



# Trust Modeling for Security Architecture Development

---

*Donna Andert, Robin Wakefield, and Joel Weise,  
Professional Services Security Practice*

*Sun BluePrints™ OnLine—December 2002*



<http://www.sun.com/blueprints>

**Sun Microsystems, Inc.**  
4150 Network Circle  
Santa Clara, CA 95045 U.S.A.  
650 960-1300

Part No. 817-0775-11  
Revision 1.0, 1/22/03  
Edition: December 2002

Copyright 2002 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, California 95045 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Sun BluePrints, Solaris, Sun Fire, and Starfire are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the US and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights—Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2002 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, Californie 95045 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque enregistrée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company Ltd.

Sun, Sun Microsystems, le logo Sun, Sun BluePrints, Solaris, Sun Fire, et Starfire sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REpondre A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.



Please  
Recycle



Adobe PostScript

# Trust Modeling for Security Architecture Development

---

Information technology architects must build applications, systems, and networks that match ordinary peoples' sociological expectations of trust in terms of identity, authentication, service level agreements, and privacy. Yet the inherent insecurity of many business systems is, in fact, the failure of the underlying security architecture upon which those systems are built. In particular, deficient trust models often fail to address every layer of business, technology, people, and process. And the consequence might be an implementation with weaker security than the designer intended or expected. The trust model relies on complete requirements that include business, technical, legal, regulatory, and fiduciary requirements. We recommend that you develop a formalized trust model as part of a security architecture methodology and risk analysis for all business systems to ensure that they are protected according to their stated requirements and identified risk thresholds.

A key principle of effective security design and implementation is that security should be built into every layer of a solution rather than added as an afterthought. This Sun BluePrints™ OnLine article describes the vocabulary of trust relationships and demonstrates the practical importance of using trust modeling to formalize the threshold for risk.

This article shows how to use a trust model to accomplish the following tasks:

- Elaborate on and provide context to the other components of a security architecture
- Determine and formalize a threshold for risk
- Support the risk analysis process utilized during the development of a security architecture
- Attenuate discovered risks

---

# Understanding Trust

As with many seemingly complex concepts, a good starting point is to consider the commonplace, everyday meaning of a word. Trust is an important part of our lives and it has numerous definitions. Consider questions like the following, which we deal with regularly even if we don't formalize a model:

- What does it take to establish trust?
- How do I determine the degree of trust to assign to an individual or process?
- Would I trust a recommendation from an auto mechanic or a child care provider the same way?

## Defining Trust

According to the ITU-T X.509, Section 3.3.54, trust is defined as follows:

“Generally an entity can be said to ‘trust’ a second entity when the first entity makes the assumption that the second entity will behave exactly as the first entity expects.”

For the sake of defining trust and trust modeling relative to security architecture methodology, the following set of principles or elements are offered:

- Trust is a characteristic and quality of a security architecture.
- Trust is a balancing of liability and due diligence. For example, you must decide how much effort to expend to reduce liability to an acceptable level for a particular business proposition and stated security policy. You must establish an equilibrium of trust.
- Trust is the enabling of confidence that something will or will not occur in a predictable or promised manner. The enabling of confidence is supported by identification, authentication, accountability, authorization, and availability.
- Trust is the binding of unique attributes to a unique identity, for example, accountability. This is both a qualitative and a subjective measure of expectations regarding another’s behavior and relative to a defined security policy. Essentially, a trust relationship is established when a satisfactory level of confidence in the attributes provided by an entity is achieved.
- Trust is defined as a binary relationship, or set of compounded binary relationships, based on individual identity or unique characteristic validation. That is, trust is the establishment of a trust relationship through a validation process and the subsequent use of that relationship in some transactional context.

## Establishing Trust

To establish trust or confidence, there must be a binding of unique attributes to a unique identity, and the binding must be able to be tested satisfactorily by a relying entity. When you achieve a satisfactory level of confidence in the attributes provided by an entity, you establish a trust relationship. This element of trust is commonly called authentication.

Trust involves a binary relationship, or a set of compounded binary relationships based on validation of unique individual identity. Consider the following examples of simple trust models:

- A trusts B. (Note that this means A can validate the unique identity of B. It does not mean that B necessarily trusts A.)
- A trusts B, and B trusts A.
- A trusts B, B trusts C, therefore, A trusts C.

It is also important to note what a trust model is not. A trust model is not the particular security mechanisms utilized within a particular security architecture. Rather, it is the combination of those security mechanisms in conjunction with the security policy when they address all business, technical legal, regulatory, or fiduciary requirements to the satisfaction of a relying entity.

The examples noted here are, in effect, simple trust models. So let's proceed to the characteristics of trust that shape a trust model.

## Defining Trust Modeling

A security architecture based on an acceptable trust model provides a framework for delivering security mechanisms. Trust modeling is the process performed by the security architect to define a complementary threat profile and trust model based on a use-case-driven data flow analysis. The result of the exercise integrates information about the threats, vulnerabilities, and risk of a particular information technology architecture. Further, trust modeling identifies the specific mechanisms that are necessary to respond to a specific threat profile. The data flow analysis process is further explained and diagrammed in “Example of a Data Flow Analysis” on page 11.

To provide a baseline, we define trust modeling as follows:

- A trust model identifies the specific mechanisms that are necessary to respond to a specific threat profile.
- A trust model must include implicit or explicit validation of an entity's identity or the characteristics necessary for a particular event or transaction to occur.

## Gradients of Trust

The purpose of a trust model is to respond to a specific threat profile. A threat profile is the set of threats and vulnerabilities identified through a use-case-driven data flow analysis that is particular to an organization. Essentially, a threat profile identifies likely attackers and what they want.

The level of trust necessary for one organization or circumstance may be different from the level of trust required by another organization or circumstance. For example, the level of assurance that an organization needs regarding the authentication of a user may be different in particular use cases.

Trust exists on a gradient—there is no one-size-fits-all solution. To illustrate this point, consider the requirements of two institutions: a public library and a financial institution.

- The library may implement a minimal trust model for those who simply want to browse the stacks. Perhaps the only check will be upon exit to ensure books are not being stolen. However, the library will probably implement a somewhat more stringent model for those who want to check out books. Commonly, the library will issue a library card with your name on it, and perhaps with your picture, to increase their trust that you are who you say you are.

There may be specialized collections (for example, rare manuscripts) that are accessible only to qualified researchers. Your library card serves to authenticate you, but you need additional privileges beyond what is standard. Granting specific access privileges is the process called authorization. A possible requirement for this authorization would be proof that you are on the staff of a recognized educational institution.

- Now, consider a financial institution that is entrusted to maintain a great deal of personal information about clients and that is heavily regulated by various governmental agencies. Clearly, a more robust trust model is in order.

For example, if a customer wants to use a home banking application to review his account, before the bank's application transmits the customer's data, it must validate the customer's identity. Commonly, banks require a login, supplemented by a personal identification number (PIN). But once a customer has been authenticated, the bank must take extra steps to protect the confidentiality of the customer's data, perhaps by establishing an encrypted tunnel between the bank's distributing application and the customer's home banking application.

These examples illustrate three points that are essential to understanding trust and trust models.

- Trust requirements must be matched to the specific kinds of threats or vulnerabilities facing an organization and to the degree of risk that the threats will occur.

- There must be a starting point in establishing credentials for identity. In our example, the library may have accepted a driver's license as the credential for granting a library card. What credential does the issuing authority of a driver's license require for granting a driver's license?
- Trust does not happen spontaneously. It requires a methodical process of credential establishment and consistent validation. Trust is not free, it takes capital and effort.

## Threat Profile and Risk Analysis

Threat profiles and risk analyses are intrinsically related. One without the other is of limited value.

Threat profiles identify the specific threats that are most likely to put your environment at risk.

The most common types of threats fall into categories such as:

- Actual or attempted unauthorized probing of any system or data
- Actual or attempted unauthorized access
- Introduction of viruses or malicious code
- Unauthorized modification, deletion, or disclosure of data
- Denial of service attacks

Looking at the above list, you might initially assume that all threats come from external sources, and that a system not on the Internet is not at risk. However, remember that poorly trained, careless, or malicious employees can represent every one of the threats mentioned.

So, how do you build and evaluate your specific threat profile? The recommended tool is a use-case-driven data flow analysis, the process of methodically tracing the flow of various use cases and their data throughout your system to identify threats and vulnerabilities. (Note that we differentiate between threats that are dependent on the specifics of a system's implementation, and vulnerabilities that are intrinsic to a system.)

## Original Entity Authentication and Bootstrapping

Original entity authentication permeates all trust models. This refers to a situation where, before trust can be established, relying entities must be convinced of the identity of all other entities with which they communicate or conduct transactions. The level of satisfaction required to convince the relying entity should be specified in a published security policy.

As the name implies, original entity authentication occurs only once, at the beginning of a trust relationship. Returning to the examples of a library and a financial institution, the library was satisfied with a rather lightweight authentication process (such as quickly checking a driver's license), whereas the financial institution required more rigorous methods. And the more rigorous the original entity authentication process is, the greater the degree of trust.

To be more precise, original entity authentication establishes a credential that can be evaluated, tested, or referenced by an authenticator or relying entity. For the library, a plastic library card may be enough. For the financial institution, there may be some kind of cryptographic key that enables the use of different encryption services.

To ensure a reliable validation or authentication process, tokens or credentials must be unique and bound to a specific entity. Furthermore, there should be an agreed upon and standardized format for credentials, as well as for the protocols used to test those credentials. Such standardization becomes an important attribute when a trust model is implemented in an application or business system.

Let's step through the process of original entity authentication.

1. Entity A requests a trust relationship with Entity B.
2. Entity B, in accordance with its stated security policy, requires Entity A to provide proof (or various proofs) of identity.
3. Entity B validates these proofs of identity.
4. Entity B returns to Entity A some unique identity credential that Entity B can test to validate Entity A in future interactions.

The last step suggests the remaining requirement: bootstrapping. This means the association of a unique entity (Entity A) with a unique credential (provided by Entity B).

As emphasized earlier, trust depends on the ability to bind unique attributes or credentials to a unique entity or user. This is precisely the bootstrapping process. Central to a trust model is the assurance that this binding is completely reliable.

To put this in context, consider the example of a financial institution. Assume that the bank uses a public key infrastructure (PKI) for its home banking application. First, the bank requires proof(s) of identity from the customer who wants to bank online. Once the bank is satisfied (according to its own policies), it issues a unique identity credential in the form of a public key certificate. That certificate provides a unique binding of the customer's identity to a set of unique cryptographic keys. These keys are subsequently used to enable the reliable implementation of various security services such as:

- *Authentication*—Verification of identity.
- *Authorization*—Granting of specific privileges.
- *Confidentiality*—Information will not be accessed by unauthorized parties.
- *Integrity*—Data will not be modified by unauthorized parties.

- *Non-repudiation*—Legitimate transactions cannot later be denied by either the customer or the bank.

There must be an agreed upon and standardized format for credentials, as well as for the protocols that are used to test those credentials. This will be an important attribute when an actual trust model is implemented in an application or business system.

## No Spontaneous Trust

As already briefly stated, spontaneous trust is a misleading concept. It implies that trust can exist without original entity authentication and bootstrapping. Without reliance on mapping to previously established credentials, there cannot be trust in an entity's identity. It is important to stress this point because too often, there is a misguided reliance on mechanisms that do not, in fact, enable a true trust relationship.

All valid trust models are predicated on the establishment of trust between any two entities. That is, original entity authentication must be performed first. Essentially, trust cannot be established spontaneously. Without performing this original entity authentication, future trust relationships cannot be created, except in a spontaneous fashion. Therefore, spontaneous trust does not, in fact, provide any confidence of an entity's identity because it does not rely on or map to any established knowledge, credentials, or characteristics. Adversaries often look for uses of spontaneous trust and try to exploit them.

Let's take the Secure Socket Layer (SSL) as an example. SSL has the ability to work in various modes, but in this example, we consider server side SSL, the most common use of SSL in the Internet today. In an SSL session between a browser and a server, the browser determines the identity of the server by testing credentials embedded in the browser by means of a PKI. This testing of credentials offers a one-way trust relationship in that the browser has some level of confidence that the server is who it claims to be. However, the server has no information for testing the browser. Essentially the server is forced to trust the browser. In effect, the server is hoping for a spontaneous trust relationship. In fact, this scenario does not provide any trust, whatsoever, for the server.

---

## Qualities of Trust Relationships

This section describes some common qualities or characteristics of trust relationships. All trust models should exhibit these qualities to be considered viable.

## Portability and Interoperability

Portability and interoperability are similar, but with subtle differences. Portability depends on standardized credential types and formats to be used anywhere, and at any time. The use of a standards-defined public key certificate recognized across multiple PKIs is an example of portability.

Interoperability depends on the standardization of protocols for testing credentials. Interoperability relies on applications and systems to implement standardized protocols for credential testing. The use of standards-defined protocols (for example, those from the Liberty Alliance) to perform security functions such as authentication and authorization across multiple platforms is an example of interoperability.

## Reliability

Reliability embraces a closely related aspect of credentials and their evaluation. Credentials and the mechanism that evaluate them must perform consistently, in a repeatable fashion over time.

## Assurance

Assurance is a critical quality of any trust relationship. Relative to trust, assurance is concerned with the preservation of the binding between a unique entity and its credentials. Here, preservation means that a credential continues to be accurately bound with the correct entity to which it is associated.

---

## Major Trust Models

This section discusses three primary trust models. Different organizations have different thresholds for risk, and the choice of a trust model should be based on that threshold. Specific security solutions should map to the applicable trust model.

### Direct Trust

Direct trust exists when you perform the validation of an entity's credentials without reliance on any other entity. There is no delegation of trust, because all relying parties are subordinate constituents of the trust hierarchy. All entities gain trust by

their association with a common entity responsible for the original entity authentication of each relying entity, always following a stated security policy. Distinct binary trust relationships are established between a common trust point and the various end entities.

A direct trust model is found in some architectures using a PKI. In this example, the root certificate authority (CA) initiates all trust relationships. The CA is the common trust entity that performs all original entity authentications and the generation of credentials that are bound to specific entities. A key difference with other models is that the direct trust model does not allow the delegation of original entity authentication. And every relying party must use this CA directly for all validation processes.

The advantage of the direct trust model is that the validation of credentials is performed by one's self with no delegation, thus ensuring a high level of confidence in every entity associated with the trust implementation. Direct trust is often necessary to reduce liability for organizations bound by regulatory or fiduciary requirements. Organizations involved with financial transactions, e-commerce, insurance, or health care should consider a direct trust model.

However, as we stated earlier, trust is not established without effort. The primary disadvantage of the direct trust model is that it may be more labor intensive and more expensive than other trust models.

## Transitive Trust

Transitive trust is trust transmitted through another party. Transitive trust allows the following:

- Entity A validates and trusts Entity B.
- Entity B validates and trusts Entity C.
- Entity A trusts but does not need to validate Entity C. For example, Entity A trusts Entity C, but does not perform original entity authentication of Entity C.

Such a trust model is common in distributive or peer-to-peer systems. It relies on participating entities to align their security policies that control credential validation (for example, original entity authentication). In the preceding example, for A to trust C, A needs confidence that B has validated C by the same standards that A used to validate B. Because you are explicitly trusting another entity to perform credential validation, you should, at the very least, evaluate that entity's security policy, validation process, and position on liability management.

The advantage of the transitive trust model is that it enables the linkage of different entities that share similar security policies while reducing the credential validation effort.

Consider the following example of transitive trust, which is common with the frequency of bank mergers in recent years. You are a customer of Bank ABC, which is acquired by Bank XYZ. Because XYZ trusts ABC's original validation process, you are trusted by XYZ to continue your normal banking activities (unless previously allowed activity somehow significantly violates the new owner's security policy). The new portfolio owner has presumably reviewed, very carefully, ABC's financial statements, security policy, and validation process to have confidence in extending trust to the customers it gains through the acquisition.

## Assumptive Trust

Assumptive trust is a formal name for a model that was earlier described as spontaneous trust. With this model, there is no mandatory, explicit, direct credential validation. With essentially no control over the validation process, you must either “take it, or leave it.”

An example of an assumptive trust model is the pretty good privacy (PGP) web of trust. The validation of entities is essentially one personally vouching for another. Although this web of trust has some value for relatively simple activities, such as signing email messages, it is not sufficient in the business realm. Many users have false confidence that PGP is based on a transitive trust model, so it is important to emphasize that which differentiates a transitive trust model from an assumptive one is the validation process (or lack thereof). A transitive trust model requires a validation process, and an assumptive one does not.

There are many other examples of assumptive trust models. If more than casual, noncritical information or processes are involved, you should consider implementing these and other protocols with at least a transitive trust model.

---

## Developing Your Own Trust Model

Just as we identified original entity authentication as the first step in establishing trust, we need to identify the starting point for defining a trust model. We have already talked about matching a model to business requirements, but at this point, we will go back an additional step. At the very foundation of trust model development, there should be a basic principle governing acceptable use of both data and processing resources. Sometimes this is referred to as the security stance. A common security principle used by many organizations in the Internet space is that data must be accessible only on a need-to-know basis, and processing resources must be accessible only to those explicitly approved to use them.

This principle would be appropriate for an organization with stringent security requirements, but another organization concerned with attracting customer interest would have a different philosophy about access to at least certain types of data.

With the underlying principle understood, the next step is to gather business requirements. These requirements must be specific to the organization, although they may well be similar to those of other comparable organizations. In many cases, requirements will be determined by legal and regulatory mandates. Perhaps a Service Level Agreement (SLA) will be a determining factor. If there is a requirement for a certain level of performance (such as speed, throughput, or availability), it must be identified, because a security mechanism with a noticeable impact on performance might be unacceptable.

Next, build a threat profile based on a data flow analysis. An example of a data flow analysis is provided below, but the key point here is to catalog threats and vulnerabilities. Some risks are going to be more significant than others and will require greater effort to mitigate. Some may even be judged relatively inconsequential and therefore not cost-effective to address.

In assessing risk, be sure not to focus solely on technical issues. It cannot be stated often enough that good security requires a combination of people, process, and technology. Look for those places where human error or poorly designed processes introduce risk.

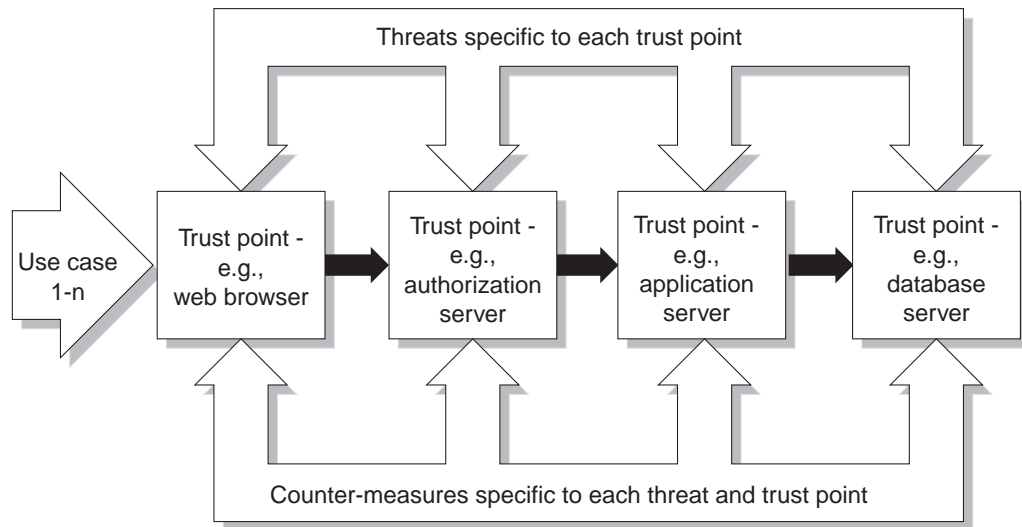
Finally, identify security mechanisms to respond to those risks that must be addressed. Response mechanisms may be technical in nature (for example, a firewall), but they may also address the human factor (for example, better training). At this point, a technical solution will likely be identified at a relatively high level (for example, to install a firewall). The choice of the specific firewall to install and the configuration details generally come later in an implementation phase.

## Example of a Data Flow Analysis

To illustrate what data flow analysis offers, consider the example of online banking. For the purposes of illustration, the following steps assume the customer wants to view his current account balance, and the bank has already established customer authentication credentials.

1. The customer initiates a session with bank's server. The protocol governing the session is SSL.
2. The SSL server dialog is established, which encompasses all of the following steps. (Note that client-side SSL authentication is not performed.)
3. The customer enters the required information (for example, name, account number, and PIN) on the login screen.

4. The authentication server validates credentials.
5. The Web services server presents a welcome and a menu of options the customer can invoke.
6. The customer's selection of the account query option is passed to the application server.
7. The application server queries the authorization server to see if the selected option is allowed.
8. The authentication server returns a response to the application server, based on rules. For this example, it allows the query to be made.
9. The application server requests the database server to obtain account records.
10. The database server returns records to the application server.
11. The application server formats data and passes it to the Web services server for presentation to the customer.
12. Note that the SSL server has encrypted all session data to and from the customer.



**FIGURE 1** Data Flow Analysis

- In this example, two types of data were processed:
- Customer authentication information (name, account number, PIN)
  - Customer account information (account balance)

A complete analysis requires examination of the type of data processed at each trust point, identification of the types of threats affecting each trust point, and specification of the appropriate countermeasure.

For the purposes of this example, start by identifying all the different servers involved:

- SSL server
- Authentication server
- Web services server
- Application server
- Authorization server
- Database server

For a seemingly simple request, the sheer number of servers dramatizes the number of communication paths, and therefore trust points, that data flows through. Each of the trust points is also a potential attack point. A security failure of any of these points could have very damaging consequences.

What is the risk of exploitation of a security failure? Presumably, it is very high, because sensitive financial data is at stake. Compromises could lead to disastrous results, such as:

- Someone other than the account owner gaining access to data
- Legitimate account owner being presented with more information than should be transmitted
- Account owner (or someone else) being able to perform unauthorized transactions

Our hypothetical financial institution must have as minimum business requirements (no doubt reinforced by legal and regulatory requirements) the need to protect data confidentiality and integrity.

## Trust Model Derived From the Example

Upon completion of a much more thorough analysis of threats, based on data flow analysis, risks, and business requirements, you have the information necessary to determine the appropriate trust model. In the example, to satisfy the requirements for confidentiality, integrity, and non-repudiation, there must be very stringent authentication and authorization mechanisms. Because sensitive data is being transmitted over the Internet, encryption should be required. The trust required is between the customer and the bank. No trust among the various customers using the online banking application is required or even allowed. Based on these facts, a direct trust model should be employed. Because the type of information a customer is allowed to access must be strictly controlled, the trust model incorporates the basic principle: data will be accessible only on a need-to-know basis.

Expressed more formally, our trust model might read something like the following:

- The security architecture must implement a direct trust model.
- All users must be identified and authenticated. Trust and authentication can never be implied or assumed.
- No transitive or assumptive trust can exist between any component of the bank's computing environment and any external system.
- Entities that are uniquely identified and authenticated may be trusted to access data and resources only on a predefined need-to-know basis.
- Data transferred over the Internet between the bank and an authenticated user must be encrypted.

---

## Conclusions

Trust modeling is not an abstract intellectual exercise. It is not something interesting to do if you think time permits but dispensable if you don't want to spend time on it. Trust modeling is an essential step in designing a secure architecture.

A trust model must be constructed to match specific business requirements. No generic trust model can be assumed to be valid for a specific situation.

Given the urgency of having a trust model and the need to construct it to match specific business requirements, it is important to assign the necessary resources to develop a model based on a threat profile and risk analysis and to identify the appropriate response mechanisms. Establishment of trust does not happen spontaneously or without effort.

Do not focus solely on technical solutions. As with all aspects of a security architecture, a successful trust model must consider people, process, and technology.

Finally, if you remember nothing else from this article, do not forget the following:

- Failure to understand what trust model (if any) is actually in effect can create a false sense of security that may lead to serious, even catastrophic, financial and legal problems.
- Adversaries exploit weak trust models.

---

## About the Authors

### Donna Andert

Donna has been in the computer and network security development field for over 10 years. As a Security Architect with the GESS security team, she designs security solutions, conducts security assessments, and performs analysis of implementations and operations. Prior to joining Sun, Donna was a UNIX® kernel developer. Projects included a proprietary authentication and firewall solution for the first cable Internet deployments in the United States and a fault tolerant distributed UNIX file system for the financial and pharmaceutical industries. Her research areas of interest include host intrusion detection, forensics of file systems and volatile memory, and secure kernel design.

### Robin Wakefield

Robin has worked professionally with computer technology for 20 years. Current subject matter expertise includes Solaris™ Operating Environment security, network security, Sun Fire™ and Starfire™ system platforms. Topographical expertise is focused on Internet architectures and security. His knowledge base includes Sun's product lines and product direction, Hewlett Packard UNIX systems, IBM RS-6000 AIX, Linux, Windows NT systems, and Cisco Networking products.

### Joel Weise

Joel has worked in the field of data security for over 20 years. As a Senior Security Architect for Sun Professional Services, he designs system and application security solutions for a range of different enterprises from financial institutions to government agencies. He specializes in cryptography and public key infrastructures. Prior to joining Sun, Joel was a Senior Project Manager for Visa International, where he was responsible for developing cryptographic standards, designing key management and cryptographic systems and architecting security solutions for chipcard, Internet, and other new products.

---

## Acknowledgements

We want to thank William “Bill” Robinson for volunteering to edit this work with us.

---

## References

ITU-T Recommendation X.509, ISO/IEC 9594-8: “Information Technology - Open Systems Interconnection - The Directory: Public-Key and Attribute Certificate Frameworks.”

---

## Ordering Sun Documents

The SunDocs<sup>SM</sup> program provides more than 250 manuals from Sun Microsystems, Inc. If you live in the United States, Canada, Europe, or Japan, you can purchase documentation sets or individual manuals through this program.

---

## Accessing Sun Documentation Online

The `docs.sun.com` web site enables you to access Sun technical documentation online. You can browse the `docs.sun.com` archive or search for a specific book title or subject. The URL is `http://docs.sun.com/`

To reference Sun BluePrints OnLine articles, visit the Sun BluePrints OnLine Web site at: `http://www.sun.com/blueprints/online.html`