The Biba Security Model

By:
Nathan Balon
Ishraq Thabet

March 17, 2004
Winter 2004
CIS 576

Introduction

Computer security is concerned with three aspects: confidentiality, integrity, and availability. Organizations implement security in accordance with their needs. An organization creates a security policy and uses security mechanisms to enforce the policy. A security policy is a statement that partitions the states of the system into a set of authorized or secure states and a set of unauthorized or unsecured states (Bishop). The goal of an information system is to control access to the subjects and objects in the system. A security policy governs a set of rules and objectives needed by an organization (Blake).

A security model can be used by an organization to help express the policy or business rules to be used in a computer system. Over time, various security models have been developed. The most common type of model is access control, which prevents the unauthorized use of a resource (Stallings). Access control is typically defined in one of two ways, either discretionary or mandatory access control. First, discretionary access control (DAC) is user-based. DAC gives ownership to the objects in the system. The owners of objects can in turn give access to others to use these objects. This model allows the most flexibility, but is the hardest to maintain. Second, in mandatory access control (MAC), objects are given a classification level. Each user in the system is given a clearance level. Users are then allowed to view objects based on their clearance level.

Access control forms the foundation for a security policy for an organization. The Bell-LaPadula model was one of the first models developed to control access to data in a computer system by guaranteeing confidentiality of the data. However, there are some shortcomings to this model. If an organization uses solely access control to enforce the business rules of an organization, they will not take into account other issues such as integrity or availability. The Biba model was created to deal with the deficiency of controlling integrity.

Bell-LaPadula Model

The Bell-LaPadula model is a classical model used to define access control. The model is based on a military-style classification system (Bishop). With a military model, the sole goal is to prevent information from being leaked to those who are not privileged to access the information. The Bell-LaPadula was developed at the Mitre Corporation, a government funded organization, in the 1970's (Cohen).

The Bell-LaPadula is an information flow security model because it prevents information to flow from a higher security level to a lower security level. The Bell-LaPadula model is based around two main rules: the simple security property and the star property. The simple security property states that a subject can read an object if the object's classification is less than or equal to the subject's clearance level. The simple security property prevents subjects from reading more privileged data. The star property states that a subject can write to an object, if the subject's clearance level is less than or equal to the object's classification level. What the star property essentially does is it prevents the

lowering of the classification level of an object. The properties of the Bell-LaPadula model are commonly referred to as "no read up" and "no write down", respectively.

The Bell La-Padula model is not flawless. Specifically, the model does not deal with the integrity of data. It is possible for a lower level subject to write to a higher classified object. Because of these short comings, the Biba model was created. The Biba model in turn is deeply rooted in the Bell La-Padula model.

Integrity

Integrity deals with the correctness of data. According to Matt Bishop, "integrity refers to the trustworthiness of data or resources, and it is usually phrased in terms of preventing improper or authorized change". Bishop also gives the following goals of integrity:

1. Preventing unauthorized users from making modifications to data or programs.
2. Preventing authorized users from making improper or unauthorized modifications.
3. Maintaining internal and external consistency of data and programs.

A military model such as the Bell-LaPadula works well in some environments and not so well in others. In a commercial environment, the integrity of the data is often more important than who can actually view it. What good is it to store large amounts of data in a database if that data is not correct? The data in a database is virtually worthless if its integrity is not maintained. For instance, a bank would require the strict enforcement of integrity. It would be catastrophic if the customers of a bank were able to make unauthorized alterations to an account. They would be able to credit their accounts with funds that would be unaccounted for. There are numerous other examples of how crucial it is that the integrity of data is maintained. In many cases, integrity is more important than confidentiality.

Various security models have been created to enforce integrity. Some of the more popular models that have been proposed to enforce integrity are Biba Model, Lipner's Integrity Matrix Model, and Clark-Wilson Model. Each of these models takes a different approach to supporting integrity.

Biba Model

The Biba integrity model was published in 1977 at the Mitre Corporation, one year after the Bell La-Padula model (Cohen). As stated before, the Bell-LaPadula model gurantees confidentiality of data but not its integrity. As a result, Biba created a model to address the need of enforcing integrity in a computer system. The Biba model proposed a group of integrity policies that can be used. So, the Biba model is acutally family of different integrity policies. Each of the policies uses different conditions to ensure information integrity (Castano). The Biba model, in turn, uses both discretionary and non-discretionary policies.

The Bell La-Padula model uses labels to give subjects clearance levels and objects classification levels. Similarly, the Biba model also uses labels to define security, but it takes a different approach. The Biba model uses labels to give integrity levels to the subjects and objects. The data marked with a high level of integrity will be more accurate and reliable than data labeled with a low integrity level. The integrity levels are in turn used to prohibit the modification of data.

Labels

In a computer system there are a set of subjects and a set of objects. Subjects are the active components in the system such as processes created by the users. On the other hand, objects are a set of protected entities in the system such as files. The Biba model requires that each subject and object is given an integrity label. The labels by themselves do not provide protection to data. The labels must be complemented with a security mechanism in order to provide protection (RFC 1457). Usually the level of a security label remains constant, but there are exceptions to this rule; some of the policies in the Biba model support dynamic labels. The Biba model can use both static and dynamic labels. Dynamic labels allow the integrity levels to vary.

An integrity label consists of two parts, a classification and a set of categories. The classification of integrity forms a hierarchical set. For example, the classification could be crucial, important, and insignificant. Crucial would have the highest classification level and insignificant would have the lowest classification level. The name for the classification levels can be what ever is chosen for the implementation as long as it is consistent. In this case: crucial > important > insignificant.

The second part of the label will consist of a set of categories also known as a compartment. The set of categories contained in the label will be a subset of all the sets in the system. The classification of the set of categories is non-hierarchical. An example of two categories are category X = {Detroit, Chicago, New York} and category Y = {Detroit, Chicago}. In this case $X \geq Y$ (X dominates Y), because Y is a subset of X. If there were a third compartment Z containing {Detroit, Chicago, Miami}, compartment Z and X in this case are non-comparable because the third element in the sets are different (Frost).

Each integrity level will be represented as L = (C, S) where L is the integrity level, C is the classification and S is the set of categories. The integrity levels then form a dominance relationship. For instance, integrity level $L_1 = (C_1, S_1)$ dominates ($\geq$) integrity level $L_2 = (C_2, S_2)$ if and only if this relationship is satisfied: $C_1 \geq C_2$ and $S_1 \supseteq S_2$ (Castano).

Integrity labels tell the degree of confidence that may be placed in the data (RFC 1457). The confidence placed in data never increases but it is possible for the integrity level of an object to decrease. For example, if a data of high integrity level is sent across a network that has a low integrity, the trust in the data will not be the same as before it was sent across the network. The integrity level of the data could be lowered as a result of

this. The data in this instance would then be relabeled to a lower classification level. The Biba model has a number of low-watermark policies that enforce this dynamic labeling.

In the case of a networked system, each system in the network must support the use of integrity labels. If some systems in the network do not support integrity labels there is a loss of confidence in the integrity of the data. Currently, there are no network protocols that support integrity labeling (RFC 1457). This creates a problem of using integrity labels in a network environment.

Access Modes

The Biba Model consists of group access modes. The access modes are similar to those used in other models, although the may use different terms to define them. The access modes that the Biba model supports are:

1. **Modify**: allows a subject to write to an object. This mode is similar to the write mode in other models.
2. **Observe**: allows a subject to read an object. This command is synonyms with the read command of other models.
3. **Invoke**: allows a subject to communicate with another subject.
4. **Execute**: allows a subject to execute an object. The command essentially allows a subject to execute a program which is the object.

Policies Supported by the Biba Model

The Biba model can be divided into two types of policies, those that are mandatory and those that are discretionary. Within these two divisions, there are a number of policies that can be selected based on the security needs. Most literature on the Biba model refers to the model as being the Strict Integrity Policy, although there are a number of other policies that can be used in the model.

Mandatory Policies:

1. Strict Integrity Policy
2. Low-Water-Mark Policy for Subjects
3. Low-Water-Mark Policy for Objects
4. Low-Water-Mark Integrity Audit Policy
5. Ring Policy

Discretionary Policies:

1. Access Control Lists
2. Object Hierarchy
3. Ring

Mandatory Biba Policies

The Strict Integrity Policy is the first part of the Biba model. The policy states:

1. Simple Integrity Condition: $s \in S$ can observe $o \in O$ if and only if $i(s) \leq i(o)$.
2. Integrity Star Property: $s \in S$ can modify to $o \in O$ if and only if $i(o) \leq i(s)$.
3. Invocation Property: $s_1 \in S$ can invoke $s_2 \in S$ if and only if $i(s_2) \leq i(s_1)$.

The Strict Integrity Policy is the most popular policy in the Biba model. The first part of the policy is known as the simple integrity property. The property states that a subject may observe an object only if the integrity level of the subject is less than the integrity level of the object. The second rule of the strict integrity property is the integrity star property. This property states that a subject can write to an object only if the objects integrity level is less than or equal to the subject's level. This rule prevents a subject from writing to a more trusted object. The last rule is the invocation property, which states that a subject s_1 can only invoke another subject s_2, if s_2 has a lower integrity level than s_1.

The strict integrity policy enforces "no write-up" and "no read-down" on the data in the system, which is the opposite of the Bell-LaPadula model. This policy restricts the contamination of data at higher level, since a subject is only allowed to modify data at their level or a low level. The "no write up" is essential since it limits the damage that can be done by malicious objects in the system. For instance, "no write up" limits the amount of damage that is done by a Trojan horse in the system. If the malicious code was hidden in a subject it would only be able to write to objects at its integrity level or lower. This is important because it limits the damage that can be done to the operating system. The "no read down" prevents a trusted subject from being contaminated by a less trusted object.

The strict integrity property is the most restrict of the policies that make up the Biba model. The strict integrity property succeeds at enforcing integrity in a system, but it is not without its weaknesses. Specifically, the strict integrity property restricts the reading of lower level objects which may be too restrictive in some cases. To combat this problem, Biba devised a number of dynamic integrity polices that would allow trusted subjects access to an un-trusted objects or subjects. Biba implemented these in a number of different low-water mark policies.

The low-watermark policy for subjects is the second part of the Biba model. The policy states:

1. Integrity Star Property: $s \in S$ can modify $o \in O$ if and only if $i(o) \leq i(s)$.
2. If $s \in S$ examines $o \in O$ the $i'(s) = min(i(s),i(o))$, where $i'(s)$ is the subjects integrity level after the read.
3. Invocation Property: $s_1 \in S$ can invoke $s_2 \in S$ if and only if $i(s_2) \leq i(s_1)$.

What the low-watermark policy for subjects in essence does is lowers the integrity level of the subject to the lowest integrity level of the subject and object involved. The first rule of this policy is integrity star property which enforces "no write up". This prevents the modification of more trusted objects. The second rule of the policy states that if a subject is to read a less trusted object, the integrity level of the subject will drop to that of object. This prevents an object from contaminating a subject because the subject's integrity level will be reduced to that of the object. Last, the low-watermark policy for subject uses the invocation property.

The low-watermark policy for subjects is a dynamic policy because it lowers the integrity level of a subject based on the observations of objects. This policy is not without its problems. One problem with this policy is if a subject observes a lower integrity object it will drop the subject's integrity level. Then, if the subject needs to legitimately observe another object it may not be able to do so because the subject's integrity level has been lowered. Depending on the times of read requests by the subject, to observe the objects, a denial of service could develop.

The low-watermark policy for objects is the third part of the Biba model. This policy is similar to the low-watermark policy for subject. The policy states:

1. $s \in S$ can modify any $o \in O$ regardless of integrity level.
2. If $s \in S$ observe $o \in O$ the $i'(o) = min(i(s),i(o))$, where $i'(o)$ is the objects integrity level after it is modified.

This policy allows any subject to modify any object. The objects integrity level is then lowered if the subject's integrity level is less than the objects. This policy is also dynamic because the integrity levels of the objects in the system are changed based on what subjects modify them. This policy does nothing to prevent an un-trusted subject from modifying a trusted object. This policy is not very practical. The policy provides no real protection in a system, but lowers the trust placed in the objects. If a malicious program was inserted into the computer system, it could modify any object in the system. The result would be to lower the integrity level of the infected object. It is possible with this policy that, overtime, there will be no more trusted objects in the system because their integrity level has been lowered by subjects modifying them.

The low-watermark integrity audit policy is the fourth mandatory policy under the Biba model. The policy states:

1. $s \in S$ can modify any $o \in O$, regardless of integrity levels.
2. If a subject modifies a higher level object the transaction is recorded in an audit log.

This policy is similar to the low-watermark policy for objects. Like the low-watermark policy for objects it does nothing to prevent the improper modification of an object. The low-watermark integrity audit policy simply records that an improper modification has taken place. The audit log must then be examined to determine the cause of the improper modification. The drawback to this policy is that it does nothing to prevent an improper modification of an object to occur.

The Ring Policy is the last mandatory policy in the Biba Model. This policy is not dynamic like the first three policies. Integrity labels used for the ring policy are fixed, similar to those in the strict integrity policy. The Ring Policy states:

1. Any subject can observe any object, regardless of integrity levels.
2. Integrity Star Property: $s \in S$ can modify $o \in O$ if and only if $i(o) \leq i(s)$.

3. Invocation Property: $s_1 \in S$ can invoke $s_2 \in S$ if and only if $i(s_2) \leq i(s_1)$.

The Ring Policy allows any subject to observe any object. The policy is only concerned with direct modification. A subject can write to an object only if the integrity level of object is less than or equal to the integrity level of the subject, which is the integrity star property. The last part of the ring policy is the invocation property.

The ring policy is not perfect; it allows improper modifications to take place. A subject can read a low level subject, and then modifies the data observed at its integrity level (Castano). An example of this would be, a user reading a less trusted object, then remembers the data they read and then, at a later time, writing that data to an object at their integrity level. The ring policy allows indirect modification of trusted data.

Discretionary Biba Policies

The Biba model has a number of discretionary policies. These policies are not used a much as the mandatory policies in the Biba model. Most literature on the Biba model does not even mention the discretionary policies. The book, entitled Database Security, gives three discretionary policies that make up the Biba model. The first discretionary policy is an access control list, which can be used to determine which subjects can access which objects. The access control list can then be modified by the subjects with the correct privileges. Second, integrity can be enforced by using an object's hierarchy. With this method, there is root and objects that are ancestors to the root. To access a particular object, the subject must have the observe privileges to that objects and all the other ancestor objects all the way up to the root. Last, discretionary policy is the ring policy, which numbers the rings in the system with the lower number being a higher-privilege. The access modes of the subject must fall within a certain range of values to be permitted to access an object. These are the three discretionary policies that make up the Biba model. These policies are not as common as the mandatory policies. Most literature on the Biba model fails to mention that the model contains these discretionary policies.

Current Implementations of the Biba Model

A theoretical model is of little use if it cannot be implemented. One instance of where the Biba model is currently used is in FreeBSD 5.0. The TrustedBSD MAC framework is a new kernel security framework in FreeBSD 5.0 (Watson). The kernel of FreeBSD can support MAC and various other security models which are part of the TrustedBSD MAC framework. To use this option the kernel must be properly configured by adding "`options MAC`" to the kernel configuration. Depending on the policies selected, different configurations are used. For instances, policies relying on file system or other labels may require a configuration step that involves assigning initial labels to system objects or creating a policy configuration file (Watson).

The TrustedBSD project included a number of different policies which it supports. Each of the policies is part of a separate model that can be loaded into the kernel by supplying the correct kernel option for that module. Some of the Policies that are part of TrustedBSD are:

- Biba Integrity Policy
- File System Firewall Policy
- Low-Watermark Mandatory Access Control
- Multi-Level Security Policy
- MAC Framework Test Policy

The Biba Integrity Policy is in the module mac_biba.ko. The Biba policy provides for hierarchical and non-hierchial labeling of all system objects with integrity data and the strict enforcement of information flow policy to prevent the corruption of high integrity subjects by low integrity subjects (Watson).

Advantages & Disadvantages

There are a number of benefits that come from using the Biba model. The first benefit of the model is that it is fairly easy to implement. It is no harder to implement the strict integrity policy in the Biba model, compared to the Bell-LaPadula model. Another advantage is that the Biba model provides a number of different policies that can be selected based on need. If the strict integrity property is too restricting, one of the dynamic policies could be used in its place.

The Biba model is not without its drawbacks. The first problem with this model is selecting the right policy to implement. The model gives a number of different policies that can be used. On one hand, it provides more flexibility and, on the other hand, the large number of polices can make it hard to select the right policy. Another problem is the model does nothing to enforce confidentiality. For this reason, the Biba model should be combined with another model. A model such as the Bell-LaPadula could be used to complement it. The Lipner model is one such model that has been developed to meet these requirements; it, in turn, combines both the Bell-LaPadula and Biba models together. Also, the Biba model doesn't support the granting and revocation of authorization.

Conclusion

The primary motivation for establishing the Biba Integrity Model was because the Bell-LaPadula model only controls confidentiality in the system. The Biba model was one of the first models that addressed the enforcement of integrity. The Biba model is essentially a family of different policies that can be used to enforce integrity. The policies that make up the Biba model range from being very restrictive like the strict integrity policy to offering very little security as the in the case of the low-watermark integrity audit policy. It is critical that an organization implements some type of integrity policy so that the integrity of data in their information system can be maintained.

References

Bishop, M. Computer Security: Art and Science, Addison Wesley, Boston, MA. 2003.

Blake, S. "The Clark-Wilson Security Model"
http://www.lib.iup.edu/comscisec/SANSpaper/blake.htm

Castano, S. (et. al). Database Security, Addison Wesley, Harlow, England. 1995.

Cohen, F. "Models of OS Protection" http://www.all.net/books/ ip/Chap3-3.html

Frost, J. "Access Control 2: Lecture Notes" http://cob.isu.edu/cis410/week3.htm

Landwehr, C. "Formal Models for Computer Security", Computing Surveys, Vol. 13, No. 3, September 1981.

RFC 1457. "Security Label Framework for the Internet"
http://www.ietf.org/rfc/rfc1457.txt

Stallings, W. Cryptography and Network Security: Principles and Practices
(3rd Edition) ,Prentice Hall, Upper Saddle River, NJ. (2003).

Watson, R. (et. al) "The TrustedBSD MAC Framework: Extensible Kernel Access Control for FreeBSD 5.0". Usenix Annual Technical Conference, 2003.