中国科学技术大学
University of Science and Technology of China

Software Architecture

SSE USTC Qing Ding
dingqing@ustc.edu.cn
http://staff.ustc.edu.cn/~dingqing

Quality Attributes of Architecture III Modifiability & Security

Agenda

- Modifiability & Security
  - Source of stimulus
  - Stimulus
  - Environment
  - Artifact
  - Response
  - Response measure
  - Tactics

MODIFIABILITY

- Modifiability is about the cost of change. It brings up two concerns.
  - What can change (the artifact)? A change can occur to any aspect of a system
    - the functions that the system computes
    - the platform the system exists on
    - the environment within which the system operates
    - the qualities the system exhibits
    - capacity

MODIFIABILITY

- When is the change made and who makes it (the environment)? Changes can be made to
  - the implementation (by modifying the source code)
  - during compile (using compile--time switches)
  - during build (by choice of libraries)
  - during configuration setup (by a range of techniques, including parameter setting)
  - during execution (by parameter setting).
- A change can also be made by a developer, an end user, or a system administrator.

- Once a change has been specified, the new implementation must be designed, implemented, tested, and deployed. All of these actions take time and money, both of which can be measured.

MODIFIABILITY

- Source of stimulus.
  - This portion specifies who makes the changes—the developer, a system administrator, or an end user. Clearly, there must be machinery in place to allow the system administrator or end user to modify a system, but this is a common occurrence.
- Stimulus.
  - This portion specifies the changes to be made.
- Artifact.
  - This portion specifies what is to be changed. the functionality of a system, its platform, its user interface, its environment, or another system with which it interoperates
- Environment.
  - This portion specifies when the change can be made. Design time, compile time, build time, initiation time, or runtime

MODIFIABILITY

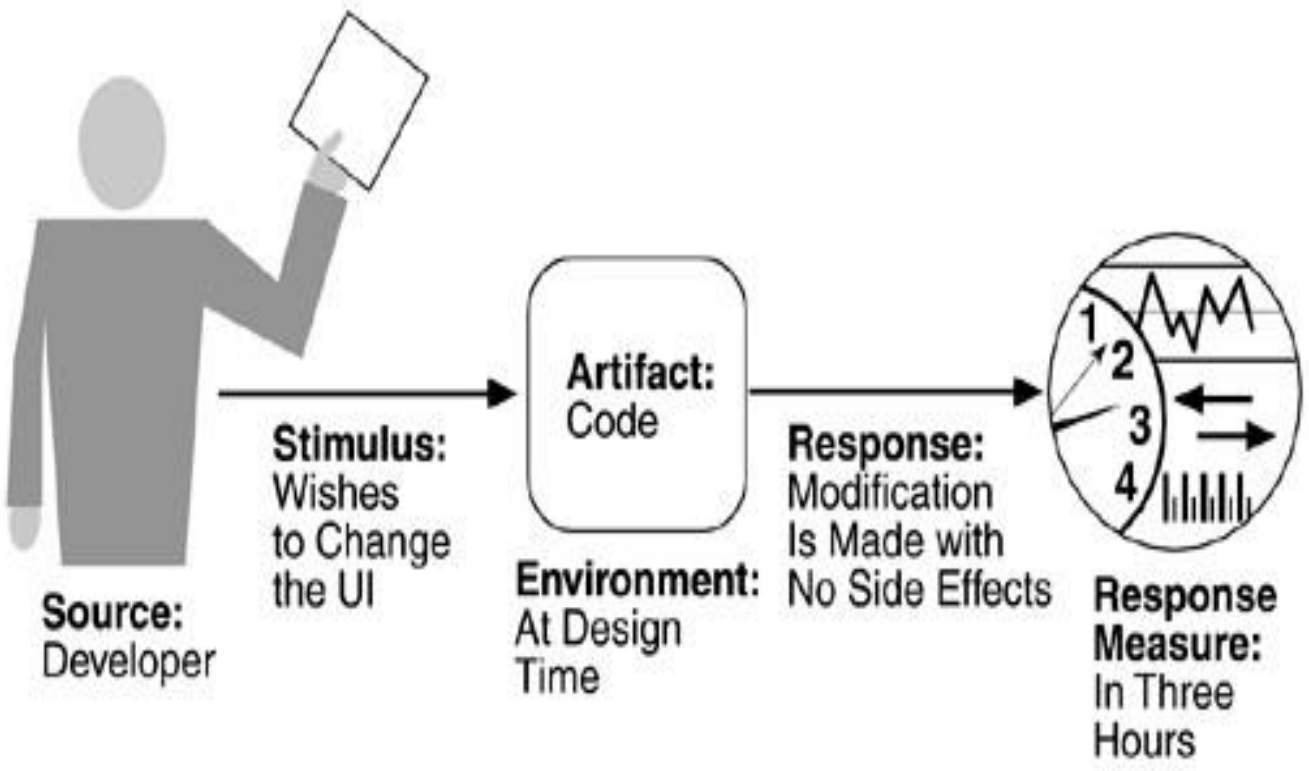
- Response.
  - Whoever makes the change must understand how to make it, and then make it, test it and deploy it.

- Response measure.
  - All of the possible responses take time and cost money, and so time and cost are the most desirable measures. Time is not always possible to predict, however, and so less ideal measures are frequently used, such as the extent of the change (number of modules affected).

MODIFIABILITY

Portion of Scenario	Possible Values
Source	End user, developer, system administrator
Stimulus	Wishes to add/delete/modify/vary functionality, quality attribute, capacity
Artifact	System user interface, platform, environment; system that interoperates with target system
Environment	At runtime, compile time, build time, design time
Response	Locates places in architecture to be modified; makes modification without affecting other functionality; tests modification; deploys modification
Response Measure	Cost in terms of number of elements affected, effort, money; extent to which this affects other functions or quality attributes

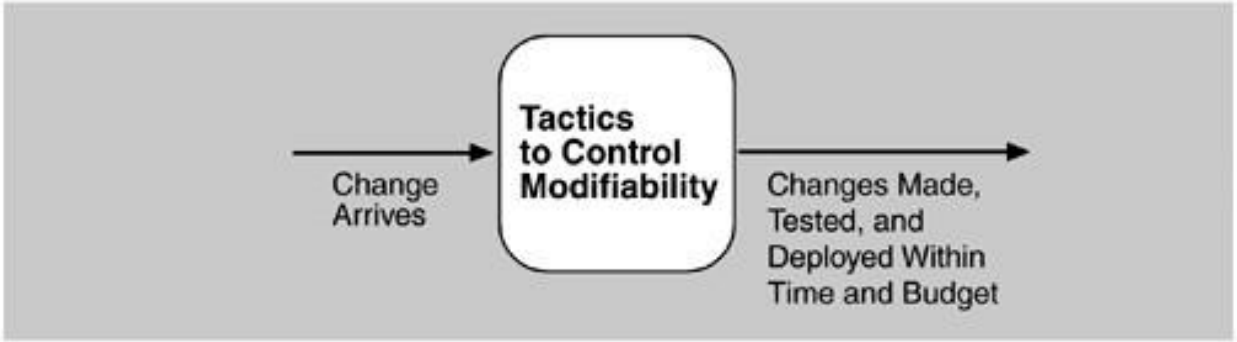
MODIFIABILITY



MODIFIABILITY

- Tactics to control modifiability have as their goal controlling the time and cost to implement, test, and deploy changes.

Goal of modifiability tactics



MODIFIABILITY

- We organize the tactics for modifiability in sets according to their goals.
  - One set has as its goal reducing the number of modules that are directly affected by a change. We call this set "localize modifications."
  - A second set has as its goal limiting modifications to the localized modules. We use this set of tactics to "prevent the ripple effect."
  - Implicit in this distinction is that there are modules directly affected (those whose responsibilities are adjusted to accomplish the change) and modules indirectly affected by a change (those whose responsibilities remain unchanged but whose implementation must be changed to accommodate the directly affected modules).
  - A third set of tactics has as its goal controlling deployment time and cost. We call this set "defer binding time."

Modifiability Tactics-localize modifications

- The goal of tactics in this set is to assign responsibilities to modules during design such that anticipated changes will be limited in scope. We identify the following tactics.

- Maintain semantic coherence.
  - semantic coherence should be measured against a set of anticipated changes. One sub--tactic is to abstract common services.
    语义连贯性应该用一组预期的变化来衡量。一种策略是抽象公共服务

- Anticipate expected changes.
  - The tactic of anticipating expected changes does not concern itself with the coherence of a module's responsibilities but rather with minimizing the effects of the changes.
  - In reality this tactic is difficult to use by itself since it is not possible to anticipate all changes. For that reason, it is usually used in conjunction with semantic coherence.

Modifiability Tactics-localize modifications

- Generalize the module. 泛化
  - Making a module more general allows it to compute a broader range of functions based on input.
  - The more general a module, the more likely that requested changes can be made by adjusting the input language rather than by modifying the module.

- Limit possible options.
  - Modifications, especially within a product line may be far ranging and hence affect many modules.
  - Restricting the possible options will reduce the effect of these modifications. For example, a variation point in a product line may be allowing for a change of processor. Restricting processor changes to members of the same family limits the possible options.

Modifiability Tactics-prevent the ripple effect

- A ripple effect from a modification is the necessity of making changes to modules not directly affected by it.

- We begin our discussion of the ripple effect by discussing the various types of dependencies that one module can have on another. We identify eight types: 我们通过讨论一个模块对另一个模块的各种依赖类型来开始对连锁反应的讨论。我们确定了八种类型：

语法化

1. Syntax of

谁使用谁负责

  - data. For B to compile (or execute) correctly, the type (or format) of the data that is produced by A and consumed by B must be consistent with the type (or format) of data assumed by B. 数据。为了使B正确地编译(或执行)，由A产生并由B使用的数据的类型(或格式)必须与B假设的数据的类型(或格式)一致。
  - service. For B to compile and execute correctly, the signature of services provided by A and invoked by B must be consistent with the assumptions of B. 服务。对于B,正确地编译和执行,由B提供的服务的签名必须与B的假设相一致

Modifiability Tactics-prevent the ripple effect

2.Semantics of

- data. For B to execute correctly, the semantics of the data produced by A and consumed by B must be consistent with the assumptions of B.
- service. For B to execute correctly, the semantics of the services produced by A and used by B must be consistent with the assumptions of B.

3.Sequence of

- data. For B to execute correctly, it must receive the data produced by A in a Hixed sequence. For example, a data packet's header must precede its body in order of reception (as opposed to protocols that have the sequence number built into the data).
- control. For B to execute correctly, A must have executed previously within certain timing constraints. For example, A must have executed no longer than 5ms before B executes.

Modifiability Tactics-prevent the ripple effect

4. Identity of an interface of A.
   - A may have multiple interfaces. For B to compile and execute correctly, the identity (name or handle) of the interface must be consistent with the assumptions of B.

5. Location of A (runtime).
   - For B to execute correctly, the runtime location of A must be consistent with the assumptions of B. For example, B may assume that A is located in a different process on the same processor.

6. Quality of service/data provided by A.
   - For B to execute correctly, some property involving the quality of the data or service provided by A must be consistent with B's assumptions. For example, data provided by a particular sensor must have a certain accuracy in order for the algorithms of B to work correctly.

Modifiability Tactics-prevent the ripple effect

7. Existence of A.
   - For B to execute correctly, A must exist. For example, if B is requesting a service from an object A, and A does not exist and cannot be dynamically created, then B will not execute correctly.

8. Resource behavior of A.
   - For B to execute correctly, the resource behavior of A must be consistent with B's assumptions. This can be either resource usage of A (A uses the same memory as B) or resource ownership (B reserves a resource that A believes it owns).

- With this understanding of dependency types, we can now discuss tactics available to the architect for preventing the ripple effect for certain types.

Modifiability Tactics-prevent the ripple effect

- Notice that none of our tactics necessarily prevent the ripple of semantic changes.

- Hide information.
  - This is the oldest technique for preventing changes from propagating.
  - It is strongly related to "anticipate expected changes" because it uses those changes as the basis for decomposition.

-这是防止更改传播的最古老的技术。
-它与“预期预期的变更”密切相关，因为它使用那些变更作为分解的基础。

- Maintain existing interfaces.
  - If B depends on the name and signature of an interface of A, maintaining this interface and its syntax allows B to remain unchanged.

如果B依赖于A的接口的名称和签名，则维护该接口及其语法允许B保持不变

Modifiability Tactics-prevent the ripple effect

- Patterns that implement this tactic include
  - adding interfaces. Most programming languages allow multiple interfaces. Newly visible services or data can be made available through new interfaces, allowing existing interfaces to remain unchanged and provide the same signature.
  - adding adapter. Add an adapter to A that wraps A and provides the signature of the original A.
  - providing a stub A. If the modiHication calls for the deletion of A, then providing a stub for A will allow B to remain unchanged if B depends only on A's signature.

· 添加接口。大多数编程语言允许多个接口。可以通过新接口提供新的可见服务或数据，从而允许现有接口保持不变并提供相同的签名。·
· 添加适配器。向A添加一个适配器，该适配器包装A并提供原始A的签名。
· 提供一个存根a。如果修改要求删除a，那么如果B只依赖于a的签名，那么为a提供一个存根将允许B保持不变

Modifiability Tactics-prevent the ripple effect

- Restrict communication paths.
  - Restrict the modules with which a given module shares data. That is, reduce the number of modules that consume data produced by the given module and the number of modules that produce data consumed by it.
  - This will reduce the ripple effect since data production/consumption introduces dependencies that cause ripples. (Flight Simulation) discusses a pattern that uses this tactic.

- Use an intermediary.
  - If B has any type of dependency on A other than semantic, it is possible to insert an intermediary between B and A that manages activities associated with the dependency.
  - All of these intermediaries go by different names, but we will discuss each in terms of the dependency types we have enumerated.
  - As before, in the worst case, an intermediary cannot compensate for semantic changes. The intermediaries are

Modifiability Tactics-prevent the ripple effect

- data (syntax).
  - Repositories (both blackboard and passive) act as intermediaries between the producer and consumer of data. The repositories can convert the syntax produced by A into that assumed by B. Some publish/subscribe patterns (those that have data Hlowing through a central component) can also convert the syntax into that assumed by B. The MVC and PAC patterns convert data in one formalism (input or output device) into another (that used by the model in MVC or the abstraction in PAC).
- service (syntax).
  - The facade, bridge, mediator, strategy, proxy, and factory patterns all provide intermediaries that convert the syntax of a service from one form into another. Hence, they can all be used to prevent changes in A from propagating to B.
- identity of an interface of A.
  - A broker pattern can be used to mask changes in the identity of an interface. If B depends on the identity of an interface of A and that identity changes, by adding that identity to the broker and having the broker make the connection to the new identity of A, B can remain unchanged.

Modifiability Tactics-prevent the ripple effect

- location of A (runtime).
  - A name server enables the location of A to be changed without affecting B. A is responsible for registering its current location with the name server, and B retrieves that location from the name server.
- resource behavior of A or resource controlled by A.
  - A resource manager is an intermediary that is responsible for resource allocation. Certain resource managers (e.g., those based on Rate Monotonic Analysis in real--time systems) can guarantee the satisfaction of all requests within certain constraints. A, of course, must give up control of the resource to the resource manager.
- existence of A.
  - The factory pattern has the ability to create instances as needed, and thus the dependence of B on the existence of A is satisHied by actions of the factory.

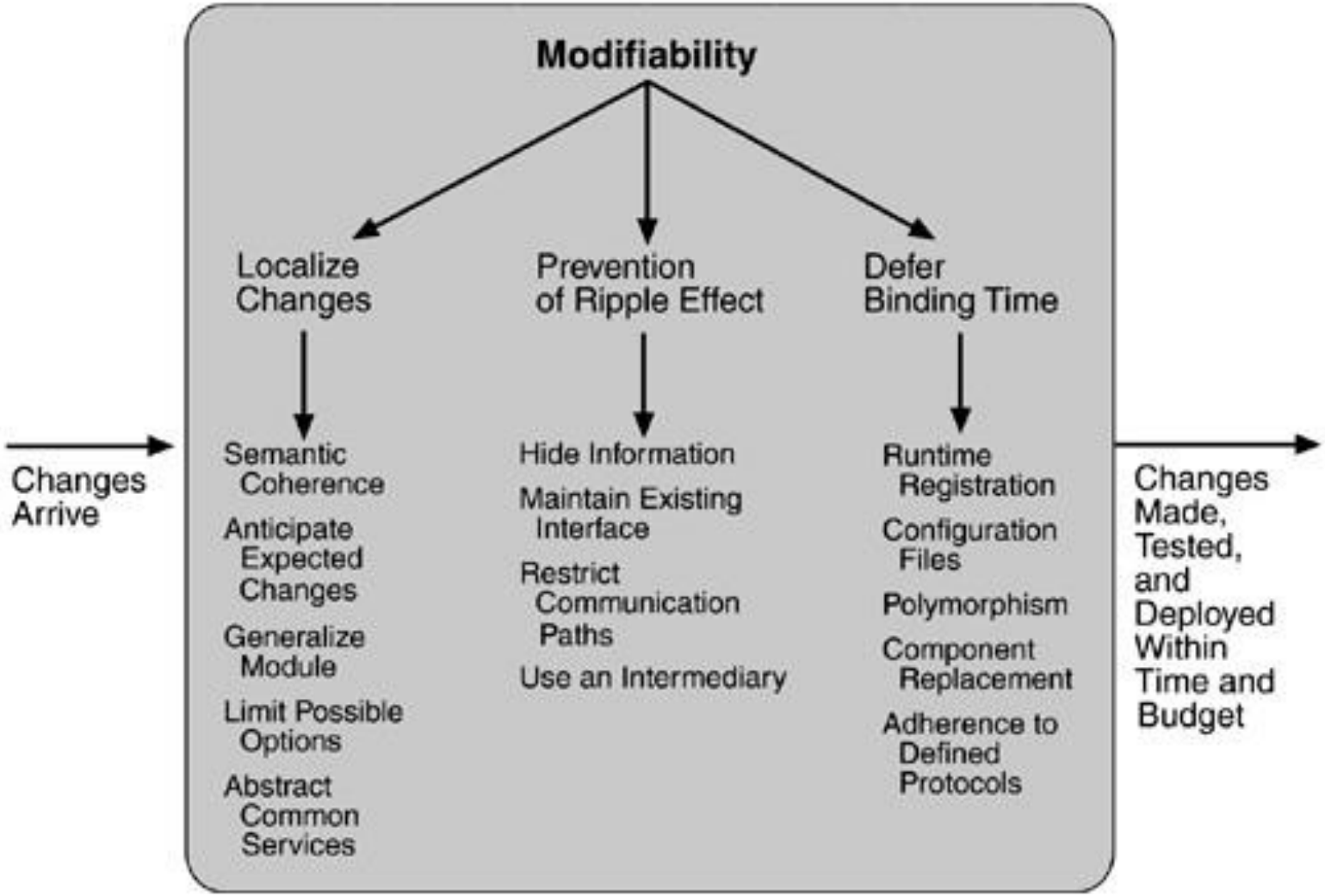
Modifiability Tactics-defer binding time

- Our modifiability scenarios include two elements that are not satisHied by reducing the number of modules to be changed
  - time to deploy and allowing non--developers to make changes.

- Many tactics are intended to have impact at lead--time or runtime, such as the following.
  - Runtime registration supports plug--and--play operation at the cost of additional overhead to manage the registration. Publish/subscribe registration, for example, can be implemented at either runtime or load time.
  - Configuration Files are intended to set parameters at startup.
  - Polymorphism allows late binding of method calls.
  - Component replacement allows load time binding.
  - Adherence to deHined protocols allows runtime binding of independent processes

- 运行时注册支持即插即用操作，但需要额外的开销来管理注册。例如，发布/订阅注册可以在运行时或加载时实现。
- 配置文件用于在启动时设置参数。
- 多态允许方法调用的后期绑定。
- 组件替换允许加载时间绑定。
- 遵守dehining协议允许独立进程的运行时绑定

Modifiability Tactics-Summary



SECURITY

- Security is a measure of the system's ability to resist unauthorized usage while still providing its services to legitimate users.
  - An attempt to breach security is called an attack and can take a number of forms.

- Security can be characterized as a system providing nonrepudiation, confidentiality, integrity, assurance, availability, and auditing. For each term, we provide a definition and an example.

SECURITY

- Nonrepudiation
  - is the property that a transaction (access to or modiHication of data or services) cannot be denied by any of the parties to it.
  - This means you cannot deny that you ordered that item over the Internet if, in fact, you did.

- Confidentiality
  - is the property that data or services are protected from unauthorized access.
  - This means that a hacker cannot access your income tax returns on a government computer.

SECURITY

- Integrity
  - is the property that data or services are being delivered as intended.
  - This means that your grade has not been changed since your instructor assigned it.

- Assurance
  - is the property that the parties to a transaction are who they purport to be.
  - This means that, when a customer sends a credit card number to an Internet merchant, the merchant is who the customer thinks they are.

SECURITY

- Availability
  - is the property that the system will be available for legitimate use.
  - This means that a denial--of--service attack won't prevent your ordering this book.

- Auditing
  - is the property that the system tracks activities within it at levels sufHicient to reconstruct them.
  - This means that, if you transfer money out of one account to another account, in Switzerland, the system will maintain a record of that transfer.

SECURITY

- Source of stimulus.
  - The source of the attack may be either a human or another system. It may have been previously identiHied (either correctly or incorrectly) or may be currently unknown.
  - The attack itself is unauthorized access, modiHication, or denial of service.
- Stimulus.
  - The stimulus is an attack or an attempt to break security.
  - We characterize this as an unauthorized person or system trying to display information, change and/or delete information, access services of the system, or reduce availability of system services.
- Artifact.
  - The target of the attack can be either the services of the system or the data within it.
- Environment.
  - The attack can come when the system is either online or ofHline, either connected to or disconnected from a network, either behind a Hirewall or open to the network.

SECURITY

- Response.
  - Using services without authorization or preventing legitimate users from using services is a different goal from seeing sensitive data or modifying it. Thus, the system must authorize legitimate users and grant them access to data and services, at the same time rejecting unauthorized users, denying them access, and reporting unauthorized access.

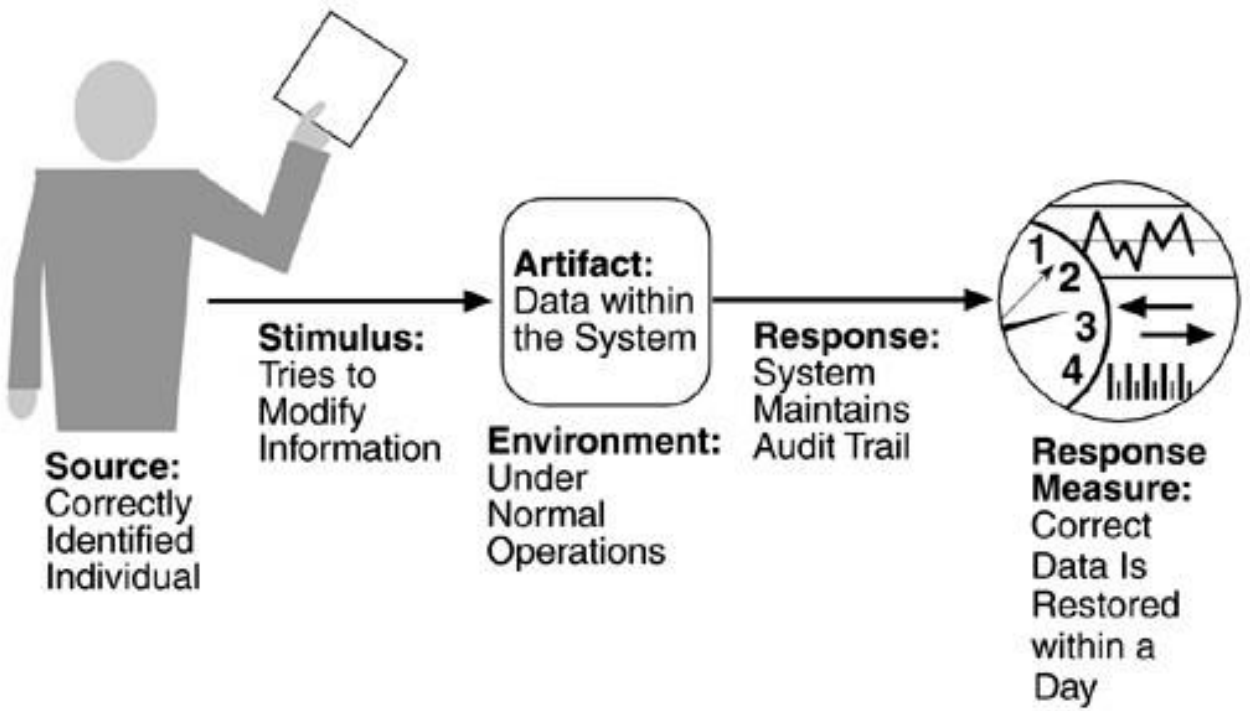
- Response measure.
  - Measures of a system's response include the difficulty of mounting various attacks and the difficulty of recovering from and surviving attacks.

SECURITY

Portion of Scenario	Possible Values
Source	Individual or system that is correctly identified, identified incorrectly, of unknown identity who is internal/external, authorized/not authorized with access to limited resources, vast resources
Stimulus	Tries to display data, change/delete data, access system services, reduce availability to system services
Artifact	System services; data within system
Environment	Either online or offline, connected or disconnected, fire-walled or open

Response	**Authenticates user; hides identity of the user; blocks access to data and/or services; allows access to data and/or services; grants or withdraws permission to access data and/or services; records access/modifications or attempts to access/modify data/services by identity; stores data in an unreadable format; recognizes an unexplainable high demand for services, and informs a user or another system, and restricts availability of services**
Response Measure	**Time/effort/resources required to circumvent security measures with probability of success; probability of detecting attack; probability of identifying individual responsible for attack or access/modification of data and/or services; percentage of services still available under denial-of-services attack; restore data/services; extent to which data/services damaged and/or legitimate access denied**

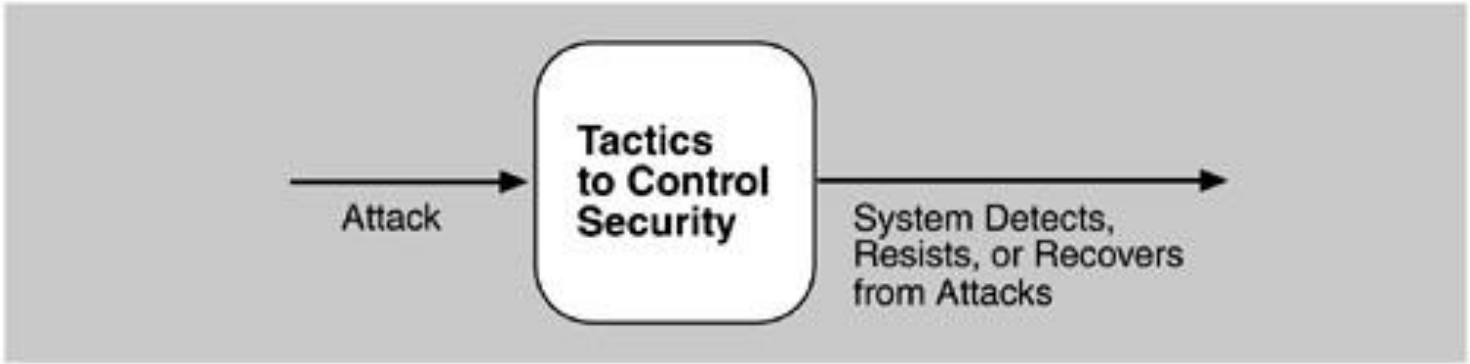
SECURITY


Source:
Correctly
Identified
Individual
Stimulus:
Tries to
Modify
Information
Artifact:
Data within
the System
Environment:
Under
Normal
Operations
Response:
System
Maintains
Audit Trail
1
2
3
4
Response
Measure:
Correct
Data Is
Restored
within a
Day

Security Tactics

- Tactics for achieving security can be divided into those concerned with resisting attacks, those concerned with detecting attacks, and those concerned with recovering from attacks. All three categories are important. 实现安全的策略可以分为抵抗攻击的策略、检测攻击的策略和从攻击中恢复的策略。这三个方面都很重要
- Using a familiar analogy, putting a lock on your door is a form of resisting an attack, having a motion sensor inside of your house is a form of detecting an attack, and having insurance is a form of recovering from an attack.

Goal of security tactics



Security Tactics-resisting attacks

- we identified
  - nonrepudiation, confidentiality, integrity, and assurance as goals in our security characterization.

- The following tactics can be used in combination to achieve these goals.
  - Authenticate users.
  - Authorize users.
  - Maintain data confidentiality.
    - Encryption
    - Communication links
      - virtual private network (VPN)
      - Secure Sockets Layer (SSL)
  - Maintain integrity.
    - checksums
    - hash results
  - Limit exposure
  - Limit access
    - Firewalls

Security Tactics-detecting attacks

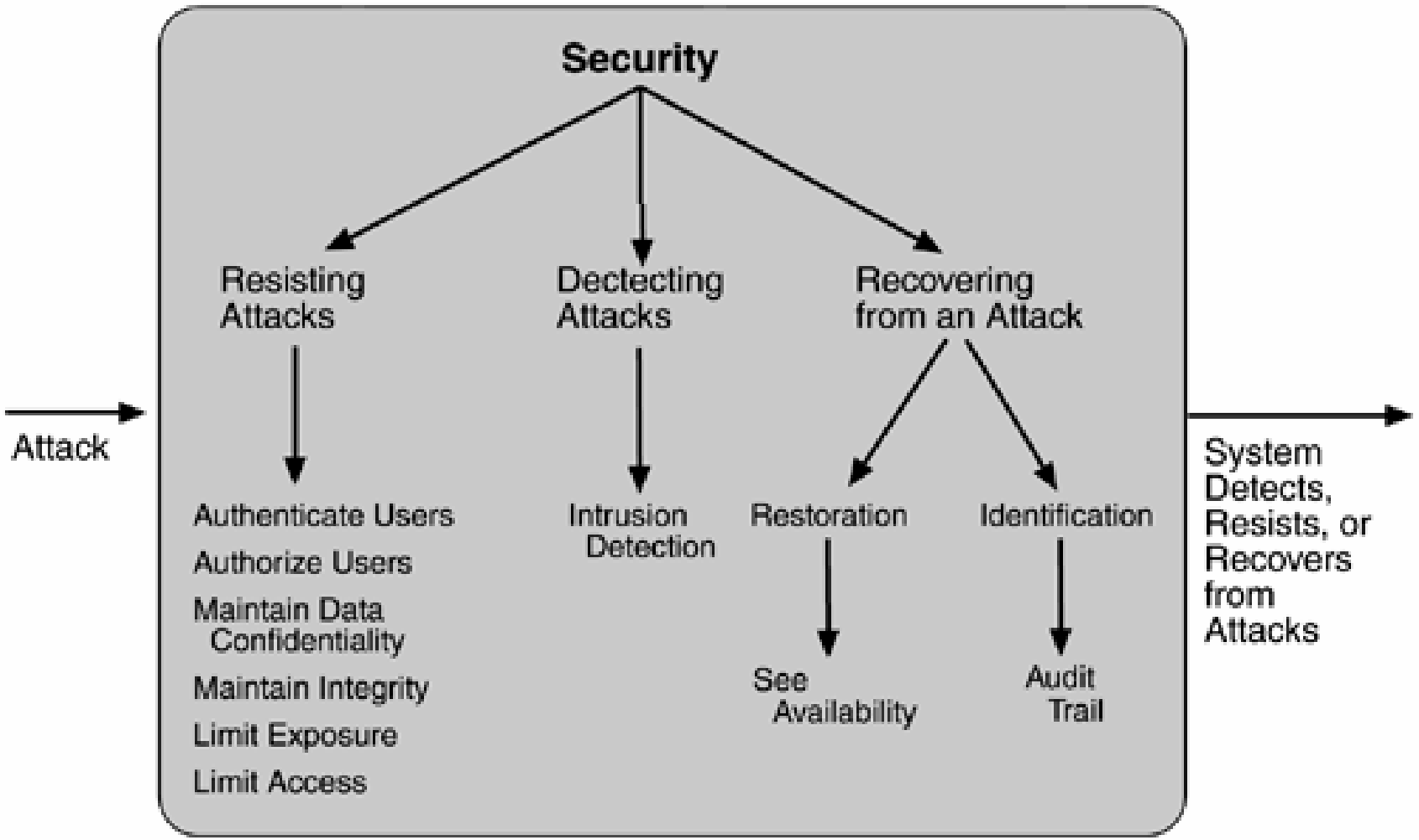
- The detection of an attack is usually through an intrusion detection system.
  - Such systems work by comparing network traffic patterns to a database. In the case of misuse detection, the traffic pattern is compared to historic patterns of known attacks. In the case of anomaly detection, the trafHic pattern is compared to a historical baseline of itself. Frequently, the packets must be filtered in order to make comparisons. Filtering can be on the basis of protocol, TCP flags, payload sizes, source or destination address, or port number.

- Intrusion detectors must have some sort of sensor to detect attacks, managers to do sensor fusion, databases for storing events for later analysis, tools for offline reporting and analysis, and a control console so that the analyst can modify intrusion detection actions.

Security Tactics-recovering from attacks

- Tactics involved in recovering from an attack can be divided into those concerned with restoring state and those concerned with attacker identification.
- The tactics used in restoring the system or data to a correct state overlap with those used for availability since they are both concerned with recovering a consistent state from an inconsistent state.
- The tactic for identifying an attacker is to maintain an audit trail.

Security Tactics-Summary



Project

- Suppose you needs to improve the modifiability & security of your SNS website. Please describe your design from the following aspects:
  - If your SNS provides API for users to develop new services and plug them into your system, what tactics would you want to adopted to support such a high modifiability?
  - If your want to prove the security of your SNS website to users, please tell them what tactics you have adopted to improve the security in the aspects of Nonrepudiation, Confidentiality and Assurance.