach may, for reasons of effectiveness or efficiency, select a control that would not be required by the Standards Correlation approach.

Phase 5: Validation

The goal of this phase is to identify, based on a cost-benefit analysis, which control requirements a particular product team currently meets and which the group still needs to implement to manage software integrity. This follows the process described in the Validation phase of the [Standards Correlation](#) approach.

Phase 6: Implement

The goal of this phase is to implement the software integrity baseline refined in the Validation phase. This phase includes approval of the new software integrity policies and procedures, and communication of the policies and procedures to all of the stakeholders within Microsoft who are affected. This follows the process described in the Implementation phase of the Standards Correlation approach.

Controls for Managing Supply Chain Risk

Microsoft manages supply chain risk through the ongoing development and usage of:

- Identity and access management controls
- Security Development Lifecycle (SDL)
- Operational Security Assurance
- Software integrity policies and procedures
- Anti-counterfeit measures

Identity and access management controls

Managing access to code is vital to ensuring protection of intellectual property. These measures promote accountability and traceability of actions related to Microsoft source code. To this end:

- Microsoft uses policies, procedures, and technology to manage personnel access to intellectual property. The company issues physical and digital credentials for authentication and access control. Personnel use a combination of these credentials for access to Microsoft's physical and digital assets, such as buildings, systems, and services, based on business need.
- Microsoft policy requires that Source Code Management systems be hosted in managed environments within corporate data centers and that physical access to data centers be controlled by multi-factor authentication. Access is limited to Microsoft personnel required to manage the systems only.
- Access to Microsoft source code is based on granting users only the rights necessary to perform their jobs. Access is granted based on business need and is linked to the individual's digital credentials.
- Microsoft manages identity by assigning only one personnel number per individual and does not reuse personnel numbers.

The Security Development Lifecycle

Microsoft's Security Development Lifecycle (SDL), a security assurance process, is a foundational element for reducing the risk of product vulnerabilities and protecting against their introduction—whether malicious or inadvertent—during software development. A mandatory engineering policy since 2004, the SDL provides a comprehensive process for applying secure development practices across a large organization. The rigor of SDL processes, tools, training, and governance reduces the number and severity of vulnerabilities, and thorough application of the tools and

processes specified by the SDL helps to minimize the risk of undetected insertion of malicious code in Microsoft's products.

The SDL introduces security into every phase of the development process, incorporating accountability and continuous process improvement, as well as ongoing security education and training of technical personnel.

Threat modeling, a central SDL activity, identifies threats that could harm each asset and determines the likelihood of harm. Threat modeling helps the product team identify the need for specific security features and areas that require especially careful code review and security testing.

A key component of the SDL, the Final Security Review (FSR), is a comprehensive review of the security of a product before it ships. The FSR validates that correct practices, tools, and processes were applied during product development. The FSR provides an independent review of the product's readiness for shipping, encompassing both development requirements and response planning.

Operational Security Assurance

Operational Security Assurance (OSA) is a framework that combines the company's knowledge with the experience of running hundreds of thousands of servers in data centers around the world that deliver more than 200 online services to more than 1 billion customers and 20 million businesses in 88 countries. Microsoft uses it to minimize risk by ensuring that ongoing operational activities follow rigorous security guidelines and by validating that guidelines are actually being followed effectively. When issues arise, a feedback loop helps ensure that future revisions of OSA contain mitigations to address them. OSA helps make Microsoft cloud-based services' infrastructure more resilient to attack by decreasing the amount of time needed to prevent, detect, contain, and respond to real and potential Internet-based security threats, thereby increasing the security of those services for customers. The goals of OSA are straightforward:

- Establish a scalable process to improve operational security across all Microsoft cloud service offerings.
- Respond to security challenges as they emerge from the evolving threat landscape by providing simple, predictable updates to the framework that continuously improve operational processes and procedures used by Microsoft engineering teams.
- Reduce threat identification and response times by ensuring that online services have effective attack-detection capabilities and are capable of fielding a unified response team that can resolve incidents rapidly and at scale.
- Complement recognized standards such as NIST Special Publication 800-53 and ISO 27001.

- Be flexible enough to work with a broad range of Microsoft cloud services, from those that make up small custom solutions to large services used by both consumer and enterprise customers.
- Maintain a high level of service availability and minimize the impact of both planned and unplanned incidents to customer.

Software integrity policies and procedures

Microsoft also employs policies, procedures, and technology to preserve the integrity of its software products, including code signing and checking for malware. Before release, Microsoft's policies require:

- Products to be scanned for viruses and malicious code. Microsoft uses specialized tools that examine each file within a product and scan it with state-of-the-art anti-malware products based on virus signatures provided by multiple scanning tool vendors.
- Code signing, which refers to the application of a digital signature to binary files within a product. Microsoft corporate policy requires that files be covered by a digital signature before they are released to the public. This process allows users to verify that the software they receive actually comes from Microsoft and has not been modified.

In addition to the mandatory engineering policies above, Microsoft employs other processes to preserve the integrity of Microsoft's products, including source code management and product build and packaging management.

Anti-counterfeit measures

Microsoft also strives to ensure that the company releases only genuine Microsoft software that has been developed pursuant to strict security standards.

- To protect customers from the risks of counterfeit software, Microsoft actively identifies counterfeit versions of its software, works to maintain the integrity of its distribution models, and works with law enforcement agencies around the world to help reduce piracy.
- Microsoft actively participates in supply chain security forums, including the US government's [Customs Trade—Partnership Against Terrorism](#) (C-TPAT), and was one of first 20 companies to be Tier 3 certified. Microsoft also provides education, engineering tools, and enforcement policies to help customers and organizations identify physical editions of counterfeit software.

- Microsoft's Authorized Replicator program restricts to a limited set of partners the ability to create and distribute physical media, such as CD/DVDs, containing Microsoft products. This program prevents unauthorized changes through strict controls, including limits on physical access, inventory tracking, and additional integrity checks of the product. Customers who download products directly through Microsoft's Volume Licensing Software Center are guaranteed genuine Microsoft products produced in accordance with company security processes.
- To protect customers from the risks of counterfeit software, which could contain vulnerabilities, Microsoft actively identifies counterfeit versions of its software, works to maintain the integrity of its distribution models, and works with law enforcement agencies around the world to help reduce piracy. Microsoft participates in groups, such as Verafirm that focus on verifying the authenticity of software used by companies throughout their supply chain.
- Microsoft also takes legal and technical action to address criminals targeting the supply chain. One such initiative is Project MARS (Microsoft Active Response for Security), which focuses on efforts to disrupt criminal infrastructure. This includes taking legal and technical action to pursue botnets and help undo the damage they cause. In 2012, Project MARS helped take down the Nitel botnet, which infected computers through vulnerabilities in the supply chain.

© 2014 Microsoft Corp. All rights reserved.

This document is provided as-is. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it. This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

