

NATIONAL SECURITY AGENCY
CENTRAL SECURITY SERVICE



R22

INFOSEC Engineering

TOKENEER



**User Authentication Techniques
Using Public Key Certificates
Part 3: An Example Implementation**

10 February 1998

Lawrence A. Reinert
Stephen C. Luther

User Authentication Techniques using Public Key Certificates
Part III: An Example Implementation

TABLE OF CONTENTS

1.0 Scope	1
2.0 Background	1
2.1 Trust	1
2.2 TOKENEER Architecture	2
2.2.1 Enrollment Station	2
2.2.2 Token (Smartcard)	3
2.2.3 ID Station	3
2.2.4 Workstation	3
2.2.5 Certificate Authority (CA)	3
2.2.6 Attribute Authority (AA)	4
2.3 TOKENEER Authentication Protocol	4
2.4 Terminology	5
2.5 Symbols and Abbreviations	6
3.0 Reference Documents	7
4.0 An Example using X.509 Authentication Information	8
4.1 Determining the Authentication Information Requirements	8
4.2 Determining the Authentication Information Placement	9
4.2.1 Assigning the System weights to the desired attribute characteristics	9
4.2.2 Using the Comparison chart	11
4.2.3 Defining the Certificates Needed by the System	11
4.3 System Scenario	13
4.3.1 Enrollment	13
4.3.1.1 System Enrollment	14
4.3.1.2 Local Enrollment	15
4.3.2 Workstation Access	16
4.3.2.1 Domain Verification	16
4.3.2.2 Workstation Verification	17
5.0 Conclusions	19
APPENDIX I - TOKENEER Privilege Certificate ASN.1 Syntax	20
APPENDIX II - TOKENEER ID Authentication Certificate ASN.1 Syntax	23
APPENDIX III - TOKENEER Timing Analysis	24
APPENDIX IV - TOKENEER Attribute Certificate Sizing	25

1.0 Scope

The intent of the three part study is to investigate placement of user authentication information within public key certificates. Part 1 [16] examines the options within X.509 (and related standards) for the placement of user authentication information and how system requirements affect the decision process. Part 2 [17] investigates specific authentication information, such as biometrics and passwords, which should be considered for placement in a public key certificate during the development of a system. This document (Part 3) presents a case study of how to architect such a system through an example implementation.

This part of the study illustrates the techniques described in Part 1. It also provides an example of how to place the authentication information described in Part 2. It is not the intent of this study to dictate a standard or suggest a rigid implementation of authentication information.

2.0 Background

The TOKENEER system will be used to illustrate the concepts discussed in this study. TOKENEER is a proof-of-concept system being used to prototype new Identification and Authentication (I&A) concepts. The following sections give a brief overview of the system and some underlying concepts of the system. Refer to the TOKENEER Operational Concept Description, the TOKENEER System Subsystem Specification, and the TOKENEER System Design Document for further details.

2.1 Trust

The following definitions are found in the "Glossary of Computer Security Terms", reference [15].

Trusted Computing Base (TCB): The totality of protection mechanisms within a computer system, including hardware, firmware, and software, the combination of which is responsible for enforcing a security policy. A TCB consists of one or more components that together enforce a unified security policy over a product or system. The ability of a TCB to enforce correctly a unified security policy depends solely on the mechanisms within the TCB and on the correct input by system administrative personnel of parameters (e.g., a user's clearance level) related to the security policy.

Trusted Path: A mechanism by which a person at a terminal can communicate directly with the TCB. This mechanism can only be activated by the person or the TCB and cannot be imitated by untrusted software.

Trusted Process: A process whose incorrect or malicious execution is capable of violating

system security policy.

“Trust” will be used in this document to denote that an entity, to which the term is being applied, can be expected to, within normal operating procedures, perform its given tasks without a significant risk of manipulation by a malicious source. This implies that the operations which it performs are well known at the time of its design and that its design has passed an evaluation by an accredited security analysis.

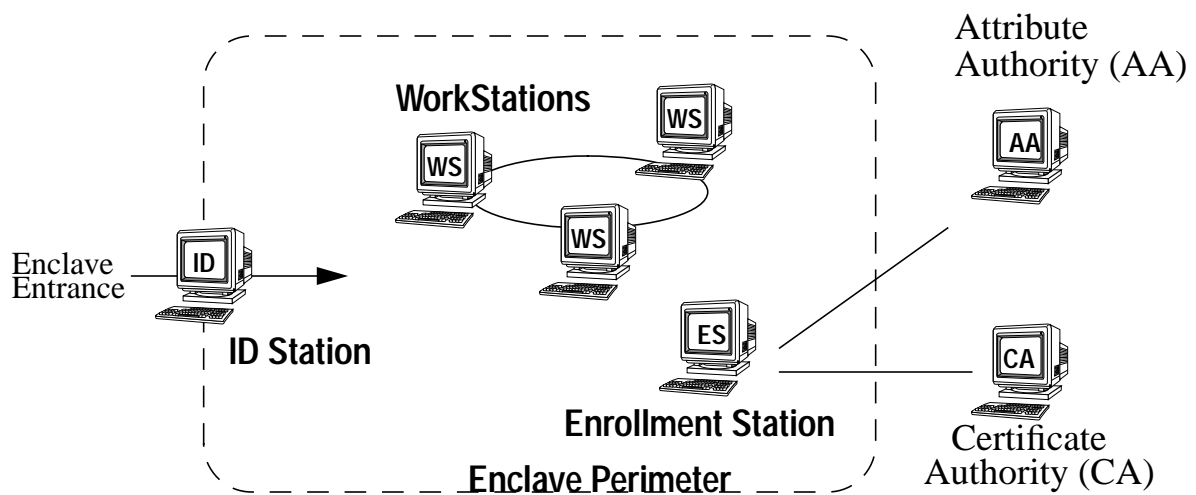
A entity can be trusted as a whole, or sub-portions of the entity can be trusted. Whole trusted components must be special purpose entities since their operations must be evaluated prior to use. If only portions of the entity are trusted, then only operations which are performed on those trusted components can be trusted.

2.2 TOKENEER Architecture

The TOKENEER system uses a nested approach to authentication based upon the premise that the trust can be distributed throughout a system to allow for untrusted components to operate. It also utilizes a trusted token to provide authentication information to service access points.

The TOKENEER architecture consists of 6 components as illustrated in the following diagram

TOKENEER Architecture



2.2.1 Enrollment Station

The enrollment station is a “trusted” entity which is responsible for registering the user into the system. The user must be approved via another mechanism (i.e. clearances) prior to being enrolled on the system. The enrollment station will gather the user’s identification information,

the biometric information, and other system required information (i.e privileges, clearance levels, etc.), the information is sent to the appropriate authority for further processing. The enrollment station retrieves the processed information from the authorities and places certificates into the token.

2.2.2 Token (Smartcard)

The smart card is a “trusted” entity which is used as a token to provide the “what you have” portion of I&A. Its basic functions will include holding the biometric template for authentication, performing signature functions for token verification, and providing a public key mechanism for data confidentiality.

2.2.3 ID Station

The ID station is a “trusted” entity responsible for performing biometric verification of the user. To perform this function, it must extract the signed biometric template from the token, verify the signature, read the fingerprint scan from the user, and compare the scanned image with that of the verified biometric template. The ID station uses the “what you have” and “what you are” portions of I&A. The “what you know” portion is obtained by the workstation. Since the functions performed on the ID station are of a critical nature to the system, connectivity with the rest of the system is not allowed. Although ID stations may be nested or placed in parallel with each other, to meet the strictest security requirements, they should not to be interconnected. Revocation lists would require manual distribution with these isolated systems. Thus, the TOKENEER system cannot use a trusted third party architecture.

2.2.4 Workstation

The workstation is an “untrusted” entity which can be any service access point which provides authentication beyond that of the ID station by requiring that the user provide a password to the token. The workstation does contain a “trusted” sub component called a CryptoCard. The CryptoCard is utilized for encrypting all data on the workstation to protect against malicious access. The workstation is responsible for performing checks on authentication data issued to the token by an ID station. Services offered by the workstation may include: system logon, document signing, document source verification, or encryption/decryption.

2.2.5 Certificate Authority (CA)

The Certificate Authority is a “trusted” entity which is a commercially available component which creates, distributes, and maintains public key certificates for the TOKENEER system. The enrollment station passes the raw (unformatted) public key to the CA in the form of a signed certificate request. If the CA accepts the certificate request, it formats and signs the certificate in accordance with the X.509 specification.

2.2.6 Attribute Authority (AA)

The Attribute Authority is a “trusted” entity which is responsible for creating attribute certificates as defined by the TOKENEER system. The enrollment station sends the attribute information (biometric, clearance, and privilege information) to the AA. Once the information is validated, the AA formats and signs the attribute certificates in accordance with X.509. The attribute certificates are returned to the enrollment station for distribution to the appropriate token.

2.3 TOKENEER Authentication Protocol

This protocol is a combination of FIPS 196, RSA public key encryption, and the Lock/Unlock protocol to provide the services of mutual authentication, a secure encrypted channel between the host and the token, and secure key storage on the token. For more detailed information, see reference [7].

2.4 Terminology

Several terms used throughout this document must be clarified before proceeding:

Authenticate: 1. Establish or prove as: a) conforming to a fact and therefore worthy of trust, reliance, or belief. b) having an undisputed origin. 2. To establish the validity of a claimed identity.

Biometric: A measurable, unique physical characteristic or personal trait used to recognize the identity, or verify the claimed identity, of an enrollee.

Certify: To confirm formally as true, accurate, or genuine.

Certificate: A document testifying to accuracy or truth.

False Acceptance: When a biometric transaction results in the acceptance of an imposter (also known as a false match or type II error).

Identify: 1. To establish that the collective aspects of the characteristics by which a thing is distinctly recognizable or known. 2. To consider similar or identical: equate.

Privilege: A special grant, immunity, right or benefit granted to an individual, class or caste.

Template: A data set representing the biometric measurement of an enrollee which is maintained on file and used by a biometric verification device for comparison against subsequently submitted biometric samples.

Token: A possession which shows the identity of its owner, such as an identity badge.

Trust: A term used to denote that the entity to which the term is being applied can be expected to, within normal operating procedures, perform its given tasks without a significant risk of manipulation by a malicious source. This does imply that the operations which the entity performs are well known at the time of its design and that its design has passed an accredited security analysis.

Verify: To prove the truth of by presenting evidence or testimony: substantiate.

Verification: The process of comparing a submitted biometric sample against the template of the single enrollee whose identity is being claimed in order to determine whether it matches the enrollee's template or not.

Verification Attempt: The submission of a biometric sample, accompanied by a claimed enrollee identity, to a biometric verification device for a verification decision.

2.5 Symbols and Abbreviations

This section contains symbols and abbreviations used in this document.

Table 1: Symbols and Abbreviations

Abbreviation	Meaning
AA	Attribute Authority
ASN.1	Abstract Syntax Notation
AI	Authentication Information
CA	Certification Authority
CRL	Certification Revocation List
DN	Distinguished Name
ECMA	European Computer Manufacturers Association
IDACert	Identification and Authentication Certificate
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
I&A	Identification and Authentication
I/O	Input / Output
PrivCert	Privilege Certificate
RSA	Rivest Shamir Adelman algorithm
VAI	Validated Authentication Information
WD	Working Draft

3.0 Reference Documents

- [1] ECMA. ECMA-219, "Authentication and Privilege Attribute Security Application with related key destruction functions." second edition, 1996.
- [2] ISO/IEC. ISO/IEC 8825, "Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)." first edition, 1995-10-15.
- [3] ISO/IEC. ISO/IEC 9594-2, "Information technology - Open Systems Interconnection - The Directory: Models." 1995.
- [4] ISO/IEC. ISO/IEC 9594-6, "Information technology - Open Systems Interconnection - The Directory: Selected attribute types." 1995-09-15.
- [5] ISO/IEC. ISO/IEC 9594-6, "Information technology - Open Systems Interconnection - The Directory: Selected attribute types AMENDMENT 2:Certificate extensions." draft edition, 1996-03-09.
- [6] ISO/IEC. ISO/IEC 9594-8, "Information technology - Open Systems Interconnection - The Directory: Authentication framework." 06/97.
- [7] Luther, S. and Reinert, L. "TOKENNEER Authentication Protocol for Smartcards," version 1.0, January 23, 1998.
- [8] NIST. FIPS Pub 190, "Guideline for the use of Advanced Authentication Technology Alternatives." September 28, 1994.
- [9] NIST. FIPS Pub 196, "Entity Authentication Using Public Key Cryptography." February 18, 1997.
- [10] NSA R223. "TOKENNEER Operational Concept Description." Build 1: version 1.0, November 25, 1996.
- [11] NSA R223. "TOKENNEER System/Subsystem Specification (Requirements Document)." Build 1: version 1.0, November 24, 1997.
- [12] NSA R223. "TOKENNEER ID Station Software Design Description." Build 1: version 1.0, July 30, 1997.
- [13] NSA R223. "TOKENNEER Enrollment Station Software Design Description." Build 1: version 1.0, August 28, 1997.
- [14] NSA X3. "An Analysis of Type II Certificate Sizes with Version 3 Certificate Extensions." prepared by Booz Allen & Hamilton Inc., July 30, 1996.
- [15] NCSC. NCSC-TG004-88, "Glossary of Computer Security Terms." October 21, 1988.
- [16] Reinert, L. and S. Luther. "User Authentication Techniques Using Public Key Certificates Part 1: Certificate Options." December 24, 1997
- [17] Reinert, L. and S. Luther. "User Authentication Techniques Using Public Key Certificates Part 2: Authentication Information Including Biometrics." December 31, 1997.

4.0 An Example using X.509 Authentication Information

The following example implementation utilizes the authentication information described in this document. It is only one of many possible implementations. The purpose of the example is to illustrate the design process used when designing a system.

4.1 Determining the Authentication Information Requirements

As discussed in Part 1 of this study, there are several factors which go into the placement of the authentication information (AI). These factors and TOKENEER's requirements are listed below:

1. What authentication information is required by the system?
 - Fingerprint biometrics
 - Clearance level
 - Role certification
 - A password (optional)
2. What information should be kept private and what data can be made public. Any information in the X.509 is public?
 - Public: user name
 - Private: biometric information, clearance, and role certification
3. Will the information to be placed in the certificate be stable during the validity period of the certificate?
 - The certificate will be valid for at least a year. All authentication information will be stable.
4. Is certificate storage a problem?
 - The token being used will have a minimum of 2.5K bytes storage for the certificates.
5. Is system throughput a factor?:
 - Since the token I/O is the only (slow) I/O, it is a factor that will be applied to the overall processing time (refer to question 6)
6. Is processing time a factor?
 - Yes, the users of the system will not tolerate more than 2 seconds for total authentication time.
7. Where will the information be used in the system?
 - For this example, clearance, and privilege will be used by all system components. Biometric information will only be used by the ID Station or the Enrollment Station. The user's name,

clearance, privileges, and biometric data will be considered private. The workstation will only require the clearance and privileges, not the biometric data. The information will be placed on a token and stored on the Enrollment Station for archiving purposes.

4.2 Determining the Authentication Information Placement

The next step determines which certificate option, discussed in Part 1, would be best for the system by using the comparison chart discussed in the conclusion of Part 1. The chart requires a system specific weight be assigned to each requirement. The following is a discussion of each of those requirements as it pertains to the TOKENEER system. The weight assigned to each is based upon the discussion.

4.2.1 Assigning the System weights to the desired attribute characteristics

Attribute Confidentiality: Maintaining confidentiality of the user's private attributes is a vital system concern. The user attributes should be encrypted by the token and given only to trusted system components.

System weight assigned = 10

Need To Know: Maintaining access control on a need to know basis is a vital system requirement. The workstation should only be given minimal knowledge of the user. The biometric information is not needed by the workstation, since it is utilized only by the ID Station. Clearance information should be filtered by the ID Station to show the workstation the minimum information needed. To accomplish this, the ID Station will create a timestamp which contains role, privilege, and validity period information that will be placed on the token for use by the workstation's login function.

System weight assigned = 8.

Certificate Storage: The certificate(s) will be stored on a smartcard. The maximum storage space cannot exceed the storage available on the chosen smartcard. The minimum amount of storage for the smartcard will be 2.5K bytes. Since most biometric templates are 500 bytes or less, It is assumed that the storage requirement can be met. Refer to "APPENDIX IV - TOKENEER Attribute Certificate Sizing" on page 25 for more detailed certificate sizing information.

System weight assigned = 3.

I/O throughput. Most of the information in the TOKENEER system is processed locally, therefore transfer rates across a network are not a factor. The relatively slow serial interface between the smartcard and the host is a concern, but as long as the total processing time does not exceed the allocated 2 seconds, it is not a major concern. The total processing time requirement may only be altered or waived if a specific overriding security requirement (such as attribute confidentiality) cannot be met within the performance requirement. Even so, a minimal level of performance must be met for a system to obtain any level of user acceptance (see “APPENDIX III - TOKENEER Timing Analysis” on page 24). Until the complete protocols are developed and their timing characterized, this category will remain a top priority.

System weight assigned = 10 (for both single certificate and all certificates).

Certificate processing time. An overall timing estimate should be made when the complete set of requirements is known for the data processing and the smartcard has been chosen. The overall processing time of 2 seconds (at either the ID station, or workstation) should not be exceeded, but allowances can be made if security is enhanced. Assuming that the technology exists to meet the 2 second requirement, this requirement is given a medium priority. Refer to the preliminary storage/timing estimate following this section.

System weight assigned = 5 (for both single certificate and all certificates).

Architectural Complexity: Adding complexity to the system is acceptable if security is enhanced by a proportional amount. Another component (i.e. an attribute authority) can be added to the system if a higher weighted requirement (i.e. attribute confidentiality) is met. Since it is assumed that added security can add complexity to the system, this requirement is given a relatively low priority.

System weight assigned = 3.

Management Complexity: Management complexity can be greater as long as it can be automated. It is assumed that greater security will add complexity and some overhead as long as it is proportional to the security provided. Any attribute certificates will be linked to the serial number of the user’s public key certificate and therefore to only one set of certificate revocation lists (CRLs) for the system. It is assumed that currently available certificate authorities can manage the public key management functions in an automated way, and that relatively little complexity will be added to the system due to adding attribute certificates. Therefore this requirement is given little priority.

System weight assigned = 3.

4.2.2 Using the Comparison chart

Using a numeric equivalent for the ranges in the comparison chart we can make a score of each method explained in Part 1 of this study. The numeric equivalents used are: Good = 10, Average=5, Poor-Average=3, Poor = 1. Scores are calculated by multiplying the each cell by the Sys-

Table 2: Comparison Chart

System Characteristic	X.509 Subject Directory Extension	PKCS#6 X.509 Extended Certificate	X.509 Attribute Certificates	System Weight
Attribute Confidentiality	1	1	10	10
Need To Know	1	5	10	8
Small Certificate Storage (assuming all certificates need to be stored)	10	5	1	3
Low I/O throughput (all certificates sent)	10	5	1	10
Low I/O throughput (one certificate sent)	1	3	10	10
Small Processing Time (all certificates processed)	10	5	1	5
Small Processing Time (one certificate processed)	1	3	10	5
Architecture Complexity	10	5	1	3
Management Complexity	10	5	1	3
Score	273	215	354	

tem weight in the row, and adding up all the cells in the column.

The scoring results on this chart suggest that the X.509 Attribute certificate has some overall advantages.

4.2.3 Defining the Certificates Needed by the System

Using the results from the previous section, it was determined that three certificates will be used: the X.509 public key certificate, an attribute certificate containing privilege and clearance

information (the PrivCert); and a second attribute certificate containing the identification authentication information (the IDACert). The complete ASN description is located in APPENDIX I (PrivCert) and APPENDIX II (IDACert). The information contained in each certificate is illustrated in the following figure:

Information in the TOKENEER X.509 and attribute certificates

X.509 Public Key Cert	PrivCert	IDACert
version serialNumber signature issuer (CA) validity subject subjectPublicKeyInfo algorithmIdentifier signatureValue (CA)	version subject (baseCertificateID) issuer (AA) signature serialNumber validity clearance role issuerUniqueID (Token #) algorithmIdentifier signatureValue (AA)	version subject (baseCertificateID) issuer (AA) signature serialNumber validity biometricInfo issuerUniqueID (Token #) algorithmIdentifier signatureValue (AA)

Bold items are attributes which were added as attributes to the certificate

The X.509 Public Key certificate only contains public key information for the following reasons:

1. All information is assessable by the public, therefore minimum information is kept in the X.509 public key certificate. The role, clearance, and biometric data are confidential, and should not be made public for this system.
2. The mutual verification protocol (i.e. challenge/response) can be performed using public information, revealing no information if one party is an imposter.
3. The time it takes to perform a mutual verification protocol is minimized since the Public Key certificate size is reduced (the signature verification time and I/O time should be minimized).

Privilege information is contained in its own certificate (PrivCert) for the following reasons:

1. Role and clearance information must be kept confidential for this system, therefore it cannot be part of the X.509 public key certificate.
2. Biometric data does not need to go to the workstation. Having role and clearance information in a separate certificate will allow this information to be given to a workstation without giving the biometric data to it. (i.e. the “need to know” principle can be enforced)
3. Information can be delivered to another entity after a mutual authentication protocol has been successfully performed (i.e. the authenticity of the recipient has been confirmed)
4. The PrivCert can be encrypted at the transport layer, protecting the confidentiality of the information as it is given to another entity.

Authentication information is contained in its own certificate (IDACert) for the following reasons:

1. Biometric data information must be kept confidential for this system, therefore it cannot be part of the X.509 public key certificate.
2. Biometric data does not need to go to the workstation. Having role and clearance information in a separate certificate will allow this information to be given to a workstation without giving the biometric data to it. (i.e. the “need to know” principle can be enforced)
3. Information can be delivered to another entity after a mutual authentication protocol has been successfully performed (i.e. the authenticity of the recipient has been confirmed)
4. The Privilege cert can be encrypted as it is shared with another entity, providing confidentiality for the sensitive information.

The authorities which create the PrivCert and the IDACert do not have to be the same. In fact, it might be good to separate the roles for security reasons.

This structure is similar to the certificate defined by ECMA.219. The main difference is the ECMA.219 authentication and privilege certificates are meant to be temporary certificates. The certificates described in this document are meant to be valid for a much longer duration. Since these certificates will be stored on a token, the time between certificate upgrades will typically be on the order of a year.

4.3 System Scenario

The following scenarios have been created to describe the TOKENEER system: enrollment and workstation access. Each scenario is stated in terms of certificate creation and utilization, so that the concepts described in this study can be illustrated.

4.3.1 Enrollment

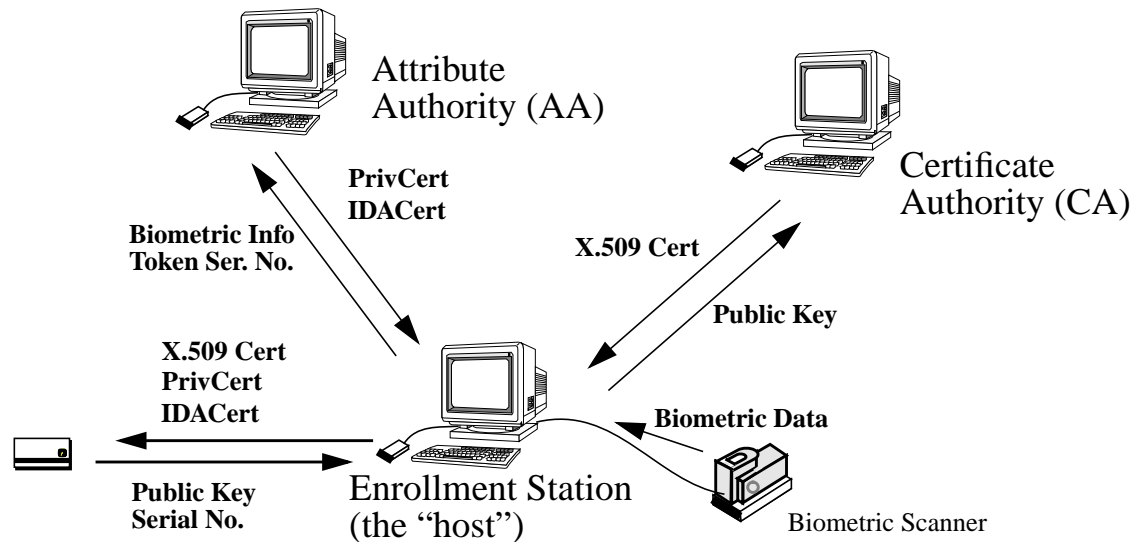
The enrollment process has been separated into two levels of enrollment: system enrollment and local enrollment. The reasons for separating enrollment are as follows:

1. A certification process can be maintained at the system level.
2. A local certification process can be added as needed.
3. Local control can be given to system administrators further enforcing the “need to know” requirement.
4. A two level enrollment process is much more difficult for an adversary to subvert than a single level enrollment. This is especially true when the enrollments are handled in two different facilities by two different sets of administrators.
5. Local enrollment can be a currently existing enrollment process (i.e. an OS enrollment) with a few modifications.
6. Revocation of enrollment can be handled at the local and system level.

4.3.1.1 System Enrollment

System enrollment is the process by which one is certified for using the system. The Enrollment Station is responsible for collecting the enrollment information and delegating the tasks that need to be performed. The system enrollment process is illustrated in the following figure.

The System Enrollment Process



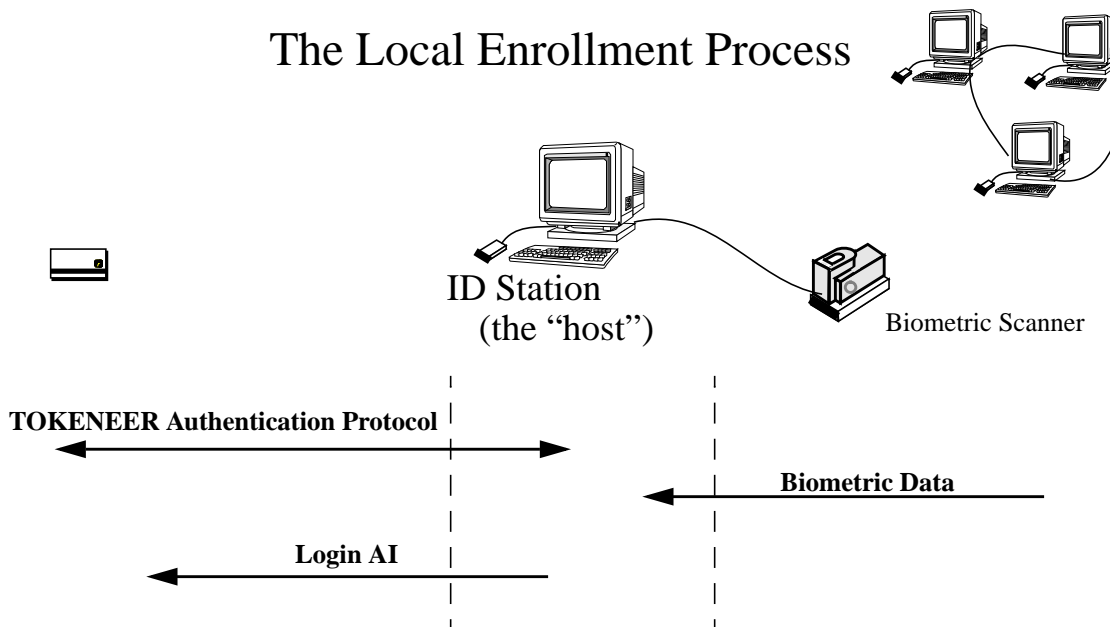
The user must be approved via another mechanism (i.e. a background check) prior to being enrolled on the system. This is typically a manual operation and is not covered by this description.

The steps to achieve system enrollment are:

1. The token generates a public/private key pair.
2. The Enrollment Station retrieves the public key from the token and sends it to the CA for X.509 formatting. The formatted key is returned to the Enrollment Station and placed in the token for storage.
3. User information including first name, middle initial, last name, organization, organizational unit, role (SSO, admin, auditor, or user), and clearances are entered into the Enrollment Station.
4. The host reads in the serial number of the token.
5. A biometric livescan is taken of the user to be enrolled. The livescan is processed into a biometric template.
6. The biometric template, the token's serial number, the public key certificate's serial number, and the user's authentication information are sent to the AA. The AA creates the X.509 formatted attribute certificates (The Privilege certificate and the ID Authentication certificate). The attribute certificates are returned to the Enrollment Station and placed in the token.
7. The user enters in a PIN or password to the token.

4.3.1.2 Local Enrollment

Local enrollment occurs when the a system enrolled user needs access to a local domain. This usually involve the creating of a system account using tools provided by the local operating system (OS). The operating system must be upgraded to provide the local enrollment access control techniques described in the following sections. The local enrollment process is illustrated in the following diagram:



The steps to achieve local enrollment are as follows:

1. The token and the local host mutually authenticate each other using the TOKENEER Authentication Protocol (reference [7]). As part of the protocol, the token sends the PrivCert and IDACert to the host over the established encrypted channel.
2. The host validates the **signatureValue** of the PrivCert and IDACert.
3. The **clearance** information of the PrivCert is compared to the classification of the local system. The **clearance** on the certificate must be equal to or higher than those of the local system.
4. The serial number of the token is compared to the **issuerUniqueID** attribute found in the PrivCert and IDACert.
5. The **serialNumber** of the public key certificate is compared to the **owner (baseCertificateID)** attribute found in the PrivCert and IDACert.
6. A livescan biometric sample is taken of the user and processed into a biometric template. The template is compared to a compatible biometric template found in the IDACert.
7. If each step in this sequence occurs, then the user is given an account to the system - using the OS local registration process (such as the Windows NT user manager).

- 8. The system may optionally provide an additional authentication attribute (such as a password) to the token which can be used during the logon process.

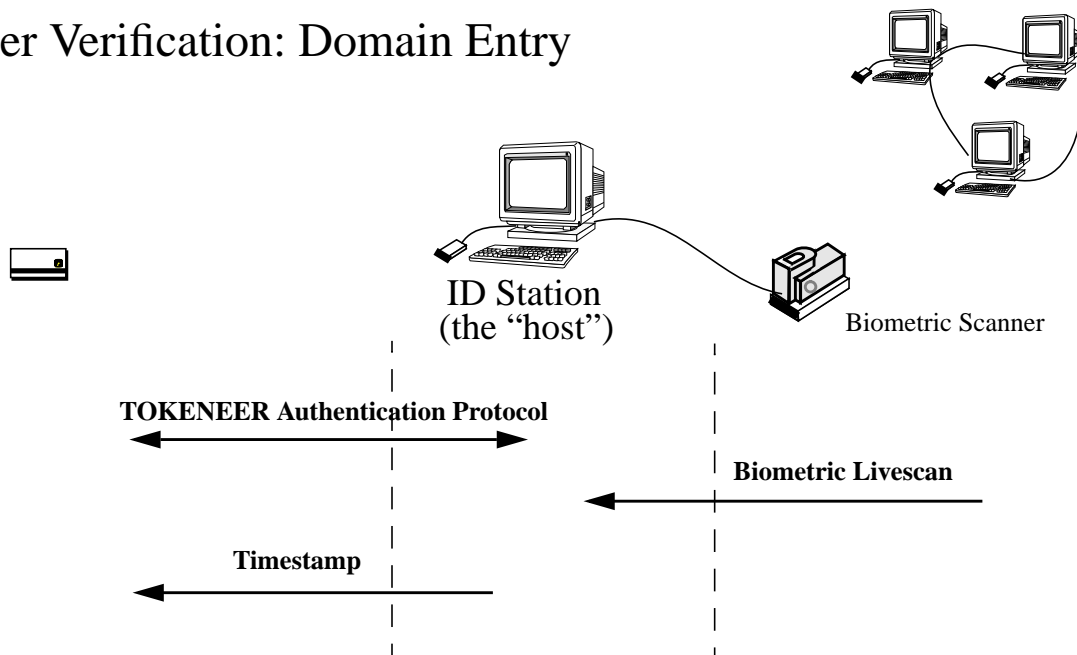
4.3.2 Workstation Access

Since a “trusted” workstation is a distant reality, the TOKENEER system distributes the trust to several components in order to limit the vulnerabilities of the untrusted workstation. The IDstation is a trusted entity which performs the biometric and token verification. If the verification is successful, a timestamped certificate (referred to as the timestamp) is issued to the token allowing access to the system for a specific amount of time. When the user logs into a workstation, this information is used to verify that the biometric verification has taken place.

4.3.2.1 Domain Verification

The ID Station is responsible for performing user verification. The ID Station is intended to be placed at an enclave entrance (such as a door) where all will have easy access. It can (optionally) be used to control a locking mechanism providing some physical security. The ID Station verifies the validity of the token by performing the TOKENEER Authentication Protocol (reference [7]). The authenticity of the biometric data contained in the IDA Certificate is accomplished by further attribute certificate validation. The user is verified by comparison of a certified template to a livescan. The following diagram illustrates the domain verification process:

User Verification: Domain Entry



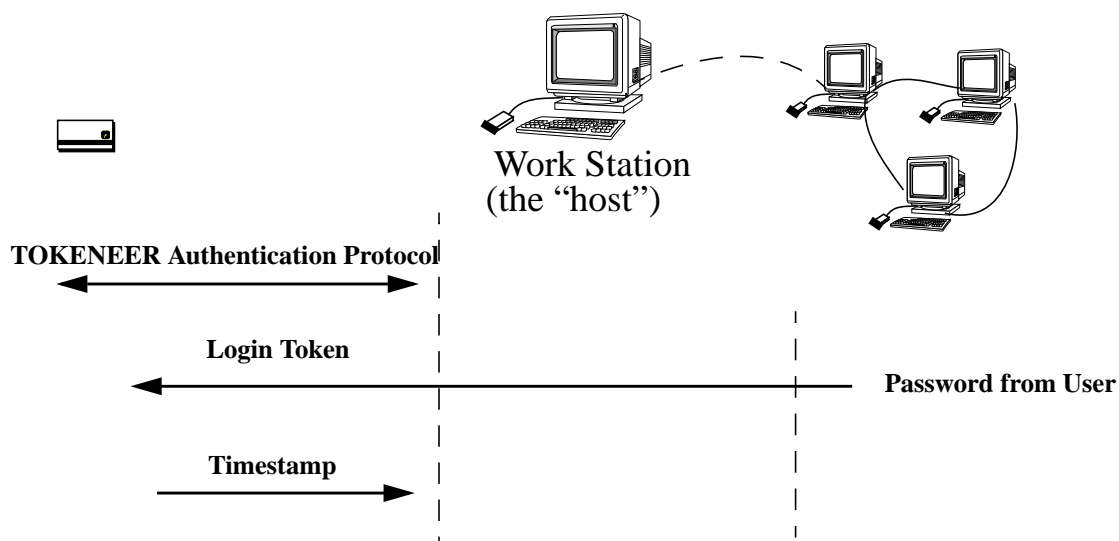
The steps to achieve domain verification are as follows:

1. The token and the local host mutually authenticates each other using the TOKENEER Authentication Protocol (reference [7]). As part of the protocol, the token sends the PrivCert and IDACert to the host over the established encrypted channel.
2. The host validates the **signatureValue** of the PrivCert and IDACert.
3. The **clearance** information of the PrivCert is compared to the classification of the local system. The **clearance** on the certificate must be equal to or higher than those of the local system.
4. The serial number of the token is compared to the **issuerUniqueID** attribute found in the PrivCert and IDACert.
5. The **serialNumber** of the public key certificate is compared to the **owner (baseCertificateID)** attribute found in the PrivCert and IDACert.
6. A livescan biometric sample is taken of the user and processed into a biometric template. The template is compared to a compatible biometric template found in the IDACert.
7. If each step in this sequence occurs, then a timestamped certificate is generated and signed by the ID Station and transferred onto the token.

4.3.2.2 Workstation Verification

The workstation OS's login function must be trusted to perform the functions specified in this section to log the user into the system. The following diagram illustrates the workstation verification process:

User Verification: Workstation Login



The steps to achieve workstation verification are as follows:

1. The token and the local host mutually authenticate each other using the TOKENEER Authentication Protocol (reference [7]).
2. The token sends the PrivCert to the host over an encrypted channel.
3. The host validates the **signatureValue** of the PrivCert.
4. The **clearance** information of the PrivCert is compared to the classification of the local system. The **clearance** on the Certificate must be equal to or higher than those of the local system.
5. The serial number of the token is compared to the **issuerUniqueID** attribute found in the PrivCert.
6. The **serialNumber** of the public key certificate is compared to the **owner (baseCertificateID)** attribute found in the PrivCert.
7. The token provides the timestamped certificate to the Host for validation.
8. The user logs into (provides a password or PIN to) the token.
9. If each step in this sequence occurs, then the user is logged onto the network.

5.0 Conclusions

Using the TOKENEER System as an example, we have successfully demonstrated how the techniques discussed in Parts 1 and 2 can be used to define the best way of placing authentication information into a public key (X.509) certificate. The highlights of the process are:

- Determining the authentication information requirements.
- Assigning the system weights to the desired attribute characteristics
- Using the comparison chart to determine which alternative for attribute placement is best

In order to illustrate how the certificates could be used, the system development process was augmented by:

- Defining the certificate structure needed by the system.
- Developing system scenarios to illustrate the use of the certificates.

APPENDIX I - TOKENEER Privilege Certificate ASN.1 Syntax

-- From X.509

AttributeCertificate ::= SIGNED { AttributeCertificateInfo }

AttributeCertificateInfo ::= SEQUENCE {

versionVersion DEFAULT v1,

subjectCHOICE {

baseCertificateID[0]IssuerSerial, -- associated with a Public Key Certificate

subjectName [1] GeneralNames }, -- associated with a name

issuer GeneralNames, -- CA issuing the attribute certificate

signature AlgorithmIdentifier,

serialNumberCertificateSerialNumber,

attrCertValidityPeriodAttCertValidityPeriod,

clearance Clearance, -- TOKENEER attributes added to the attribute certificate

role SigOrKMPPrivileges ,

subjectUniqueID,

issuerUniqueID UniqueIdentifier OPTIONAL,

extensions Extensions OPTIONAL }

IssuerSerial ::= SEQUENCE {

issuer GeneralNames,

serial CertificateSerialNumber,

issuerUID UniqueIdentifier OPTIONAL }

AttCertValidityPeriod ::= SEQUENCE {

notBeforeTimeGeneralizedTime,

notAfterTimeGeneralized Time }

clearance ATTRIBUTE ::= {

WITH SYNTAXClearance

IDid-at-clearance }

-- From X.509

Clearance ::= SEQUENCE {

policyIdOBJECT IDENTIFIER,

classListClassList DEFAULT {unclassified},

securityCategoriesSET OF SecurityCategory OPTIONAL }

```
ClassList ::= BIT STRING {
```

```
    unmarked(0),
    unclassified(1),
    restricted(2),
    confidential(3),
    secret(4),
    topSecret(5) }
```

```
-- From SDN.702
```

```
sigOrKMPrivileges ATTRIBUTE ::= {
```

```
    WITH SYNTAX PrivilegeFlags
    EQUALITY MATCHING RULE sigOrKMMatch
    ID id-sigOrKMPrivileges }
```

```
PrivilegeFlags ::= CHOICE {
```

```
    sigFlags [0] SigPrivFlags,
    kmFlags [1] KmPrivFlags }
```

```
SigPrivFlags ::= SEQUENCE {
```

```
    sigPrivId OBJECT IDENTIFIER,
    sigPrivFlags SEQUENCE OF INTEGER OPTIONAL }
```

```
-- The integer values currently recognized are :
```

```
    -- orgRelease (0),
    -- PCA (1),
    -- PAA (2) --this integer value is obsolete,
    -- guard (3),
    -- AA (4),
    -- SAA (5),
    -- MFI (6),
    -- DSA (7),
    -- MLA (8),
    -- domainManager (9),
    -- securityOfficer (10),
    -- SRA (11),
    -- acAdmin (12),
    -- ORA (13),
```

- MTA (14),
- MS (15),
- **auditManager** (16),
- **netManager** (17),
- rekeyManager (18)

KmPrivFlags ::= SEQUENCE {
 kmPrivId OBJECT IDENTIFIER,
 kmPrivFlags SEQUENCE OF INTEGER OPTIONAL }

-- The integer values currently recognized are :

- rekeyManager (0),
- guard (1),
- auditManager (2),
- readOnly (3),
- netManager (4),
- MLA (5)

-- From X.509

subjectUniqueID UniqueIdentifier (Tokens serial number)

APPENDIX II - TOKENEER ID Authentication Certificate ASN.1 Syntax

-- From X.509

AttributeCertificate ::= SIGNED { AttributeCertificateInfo }

AttributeCertificateInfo ::= SEQUENCE {
 versionVersion DEFAULT v1,
 subjectCHOICE {
 baseCertificateID[0]IssuerSerial, -- associated with a Public Key Certificate
 subjectName [1] GeneralNames }, -- associated with a name
 issuer GeneralNames, -- CA issuing the attribute certificate
 signature AlgorithmIdentifier,
 serialNumberCertificateSerialNumber,
 attrCertValidityPeriodAttCertValidityPeriod,

 authenticationInfo AuthenticationInfo, -- Refer to Part 2 of this study
 attributes added to the attribute certificate

UniqueissuerID,

issuerUniqueIDUniqueIdentifier OPTIONAL,
extensions Extensions OPTIONAL }

IssuerSerial ::= SEQUENCE {
 issuer GeneralNames,
 serial CertificateSerialNumber,
 issuerUID UniqueIdentifier OPTIONAL }

AttCertValidityPeriod ::= SEQUENCE {
 notBeforeTimeGeneralizedTime,
 notAfterTimeGeneralized Time }

APPENDIX III - TOKENEER Timing Analysis

In general, the greatest amount of time will be spent performing the TOKENEER Authentication Protocol. A summary of timing information is included below. The complete timing analysis for the TOKENEER Authentication Protocol is located in reference [7]. The smartcard I/O rate is set to 32000 bits/sec to allow for protocol overhead (the actual rate is 38400 bits/sec). The public key certificate is estimated at 580 octets in size. The sum of all attribute certificates is limited to 2000 octets (for 2.5K bytes of storage). APPENDIX IV - TOKENEER Attribute Certificate Sizing has more detailed information on attribute certificate sizes. This timing information is typical of a commercially available smartcard using a 512 bit RSA key in the late 1997 time frame.

Authentication Request = Cert_H || ACertID₁ || ... || ACertID_N

- estimated processing time: 190 msec
- estimated I/O time: 150 msec = 599 bytes * (8bits/byte / 32000 bits/sec)

Token_{SH1} = RSA[Rand_S]^H || Cert_S || RSA[β]^H || (ACertID_i || ... || ACertID_j)

- estimated processing time: 1725 msec
- estimated I/O time: 182 msec = 727 bytes * (8bits/byte / 32000 bits/sec)

Token_{HS} = RSA[Rand_H]^S || β || Cert_X

- estimated processing time: 1725 msec
- estimated I/O time: 177 msec = 707 bytes * (8bits/byte / 32000 bits/sec)

KEY = Rand_S || Rand_H

Token_{SH2} = DES[Rand_H || ACert₁ || ... || ACert_N]^{KEY}

- estimated time: 1315 ms
- estimated I/O time: 518 msec = 2072 bytes * (8bits/byte / 32000 bits/sec)

Total estimated time for TOKENEER Authentication Protocol: 6.722 seconds

Additional time will be spent at the IDStation obtaining a livescan biometric sample. This operation is relatively fast, and if performed in parallel with the smartcard operations, will add no additional time. If done sequentially, the operation will add from 1 to 5 seconds. Generation and transfer of a timestamped certificate will add less than 500 msec.

At the workstation, additional time will be required to transfer and process the timestamped certificate. As the processing is done by the host, the delay should be minimal, less than 500 msec. If a password or PIN is required, this will add a variable amount of delay to the login, but as this portion is interactive, the user will not notice it.

APPENDIX IV - TOKENEER Attribute Certificate Sizing

Tables 3 and 4 contain the contents of the two attribute certificates to be used by the TOKENEER system. This information was primarily obtained from reference [14].

Table 3: Contents of Attribute Cert - Privilege Cert

item	item size	# of items	Total Size
version	5 octets	1	5 octets
owner (baseCertificateID)	8 octets	1	8 octets
issuer (AA)	183 octets	1	183 octets
signature	9 octets	1	9 octets
serialNumber	6 octets	1	6 octets
validity	32 octets	1	32 octets
authenticationInfo - role	*	1	*
clearance	42 octets	1	42 octets
issuerUniqueID (Token Serial #)	16 octets	1	16 octets
algorithmIdentifier	9 octets	1	9 octets
signatureValue	70 octets	1	70 octets
Total			r + 380 octets

* - Sizing information for the role attribute was not available. However, it is not expected to be any larger than that of the clearance attribute.

Table 4: Contents of Attribute Cert - Identification & Authentication Cert

item	item size	# of items	Total Size
version	5 octets	1	5 octets
owner (baseCertificateID)	8 octets	1	8 octets
issuer (AA)	183 octets	1	183 octets
signature	9 octets	1	9 octets
serialNumber	6 octets	1	6 octets
validity	32 octets	1	32 octets
authenticationInfo -biometricInfo	500 octets	1	500 octets
issuerUniqueID (Token Serial #)	16 octets	1	16 octets
algorithmIdentifier	9 octets	1	9 octets
signatureValue	70 octets	1	70 octets
Total			838 octets