Vol. 37 No. 5 Transactions of Information Processing Society of Japan May 1996

Regular Paper

Using BAN Logic for the Proof of a Network Address Registration Protocol

YUKO MURAYAMA†

One of the main problems with the current computer network environment is that information held in information systems is not necessarily trustworthy. We look particularly at addresses as an example of network configuration information, and list the serious consequences of incorrect addresses. The current address information flow is examined, and then improved by the use of "certificates." This paper uses the logic introduced by Burrows, Abadi, and Needham, called BAN logic, to prove the flow of the procedure for issuing a certificate. Our use is somewhat different from the original use of the logic for authentication protocols as regards the way in which information integrity is examined.

1. Introduction

Just as the workings of our society are based on information, so also are the operations of computer networks. We call a system whose operation is based on information an information-based system. How well the system works depends on the *credibility* of the information used. This paper is concerned with the semantic level of information; that is to say, how trustworthy is the information being carried?

Information systems, such as name servers and routing information exchange systems, have been traditionally responsible for maintenance of the information used in network operations. The current problem with these systems is that there is no way of knowing how reliable the information is.

In this paper, we look at network addresses as an example of information used by computer networks, and list the problems caused by incorrect network addresses. We propose a registration protocol for issuing a "certificate" for a network address, so that networks can be protected from some of these problems. The protocol is examined in a formal manner, using the logic introduced by Burrows, Abadi, and Needham[2]. Although Nessett argued that the logic intentionally ignores the confidentiality aspect in order to simplify the process[1),13)], this is not a problem with our use of the logic, because our primary concern is to examine information integrity.

The organization of the paper is as follows. The next section describes the current address information flow, and discusses its problems. Section 3 proposes a solution based on the use of certificates, and presents a registration protocol for issuing certificates. Section 4 examines the protocol by using BAN logic. Section 5 suggests the need for adding the information path to the logic. Section 6 offers some conclusions.

† Faculty of Information Sciences, Hiroshima City University

2. The current network address information flow and its problems

We look at the flow of an address from its generation to its registration. The current address flow is as follows. In the resource admission phase, a host is admitted to be configured. In the naming phase, an address is assigned to the host. In the configuration phase, the host is introduced into the networks. In the registration phase, the host starts propagating its address over the networks.

In such an environment, it is possible for any host to start operating with an address that is not intended to be assigned to it. This could happen either inadvertently or with malicious intent. In the Internet environment, it could lead to the following types of network-level threat:

(1) **Unauthorized tampering**
(2) **Unauthorized use of the resource (resource stealing)**
(3) **Doubled traffic**
(4) **Denial of services as a result of a network storm**

Unauthorized tampering results in network packets being redirected to a bogus host, giving it control over the network packets, through misuse of a routing information ex-

change protocol.

To obtain **unauthorized use of the network resource**, a deceiver may steal network bandwidth without being noticed, by using an unassigned local address. A deceiver may gain unauthorized access to a network by communicating with another bogus host.

The **doubled traffic** and **network storm** problems are as follows. The network traffic would be doubled if two nodes had the same lower-layer address[11]. A network storm would be invoked if more than two routes had the same lower-layer address. This effect could be invoked easily by the misconfiguration of a host with a broadcast address, because all routers take packets sent with the subnet-level broadcast address, and try to forward them[10].

3. Use of a certificate and a registration protocol

3.1 Overview

What is missing in the current network system is that there is no mechanism for verifying the credibility of an address and for binding a host and an address. One might argue that having mechanisms for detecting improper addresses would be enough, without any protection mechanisms. However, we believe that the latter are appropriate, particularly in "plug-and-play" environments managed by novices, who may not know what to do even if trouble is detected and reported.

We propose the use of address certificates in address resolution to prevent the problems listed in the previous section, and present a registration protocol for issuing a certificate. The idea is that only certified addresses would be used in address resolution; a certificate would only be given to a host whose configuration had been verified. This would prevent a source node from sending a packet to a bogus host, because the node would invoke address resolution to locate the destination. In this way, packets would not be directed to bogus routers, even if the routing information was manipulated illegally; this would prevent the unauthorized tampering problem. As a host with an unassigned address would not be given a certificate, it would not be able to communicate with other ordinary hosts that use address resolution to locate the communicating party; in other words, bogus hosts would not be able to connect with the servers. However, if two hosts communicated without address resolution, they would still be able to steal the bandwidth. In that case, we could detect two such parties communicating with unauthorized addresses only by monitoring the network with a list of certified addresses. The doubled traffic and network storm problems would be prevented, firstly because a misconfigured host would be detected when the host configuration was verified and secondly because a source node could not send traffic to such a host without a certificate.

Address resolution is the translation of a network address into a lower-layer address such as a MAC address in a LAN. A certificate includes network and lower-layer address mapping, and is issued by a certification authority (CA) only when the host configuration is verified over the network.

The next subsection discusses some modifications to the current address information flow, highlighting two additional phases. Section 3.3 introduces the agents involved. Section 3.4 outlines a procedure on how a certificate can be issued.

3.2 Our model of address information flow

Our scheme includes adding two processes to the current address information flow: certification registration, and configuration confirmation (see **Fig. 1**).

The network and lower-layer addresses, along with other information on the configuration of the host, including cryptographic system attributes, are registered off-line in the certificate registration phase. If the registration is successful, a registration ID is issued. The host initiates configuration confirmation with the registration ID, and the configuration of the object is verified. If the verification is successful, an authority will issue a certificate for the network address, which will be registered in the relevant information systems. The certified address mapping of the new host can be found either through a dynamic address resolution operation such as the Ethernet Address Resolution Protocol[14] or from some other static database.

3.3 The agents involved and their functions

The following agents are involved in our registration procedure:

- Naming authority (NA)
- Management system (M)
- Certification authority (CA)
- Information system (R)
- System (A)

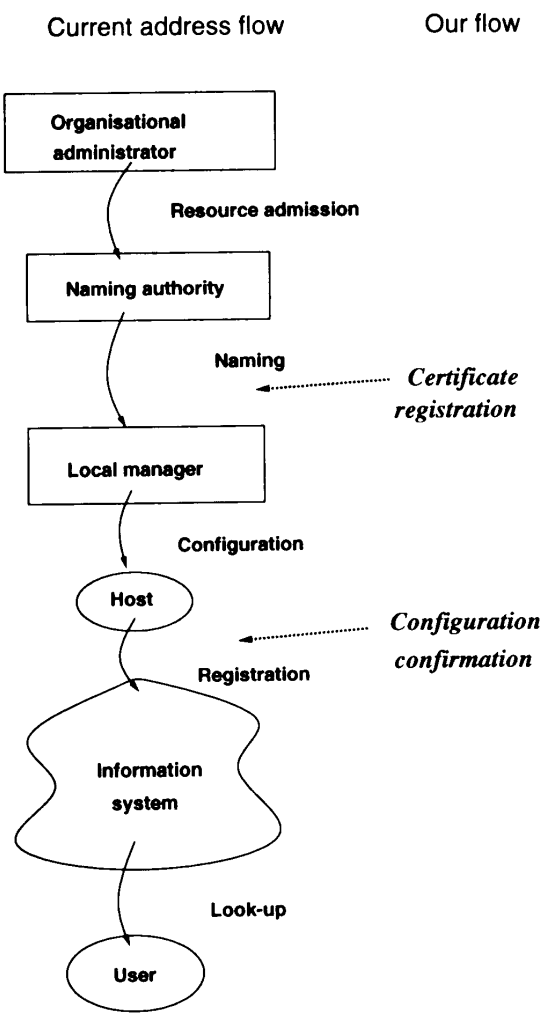


Fig. 1 Our model of network address flow.

- Host (h)

The naming authority (NA) assigns a network address to a network interface, and is supposed to do it reliably — that is, it checks for duplicate addressing. On the other hand, lower-layer addresses are pre-assigned by a naming authority (possibly a vendor) outside the organization, but it would be possible for a local user to change them. Their reliability, therefore, is unknown. However, our certification authority (CA) is supposed to check up on duplicate lower-layer addresses as well as network addresses before issuing a certificate.

In a physical system (A), a network object called host (h), which is composed of a set of network processes, would reside with the allocated addresses. When A starts up, it initiates configuration confirmation for h, reporting its registration ID to the management system (M).

CA functions in two ways: one is as an off-line registrar in certification registration, and the other is as an authority for issuing a certificate after configuration confirmation. It does issue certified information, in a similar way to (CA) described in X.509[4], though it does so only if an information item has been verified recently by M. Our CA also maintains a list of certified address pairs, and checks up on duplicate addresses before issuing a new certificate.

M is in charge of verification of the configuration of h; it compares the information obtained from CA with the current configuration of h.

An information system (R) is responsible for maintenance of information items associated with the objects that are used for a specific network operation. There are two types of information systems, static and dynamic. The static ones are basically database systems, and the dynamic ones are dynamic learning systems in which a host is an information provider.

3.4 A registration protocol

In the following we introduce a protocol for both configuration confirmation and registration. **Figure 2** shows the protocol flow.

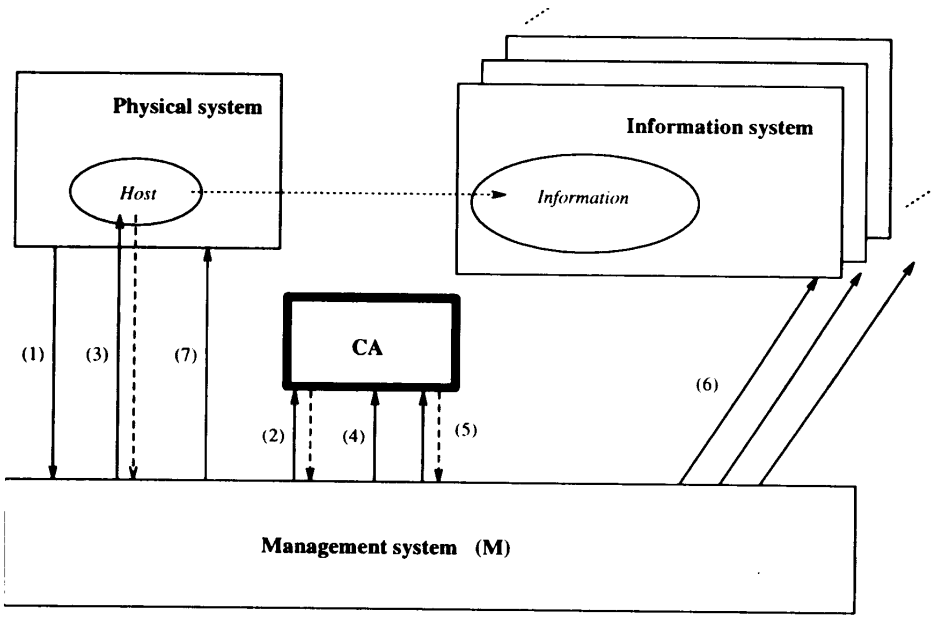
The configuration confirmation sequence is as follows:

(1) A system, A, uses the network to inform the manager, M, of the completion of the configuration of network software, h, using the registration ID.
(2) The manager, M, gathers the information on h, including the associated registration ID that was issued at the time of certificate registration.
(3) The configuration of h is verified, using the previously registered information.
(4) Upon successful verification, the manager recognizes the addition of h.
(5) The lower-layer and network addresses are certified and a certificate is produced.
(6) The certificate is registered in the information system.
(7) The certificate is also given to the host.

A certificate for the network and lower-layer address mapping issued in step 5 will be used in the address resolution operation. The certificate could be as follows:

$$\{\text{lower-layer address, network address, time stamp}\}SK_{CA}$$

where the lower-layer address and the associated network address are encrypted by using the secret key of CA. The certificate could be decrypted by receivers using the well-known public key of CA.



Legend:

(1): A physical system initiates the registration of a host with a registration ID.
(2): M requests information associated with the registration ID.
(3): M verifies that the host is operational.
(4): M reports to CA that the host is operational.
(5): M asks CA for certification.
(6): M distributes the updates to the information systems.
(7): M gives the certificate of the host to the system.

——→ : Request
←------ : Reply

Fig. 2 Flow of configuration confirmation and registration.

4. Formal analysis of the protocol

4.1 Overview

We wish to verify our registration protocol formally, so that we can be sure that the protocol achieves what it is expected to achieve.

We are interested in showing whether the original goal of our registration protocol is achieved or not. The purpose of the protocol was to register successfully a certified network address, so that only the authorized hosts configured with the valid addresses can participate in network operations. This prevents deceivers and misconfigured hosts from causing the four problems described in Section 2. The registration is done by verifying a host configuration and issuing a certificate when the verification is successful. We therefore need to show that through the protocol sequence each participant can be confident that a host has really been configured with an assigned address.

Formal methods have been primarily concerned with verification of the following operational aspects of systems[6]:

- Safety — nothing bad will happen
- Liveness — something good will happen

In concurrent systems and communication protocols, for instance[15], safety properties would include mutual exclusion, and a liveness property would be required, to ensure that deadlocks are prevented.

Various verification methods have been used, from state transition models to temporal logic[7]. Model checking could be used to show, for instance, whether the system requirements in a temporal logic formula are satisfied in the system given as an explicit transition system with a finite number of states[5]. One could use those methods to verify operational aspects such as the concurrency of a protocol.

The reason that we need formalism is to show the credibility of an information item. In other words, we are interested in showing that every party involved in the protocol believes cer-

tain information. This is similar to an authentication problem. Authentication is a procedure by which communicating principals satisfy themselves mutually about each other's identity. Usually, the principals share a secret, and the objective of authentication protocols is that each principal should believe such a secret. Indeed, our protocol is similar to an authentication protocol in the following respects:

- It requires one participant to have the authority to create a belief.
- Cryptographic technique is used for integrity, not for confidentiality.
- The protocol could be considered as sequential, with a final state.

There have been many modal logics of belief[9], although they are not often incorporated in formal methods for computer science. A logic introduced by Burrows, Abadi, and Needham[2], known as BAN logic, is based on such logics, and offers many operators for dealing with beliefs based on time present and past. It was originally designed for analyzing authentication protocols at a fairly abstract level, and was used successfully to identify the existence of flaws in several well-know protocols. Indeed, authentication protocols were first formally analyzed by using this logic. The transformation of a protocol for analysis in BAN logic is following the message sequence and is much more straightforward than in temporal logic and other formal methods; this is very helpful for a novice user of formal methods. For these reasons, we use BAN logic.

We introduce this logic in the next subsection. Section 4.3 describes the goal of our analysis, and Section 4.4 states the assumptions. Section 4.5 presents our analysis of the protocol.

4.2 Introduction to BAN logic

BAN logic is based on modal logics. It is designed specifically for analysis of authentication protocols at a fairly abstract level.

Protocol analysis using BAN logic is performed as follows:

- The idealized protocol is derived from the original one.
- Assumptions about the initial state are written down.
- Logical formulas are attached to the statements of the protocol, as assertions, in order to discover the beliefs held by the parties in the protocol.

The above procedure is repeated as new assumptions and assertions are found to be necessary, and as the idealized protocol is refined.

A protocol is a sequence of "send" statements, $S_1, \ldots, S_n$, each of the form $A \rightarrow B : X$ with $A \neq B$; A is a sender and B is a receiver of the message X. In an idealized protocol, the message is transformed into a logical formula. The idealized protocol is annotated with assertions, much as in a proof in Hoare logic[8]:

$[assumptions]S_1[assertion1]S_2 \ldots$
$\ldots[assertion\ n-1]S_n[conclusions].$

The syntax of the logic is as follows. There are three types of objects: principals, encryption keys, and formulas. Keys in public key cryptosystems are annotated in BAN logic as $\overset{K}{\mapsto} A$ for the public key of principal A and K^{-1} for the secret key; however, in our proof in a later section, we use the term PK_A for a public key and SK_A for a secret key. The propositional connective is a conjunction, annotated by a comma, with properties such as associativity and commutativity. Moreover, there are ten constructs, of which we use the following six:

$A \equiv X$: means that A believes that X is a true information item.

$A \Rightarrow X$: means that A has jurisdiction over X.

$\sharp(X)$: originally means that the information item X was generated recently. We modify this definition slightly to state that the production of X has been validated and verified recently; that is, X was recently perceived as correct; however, it is not clear whether it was generated recently.

$A \triangleleft X$: A sees X.

$A \mid\sim X$: A once said X.

$\{X\}_K$: a formula X is encrypted under the key K.

We introduce the following new notation, which indicates that the information item has been notarized by a trustful authority:

$\langle\langle X \rangle\rangle_C$: X is certified by the authority C.

In authentication, the concept of timeliness is important for detecting the replay of a message used in the past. Consequently, BAN logic has two distinguished epochs, present and past. The present epoch begins at the start of the particular run of of the protocol message exchange. All beliefs held in the present are stable for the entirety of that exchange; when principal A says X, she is supposed to believe X. However, beliefs held in the past are not necessarily carried forward into the present.

The following semantics, according to which principals develop beliefs, are given to the logic. In order to obtain new beliefs, principals are supposed to examine their current beliefs and apply rules. Several postulates are provided, but we describe only the three that we use.

The first rule is that if A knows B's public key, PK_B, and A sees the information item, X, encrypted with B's secret key, SK_B, then A believes that B once said X:

$$\frac{A \models PK_B, A \triangleleft \{X\}SK_B}{A \models B \mid\sim X}$$

The second rule is that if A believes X was uttered only recently and B once said X, then A believes that B has recently said X:

$$\frac{A \models \sharp(X), A \models B \mid\sim X}{A \models B \models X}$$

The third rule is that if A believes that B has jurisdiction over X, then A trusts B as regards the truth of X:

$$\frac{A \models B \models X, A \models B \Rightarrow X}{A \models X}$$

The public key cryptosystem assumption is that a secret key is only known to one principal, whereas the public key is known to many principals, since it is supposed to be known publicly[12].

We should note that the semantics of BAN logic give a meaning to only two operators, $\models$(*believes*) and $\triangleleft$(*sees*).

BAN logic is used to examine the knowledge obtained by participants at the end of each protocol message. It is particularly useful for examining whether some information has flowed as expected by a protocol designer. We are interested in how an allocated address is passed by its allocator, the naming authority, to the host and the information systems, and how its integrity is preserved.

In later subsections, we also use the following abbreviations:

- *Reg* : the registration ID
- X_{anet} : the network address of h
- X_{alow} : the lower-layer address of h
- X_d : the description of h
- $X_c(Z)$: the cryptographic system attributes of Z
- T_Z : the time stamp of Z

4.3 The goal of analysis

It is quite useful to express the goal of a protocol by using BAN logic notation — that is, what knowledge each participant has to obtain — as it will show any protocol fraud if the goal is not achieved during protocol analysis.

Since the notation we use was originally intended for authentication, the successful result of a protocol between two principals, A and B, is that both principals know the shared secret, and each also knows that the other knows it. The notation for this result is as follows:

$$A \models X \text{ and } A \models B \models X$$

and

$$B \models X \text{ and } B \models A \models X$$

However, our goal is different, in that we need to ensure that the address mapping information, for example, (X_{anet}, X_{alow}), generated by the *information originator*, NA, and verified by CA, is assigned to the *information provider*, A, as is intended, and eventually arrives at an *information system*, R.

We use the the following notations:

- $Bind(Xanet) = Xalow$ means that the network address, *Xanet*, is assigned for the network interface with a lower-layer address, *Xalow*.
- $Claim(Xalow) = Xanet$ means that the network interface with the lower-layer address *Xalow* is claiming that its associated network address is *Xanet*

The goals are then expressed as follows:

(1) $NA \models Bind(\mathrm{X}anet)=\mathrm{X}alow$
(2) $NA \models Claim(\mathrm{X}alow)=\mathrm{X}anet$
(3) $CA \models NA \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet$
(4) $CA \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet$
(5) $A \models NA \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet$
(6) $A \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet$
(7) $R \models CA \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet$
(8) $R \models NA \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet$
(9) $R \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet$

The first goal signifies that the naming authority NA believes the assignment of a host address. We should note that NA assigns an address to the host, but does not know whether the address is valid, in the sense that it is workable in the actual network, until the certification authority CA reports the real configuration. The second goal signifies that NA believes the current address configuration.

The third goal signifies that CA has to know that NA confirms that the assigned address is really configured to the host to which the address has been assigned by NA. The fourth goal signifies that CA also has to believe that.

The fifth and sixth goals signify that the host system A believes its own address configuration and also that the configuration has been confirmed by NA.

The seventh and eighth goals signify that a static type of information system, R, such as a network management database system, comes to believe that the address configuration information being registered has been confirmed by the authorities, CA and NA. The ninth goal signifies that the information system believes the address information as well.

In the network operations, the actual participants are hosts and routers, both of which we denote as *hosts*(A) in our model, as well as information systems (R). It is therefore essential that A and R should believe in the validity of the addresses in use. Such a belief has to be based on the existence of some authority, as in authentication. We have two authorities, one in charge of address assignment (NA) and the other in charge of certificates (CA). Those two need to agree on the correct configuration of a host. We have to note that, in order to achieve the above objectives, it is essential for the management system (M) to acquire a belief in the validity of an address, but that it is not a goal in itself.

Such goals together ensure that an address certificate is issued to a host when it is confirmed by the authority that the host is configured with the assigned address. In network operations such as address resolution, the host will send its address to others by using its certificate, so that the other hosts do not send packets to a bogus host that has never had a certificate. This prevents packets being redirected to a bogus router (the unauthorized tampering problem) and also prevents resource stealing partly. Doubled traffic and network storms can be avoided, because double addressing is examined before the issue of a certificate by CA. The first goal signifies that NA believes its address assignment after the configuration has been verified and double addressing has been checked by CA.

4.4 Assumptions

We make the following assumptions:

Assumption 1: Since CA generated a registration ID, *Reg*, it knows the freshness of *Reg*.

$CA \models \sharp(Reg)$

Assumption 2: $CA \models NA \Rightarrow Bind(\mathrm{X}anet)=\mathrm{X}alow$

Assumption 3: The timers in all the agents involved are synchronized, so they believe each others' time stamps.

For $x \in \{ M, R, CA, A \}$,

$x \models (\sharp(T_M), \sharp(T_{CA}), \sharp(T_R), \sharp(T_A))$

Assumption 4.1: A, M, and R believe that CA has authority to issue *Reg* associated with a network object whose attributes are X*alow*, X*d* and X*c*.

$\{ A, M, R \} \models CA \Rightarrow (Reg, \mathrm{X}alow, \mathrm{X}d, \mathrm{X}c)$

Assumption 4.2: A, M, and R believe that NA has the authority to assign a network address, X*anet*.

$\{ A, M, R \} \models NA \Rightarrow (\mathrm{X}anet)$

Assumption 5: A, M, and R believe that CA has authority to declare that an assigned network address is really configured to the designated host correctly.

$\{ A, M, R \} \models CA \Rightarrow Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet$

That is, if

$\{ A, M, R \} \models CA \vdash x,$

where $x \in \{ Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet \}$,

then $\{ A, M, R \} \models x$.

Assumption 6: CA knows that A, M, and R believe that CA has authority to issue a certificate stating that an assigned network address is really configured to the designated host correctly.

$CA \models \{ A, M, R \} \models$

$CA \Rightarrow \{ Claim(Bind(\mathrm{X}anet)) = \mathrm{X}alow \}$

Assumption 7: A, M, and R believe that NA knows that an assigned network address is really configured to the designated host correctly, if and only if A, M, and R know that CA believes so.

$\{ A, M, R \} \models NA \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet,$

if and only if

$\{ A, M, R \} \models CA \models Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet.$

Assumption 8: CA believes that M has the authority to declare verified information as follows:

$CA \models M \Rightarrow$ (Add, object ID, information)

CA knows that M would report the result after verifying the configuration of the host.

Assumption 9: A host has authority to announce its own configuration.

$M \models A \Rightarrow (Claim(\mathrm{X}alow) = \mathrm{X}anet)$

Assumption 1 states that CA believes its own generation of a registration ID; it therefore requires some kind of number generator. The number generation is not strict in comparison with the one that generates nonce in authentication protocols, because the registration number is used only as a hint for the management system to retrieve configuration information from CA.

Assumption 2 is that the authority CA believes that the function of the other authority, NA, is address assignment.

Assumption 3 is that time is synchronized throughout this network environment in which the registration takes place. In distributed systems, we can make such an assumption.

It is necessary to have an authority to distribute belief to participants in the network operation. We set up two authorities, namely, CA and NA. Assumption 4.1 is that CA is believed to have the authority to issue a registration ID and the host configuration information that CA has obtained off-line in the certificate registration phase, as described in Section 3.2. Assumption 4.2 is that NA is believed to have the authority to assign a network address.

Assumption 5 is that CA has the authority to issue an address certificate, that will be believed by the others. This is the basic meaning of a certificate.

A certificate is believed because it is issued by the trusted authority, CA. This is Assumption 6: that the participants in the registration operation — the host, the management system, and the information system — believe in CA's authority to issue a certificate, which is its basic function.

A certificate not only shows that CA believes the assigned address is configured correctly, but also that NA confirms it. Assumption 7 is that the host, the management system, and the information system all know this.

In our model, the management system reports the verification result to CA. Assumption 8 is that CA understands that the management system has the right to do this.

The management system has to verify the actual configuration of a host by requesting some configuration attributes, including an address to return. Assumption 9 is that a host is supposed to send its current address. In practice, a network management system accepts such a reply from a managed object, particularly if the speaker is authenticated[3]. In BAN logic, such authentication is guaranteed if the message is encrypted with the sender's secret key; see the first postulate in Section 4.2.

We also assume that relevant public keys have been distributed previously to the manager system M, the information system R, and the object site system A.

4.5 Protocol analysis

The protocol can be summarized as follows:

A system, A, informs the manager, M, of the completion of the configuration of network software, h, using the registration ID.

Message 1:

$A \rightarrow M$: [Report, h's ID, $\{ Reg, T_A \} PK_M$]

The manager, M, asks for the information on h associated with the registration ID, which was registered off-line long before, during certificate registration.

Message 2.1:

$M \rightarrow CA$: [Request, h's ID, $\{\{Reg, T_M\} SK_M\} PK_{CA}$]

CA returns the attributes of h to M.

Message 2.2:

$CA \rightarrow M$: [Reply, h's ID, {{ Reg, $Bind(\mathrm{X}anet) = \mathrm{X}alow$, $\mathrm{X}d$, $\mathrm{X}c(A)$, $T_M + 1$ $\}SK_{CA}\}PK_M$]

The configuration of h is verified by M; M asks the host what address and other attributes it is configured with.

Message 3.1:

$M \rightarrow A$: [Request, { h,ID.Xa,ID.Xd,$T_M\}SK_M$]

A replies by informing M of the network address (XX$anet$) that the network interface (XX$alow$) is actually configured with.

Message 3.2:

$A \rightarrow M$: [Reply, { $Claim(\mathrm{XX}alow) = \mathrm{XX}anet$, $\mathrm{XX}d$, $T_M + 1\}SK_A$]

The current configuration of h is compared with the information obtained from CA: XX$anet$= X$anet$, XX$alow$= X$alow$, and XXd = Xd. If they match, M reports to CA that the verification is successful; otherwise the procedure stops here.

Message 4:

$M \rightarrow CA$: [{ Report, Add,h, $Claim(\mathrm{X}alow) = \mathrm{X}anet$, $\mathrm{X}d$, $\mathrm{X}c$(A), $T_M\}SK_M$]

M asks CA for a certificate of the network address, X$anet$.

Message 5.1:

$M \rightarrow CA$: [{ Request, h, X$anet$, $T_M\}SK_M$]

CA issues a certificate; this includes the lower-layer and network addresses with time stamps, encrypted with CA's secret key.

Message 5.2:

$CA \rightarrow M$: [{ Reply, h, $\langle\!\langle Bind(X anet) = X alow, T_{CA} \rangle\!\rangle_{CA}$, $T_M + 1\}SK_{CA}$]

The certificate is registered in a static type of information system such as a database system for network management R.

Message 6:

$M \rightarrow R$: [{ Report, h, $\langle\!\langle$ $Bind$ (X$anet$) = X$alow$, $T_{CA} \rangle\!\rangle_{CA}$, $T_M\}SK_M$]

The certificate is also given to the host for use in dynamic types of information system such as address resolution.

Message 7:

$M \rightarrow A$: [{ Set, h,
$\langle\langle$ *Bind*(X*anet*) = X*alow*, $T_{CA}\rangle\rangle_{CA}$, $T_M\}SK_M$]

We now describe protocol analysis. However, we do not present an idealized protocol here; instead, we present idealized operations and analysis together.

With Message 1, a registration ID is sent from a host in clear text, so that only the management system sees it:

$$M \triangleleft Reg$$

From Message 2.1, CA sees the registration ID encrypted by the manager's secret key, and believes that the manager uttered the ID:

$$CA \models M \mid\sim Reg$$

From Message 2.2 from CA, the manager M obtains the information associated with the host that sent the registration ID in Message 1. Since M knows that CA is entitled to announce such information, it believes what it sees in the message:

$$M \models CA \mid\sim$$
$$(Reg, Bind(X anet) = X alow, X a, X d, X c(A)),$$

and

$$M \models CA \Mapsto$$
$$(Reg, Bind(X anet) = X alow, X d, X c(A));$$

hence,

$$M \models (Reg, Bind(X anet) = X alow, X d, X c(A)).$$

M now verifies the configuration of the host by sending Message 3.1, which requests the host to send its configuration information. From Message 3.2, the reply from the host, M learns that A has uttered the current address. Since the message includes $T_M + 1$, which shows that the message was sent recently as a reply to Message 3.1, M believes that A believes its current configuration; the network address XX*anet* is configured to the network interface with the lower address XX*alow*.

$$M \models A \models (Claim(XX alow) = XX anet, XX d).$$

M then verifies locally whether the configured address is the same as the assigned address obtained from CA as follows: XX*anet*= X*anet*, XX*alow*= X*alow*, and XXd = Xd.

If they match, M believes that A believes its current configuration as follows:

$$M \models A \models (Claim(X alow) = X anet).$$

From Assumption 9, we deduce the following:

$$M \models (Claim(X alow) = X anet).$$

From Message 4, M reports that the host A has been configured with the assigned address.

$$CA \models M \models A \models (Claim(X alow) = X anet).$$

From Assumption 8,

$$CA \models M \Mapsto (Add, h, Claim(X alow) = X anet);$$

hence,

$$CA \models Claim(X alow) = X anet.$$

On receiving Message 5.1 requesting a certificate from M, CA remembers that it already knows that the network address X*anet* has been assigned to a network interface with the lower address X*alow*:

$$CA \models (Reg, Bind(X anet) = X alow),$$

and from Message 4,

$$CA \models Claim(X alow) = X anet.$$

Thus CA could issue a certificate straight away. However, at this point it checks the inconsistency of addressing by looking through a list of addresses that it has certified. NA will be informed if there is any double-allocation of an address.

Moreover, after the validation there needs to be a transaction (probably off-line) between CA and NA so that they can mutually confirm that the allocated network address, X*anet* and its associated lower-layer address X*alow* are indeed usable, as follows:

$CA \rightarrow NA$: { *Claim*(X*alow*)=X*anet*,
T$_{CA}$ }$_{SK_{CA}}$. (5.1.add1)

If NA perceives that *Bind*(X*anet*)=X*alow*, then it sends the following reply:

$NA \rightarrow CA$: { T$_{CA}$+1}$_{SK_{CA}}$. (5.1.add2)

From (5.1.add1),

$$NA \models CA \models (Claim(X alow) = X anet).$$

From (5.1.add2), we can see that NA confirms that the assigned address is really configured to the host to which the address was intended to be assigned.

To be more precise, we need another assumption, similar to Assumption 9, that NA believes that CA has the right to report on the current configuration to NA:

Assumption 10:

$$NA \models CA \Mapsto (Claim(X alow) = X anet).$$

This assumption is that NA knows that the completion of verification will be reported by CA. Since our scheme has two authorities, this assumption that those authorities respect each other's function is reasonable.

$$NA \models Bind(X anet) = X alow \quad (1),$$

and

$$NA \models Claim(X alow) = X anet \quad (2).$$

From (5.1.add2), CA also deduces that NA confirms that the assigned address is indeed configured to the right host, as follows:

$$CA \models NA \models (Claim(Bind(X anet)) = X anet) \quad (3).$$

CA issues a certificate for

Claim(*Bind*(X*anet*))=X*anet*

in Message 5.2.

From Message 5.2, since CA has sent the cer-

tificate, it must know the current address configuration of the host:

$$CA \models Claim(Bind(Xanet)) = Xanet, \quad (4)$$

$$M \models CA \models Claim(Bind(Xanet)) = Xanet,$$

and from Assumption 5,

$$M \models Claim(Bind(Xanet)) = Xanet.$$

On issuing the certificate, CA should terminate the *Reg*. The timeout limit should be set in *Reg*, in case for some reason the registration is never invoked.

From Assumption 6, CA knows that M knows that CA believes that the assigned address has been configured to the right host.

$$CA \models M \models CA \models Claim(Bind(Xanet)) = Xanet.$$

From Message 6, the static type of information system R knows that CA and NA know that the assigned address is configured correctly:

$$R \models CA \models Claim(Bind(Xanet)) = Xanet. \quad (7)$$

From Assumption 7,

$$R \models NA \models Claim(Bind(Xanet)) = Xanet. \quad (8)$$

From Assumption 8, R believes that as well.

$$R \models Claim(Bind(Xanet)) = Xanet. \quad (9)$$

From Message 7, A receives the certificate of its address configuration.

$$A \models M \vdash (Set, h, \langle\!\langle\ Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet, T_{CA}\ \rangle\!\rangle_{CA}),$$

and

$$A \models M \Rightarrow (Set, \subset g, \langle\!\langle\ Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet, T_{CA}\ \rangle\!\rangle_{CA});$$

hence,

$$A \models (Set, g, \langle\!\langle\ Claim(Bind(\mathrm{X}anet))=\mathrm{X}anet, T_{CA}\rangle\!\rangle_{CA}).$$

That is, A believes that the current address configuration is confirmed by CA.

$$A \models CA \models Claim(Bind(Xanet)) = Xanet,$$

and also

$$CA \Rightarrow Claim(Bind(Xanet)) = Xanet;$$

hence,

$$A \models Claim(Bind(Xanet)) = Xanet. \quad (6)$$

Now A believes its own address configuration, and from Assumption 7, A knows that NA confirms the configuration as well.

$$A \models NA \models Claim(Bind(Xanet)) = Xanet \quad (5)$$

Now that A has received the certificate,

$$A \models Reg.ID,$$

and it knows that the registration has been completed. Q.E.D.

5. The information path

We have managed to use the logic to prove the goals; however, we encountered a little difficulty in doing so. From the beginning of the flow up to the configuration phase, the security of the information paths is unknown. Moreover, our protocol involves more than three participants, so that the structure of the information flow is complex. In particular, after the certificate registration phase, there are multiple paths for the address flow; one goes though to a system that will have the object configured; another goes to the on-line CA. We have no clear way of differentiating the information that is set to the host in the configuration phase from the information stored in CA. For instance, in our proof, we denoted XX_{anet} as the network address in the host, and X_{anet} as the one stored in CA. For a precise examination of the information flow, we may need some more explicit way of expressing what an information item has been coming through, namely, information paths. We could specify that $M \models h \models NA \models X$, but if we specify that $M \models (((X)_{NA})_h)$, it will imply that h might have made a mistake in announcing X.

6. Conclusion

The main problem in current networks is that there is no way of knowing how reliable the information is. The reliability in operation of an information-based system such as a network, depends on how reliable the information is. The information system is responsible for the information used in network operations. We looked particularly at information registration.

The protocol consists of two levels of operation: an *engineering-level* check and a *semantic-level* check. The basic verification at the *engineering level* is to maintain the data integrity. Moreover, authentication would be needed for a sender; however, our main purpose is to ensure that given information is correct at the *semantic level*. This is done by verification and authorization.

We have introduced our protocol in terms of transactions between management agents. The protocol was analyzed formally by using the logic recently introduced by Burrows, Abadi, and Needham. We have proved how the integrity of the information, an address, is maintained; however, our protocol requires us to incorporate a few additional transactions, so that the naming authority comes to know that the assigned address is configured to the designated host correctly.

We attempted to use formal notation and logic, intended originally for proving authentication protocols, to prove the integrity of the information flow. One difficulty was to differen-

tiate two addresses that originated in the same address but followed different paths. We denote such paths as information paths.

Apart from that, our trial of use of the logic has shown that it is useful for expressing the flow of information on a semantic level.

Acknowledgments The author would like to thank Peter T. Kirstein for his support during the research presented in this paper. Thanks to Mike Burrows, Martin Abadi, Roger M. Needham, and Hisashi Komatsu for the useful insights and information on BAN logic and logic in general. Special thanks to anonymous reviewers for proofreading and providing the author with constructive comments.

References

1) Burrows, M., Abadi, M. and Needham, R.: Rejoinder to Nessett, *ACM Operating Systems Review*, Vol.24, No.2, pp.39–40 (1990).
2) Burrows, M., Abadi, M. and Needham, R.: A Logic of Authentication, Technical Report 39, DEC Systems Research Center (1989).
3) Case, J., Fedor, M., Schoffstall, M. and Davin, J.: A Simple Network Management Protocol (SNMP), RFC 1098 (1989).
4) CCITT and ISO: Recommendations X.500 Series; the Directory – X.509 (ISO 9594-8) Authentication Framework, International Standard X.509 (1988).
5) Clarke, E., Emerson, E.A. and Sistla, A.P.: Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications, *ACM Transactions on Programming Languages and Systems*, Vol.8, No.2, pp.244–263 (1986).
6) Cousot, P.: Methods and Logics for Proving Programs, *Handbook of Theoretical Computer Science*, van Leeuwen, J. (Ed.), Japanese edition, pp.819–962, Maruzen (1994).
7) Gotzhein, R.: Temporal Logic and Applications – A Tutorial, *Computer Networks and ISDN Systems*, Vol.24, pp.203–218 (1992).
8) Hoare, C.A.R.: An Axiomatic Basis for Computer Programming, *Comm. ACM*, Vol.12, No.10, pp.576–580 (1969).
9) Hughes, G.E. and Cresswell, M.J.: *An Introduction to Modal Logic*, Methuen, London (1968).
10) Manber, U.: Chain Reactions in Networks, *IEEE Computer*, Vol.23, No.10, pp.57–63 (1990).
11) Murayama, Y.: Configuration Detection in Computer Networks within an Organisation, *Proc. INET'92*, pp.307–316, Kobe, Japan (1992).
12) Needham, R. and Schroeder, M.: Using Encryption for Authentication in Large Networks of Computers, *Comm. ACM*, Vol.21, No.12, pp.993–999 (1978).
13) Nessett, D.M.: A Critique of the Burrows, Abadi, and Needham Logic, *ACM Operating Systems Review*, Vol.24, No.2, pp.35–38 (1990).
14) Plummer, D.: An Ethernet Address Resolution Protocol, RFC 826 (1982).
15) Schwartz, R.L. and Melliar-Smith, P.M.: From State Machines to Temporal Logic: Specification Methods for Protocol Standards, *IEEE Trans. Comm.*, Vol.COM-30, No.12, pp.2486–2496 (1982).

(Received September 7, 1995)
(Accepted February 7, 1996)

Yuko Murayama received a B.Sc. in mathematics from Tsuda College. She attended University College London, and had an M.Sc. in computer science in 1984, and a Ph.D. in 1992, both from Univ. of London. She had been a visiting lecturer in Keio University from 1992 to 1994. She has been a lecturer in Hiroshima City University since 1994. Her current interests include the security aspects of internetworking and multimedia communications. She is a member of IPSJ, IEICE, ITEJ, ORSJ, ACM, and IEEE.