Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

ELSEVIER

Computer Networks 43 (2003) 601–618

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

# Formal verification: an imperative step in the design of security protocols ☆

Tom Coffey, Reiner Dojen *, Tomas Flanagan

*Data Communication Security Laboratory, Department of Electronic and Computer Engineering, University of Limerick, Limerick, Ireland*

Received 25 September 2002; received in revised form 13 March 2003; accepted 1 April 2003

Responsible Editor: J. Quemada

**Abstract**

Traditionally, security protocols have been designed and verified using informal techniques. However, the absence of formal verification can lead to security errors remaining undetected. Formal verification techniques, on the other hand, provide a systematic way of discovering protocol flaws.

This paper discusses the process of formal verification using modal logics. The verification process is demonstrated by way of case studies on three security protocols, which are designed for use in mobile communications. Our formal analysis discovers all known flaws in the three chosen protocols. Further, a hitherto unknown flaw is identified in these protocols. This flaw causes a protocol failure, which can be exploited in an attack where an adversary impersonates a legitimate protocol participant. A new protocol, resistant to this attack, is proposed and formally verified, giving confidence in the correctness of the protocol.

The result of these case studies, where formal verification successfully discovers all these flaws, demonstrates that using formal verification techniques is an imperative step in the design of security protocols.
© 2003 Elsevier B.V. All rights reserved.

*Keywords:* Formal verification; Security protocols; Modal logics; Cryptography; BCY protocol attack

## 1. Introduction

Secure communications are paramount in today's environment, where mobile and fixed networks are trusted with highly sensitive information. Cryptographic protocols are used to provide security services, such as confidentiality, authentication and non-repudiation. Frequently, informal and intuitive techniques are used to verify such protocols. However, the absence of formal verification of these protocols can lead to

☆ This work was partly funded by the Irish Research Council for Science, Engineering and Technology (IRCSET)—Basic Research Award SC02/237.
* Corresponding author.
*E-mail addresses:* tom.coffey@ul.ie (T. Coffey), reiner.dojen@ul.ie (R. Dojen), tomas.flanagan@ul.ie (T. Flanagan).

1389-1286/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S1389-1286(03)00292-5

flaws and security errors remaining undetected. For example, the public-key authentication protocol of Needham and Schroeder [1] was considered secure for over a decade. Verification of the Needham–Schroeder protocol using formal logics [2] exposed vulnerability to replay attacks in this protocol. This highlights the fact that even comparably simple protocols are difficult to verify by intuitive techniques.

Formal verification aims at providing a rigid and thorough means of evaluating the correctness of a cryptographic protocol so that even subtle defects can be uncovered. Protocol verification methods include state-space searching [3,4], the use of process algebras [5] and logic-based analysis [6–8]. Formal logics are particularity suitable for protocol verification due to their comparative simplicity and effectiveness.

Several protocols have been proposed recently for wireless communications. These include the authentication and key agreement protocol (AKA) used for the universal mobile telecommunication system [9], the Wong–Chan protocol [10], the ASPeCT protocol [11], the Boyd–Park protocol [12] and the BCY protocol [13]. The BCY protocol, originally designed by Beller, Chang and Yacobi, subsequently revised by Carlsen [14] and Mu and Varadharajan [15] demonstrated the feasibility of public-key cryptography in mobile communications.

This paper discusses the importance of formal verification in protocol design. The verification process is demonstrated by way of case studies, where logic-based analysis of the protocols of Beller et al. [13], Carlsen [14] and Mu and Varadharajan [15] is performed. A previously unknown failure is discovered which allows an intruder to masquerade as a user of mobile services by exploiting the inability of the service provider to recognise the replay of an old session key. This weakness results in the failure of the protocol to provide authentication and key agreement. In addition to discovering this new failure, the verification also detects all known flaws associated with the BCY protocol and its derivatives. A new protocol, resistant to these attacks, is proposed. Formal verification of the new proposal is presented. The successful verification gives confidence in the correctness of the proposed protocol.

The presence of the weaknesses, outlined in the case studies, highlights the inadequacy of using informal and intuitive techniques in security protocol design. Our analysis demonstrates that formal verification is imperative in the efficient design of security protocols.

## 2. Formal verification

Formal verification of cryptographic protocols is essential as it can detect design flaws that lead to protocol failure. The most popular techniques are state-space searching and formal logics. State-space searching involves exhaustive testing or scenario analysis. The protocol is specified and tests are carried out to determine whether undesirable states can be reached. However, state-space searching does not guarantee security from an active attacker and "drastic assumptions are required to keep the state-space to a practical size" [5]. Logical techniques involve a process of deductive reasoning, where the desired protocol goals are deduced by applying a set of axioms and inference rules to the assumptions and message exchanges of the protocols. These logic-based techniques have been shown to be effective in discovering protocol flaws.

### 2.1. Logic-based verification

The technique of logic-based formal verification is accredited largely to Burrow, Abadi and Needham, developers of the BAN logic [2]. This work initiated intense research in the area of logic-based formal verification. Several logics [6–8] have been developed on the basis of BAN. These logics can be used to generate concise proofs and have identified a number of flaws in protocols previously considered secure. They incorporate several improvements over BAN and are applicable to a wider range of protocols.

Logic-based formal verification involves the following steps:

1. Formalisation of the protocol messages;
2. Specification of the initial assumptions;
3. Specification of the protocol goals;
4. Application of the logical postulates.

The first step in logic-based verification involves specifying the protocol in the language of the logic by expressing each protocol message as a logical formula. This step is known as protocol formalisation (some authors also refer to it as idealisation). A formal description of the protocol, obtained by formalisation, does not simply list the components of each message but attempts to show the purpose of these components so as to avoid ambiguity.

The second step in the verification process involves formally specifying the initial protocol assumptions. These assumptions reflect the beliefs and possessions of the involved principals at the beginning of each protocol run.

In the third step, the desired protocol goals are expressed in the language of the logic. These goals are specified in terms of the beliefs and possessions of the protocol participants at the end of a successful protocol run.

The final verification step concerns the application of logical postulates to establish the beliefs and possessions of protocol principals. The objective of the logical analysis is to verify whether the desired goals of the protocol can be derived from the initial assumptions and protocol steps. If such a derivation exists, the protocol is successfully verified; otherwise, the verification fails.

A successfully verified protocol can be considered secure within the scope of the logic. On the other hand, even the results of a failed verification are helpful, as these may point to missing assumptions or weaknesses in the protocol. If a weakness is discovered, the protocol should be redesigned and re-verified.

However, verification logics have their limitations, not least of which is the likelihood of errors in protocol formalisation. Opportunities to make such mistakes abound as the verification process is complicated, requiring a thorough understanding of the used logic. During the verification process the semantics of the protocol must be interpreted, in order to specify the meaning that a protocol message is intended to convey. This 'interpretation process' is somewhat controversial—different authors may interpret the same messages differently. If the formalised protocol does not properly represent the original design, then the proof demonstrates only that the protocol corresponding to this formal description is secure. However, no claims can be made on the security of the original design.

Lack of clarity about the protocol goals and initial assumptions is a further cause for concern. In some cases the same protocol may be used for slightly different purposes. For example if a protocol is used to generate a new session key, each principal involved in the protocol run may require that the other principal believes the session key to be a shared secret. This property is known as second level belief. If a protocol is verified as secure for first level belief only and used in an application where second level belief is required, serious security breaches are likely. Hence, it is vital to note the assumptions and goals under which a security protocol is considered secure during its formal verification.

Despite these criticisms, logics have identified numerous protocol weaknesses and are considered a success. Gligor et al. [16] summarise the virtues of authentication logics as follows:

- They help formalise reasoning about useful abstract properties of cryptographic protocols.
- They force designers to make explicit security assumptions.
- They achieve a reasonably well-defined set of authentication goals.

The importance of formal verification cannot be overstated because it aids the early discovery of design flaws, thereby helping to prevent serious security breaches. The requirement for well-defined protocol assumptions and goals is valuable because it results in a clear definition of the protocol scope. In addition, the verification process often identifies protocol assumptions that would be otherwise overlooked.

## 3. The BCY protocol and its verification

In this section both the BCY protocol and the GNY logic are introduced. The logic is then used to formally verify the protocol and several weaknesses of the protocol are identified. The notation used throughout the paper to describe protocols is shown in Table 1. Although this format differs slightly from the published protocol descriptions, it conveys the same semantics.

### 3.1. The BCY protocol

The BCY protocol [13], as shown in Fig. 1, is aimed at providing authentication and key agreement in low-power portable devices such as mobile phones. It demonstrates the computational feasibility of using public-key cryptography in mobile communications. The protocol combines a Diffie–Hellman key exchange [17] with the modular square root (MSR) encryption technique [18]. The MSR technique reduces the computational burden placed on the mobile device. Certificates, denoted by $\{X_1, \ldots, X_n\}Ks-$, contain the components $X_1$ to $X_n$ as well as a hash of these components signed by a certification authority. As the service provider uses two public keys, one for Diffie–Hellman key agreement and one for MSR encryption, these will be termed *Kvd+* and *Kvm+* respectively.

Table 1
BCY protocol notation

| | |
|---|---|
| *U* | An identifier of the user |
| *V* | An identifier of the service provider |
| *S* | An identifier of the certification authority |
| *Kx+* | Public key of *X* |
| *Kx−* | Private key of *X* |
| *KK* | A key-encrypting key |
| *SK* | A session key |
| $r_X$ | A random nonce generated by *X* |
| *TS* | An expiration time |

BCY1: $V \rightarrow U$: {V, Kvd+,Kvm+}Ks-

U computes Y = {$r_U$}Kvm+, KK={Kvd+}Ku-, SK = {$r_U$}KK

BCY2: $U \rightarrow V$: Y, {{U,Ku+}Ks-}$r_U$

V computes $r_U$ ={Y}Kvm-, KK={Ku+}Kvd-, SK = {$r_U$}KK

BCY3: $V \rightarrow U$: {dataV}SK

BCY4: $U \rightarrow V$: {dataU}SK

Fig. 1. The original BCY protocol.

BCY1: The service provider initiates the BCY protocol by sending its identity, $V$, to the user, along with its two public keys, $Kvd+$ and $Kvm+$, all signed by a certification authority. Next, the user generates a nonce, $r_U$, which it encrypts with the MSR public key of the service provider, $Kvm+$, to form the message component $Y$. Further, $U$ computes the key-encrypting key, $KK$, using the Diffie–Hellman technique and generates the session key, $SK$, by encrypting $r_U$ under $KK$.

BCY2: The user's certificate, encrypted under $r_U$, is sent to the service provider along with the parameter $Y$. The service provider recovers $r_U$ using his private key and can thus decipher the rest of the message. $V$ also computes the key-encrypting key, $KK$, using the Diffie–Hellman technique and uses this key to generate the session key.

BCY3 and BCY4: The final protocol steps are intended to demonstrate possession of the session key by both principals. The exchange of recognisable data, encrypted under a fresh session key, completes authentication. This data will be referred to as identification data for the remainder of this paper.

### 3.2. The GNY logic

The GNY logic [6] is used to reason about cryptographic protocols. GNY is a direct successor of the BAN logic and is quite powerful in its ability to uncover even subtle protocol flaws. Discussions of the virtues and limitations of the logic can be found in [19,20].

In GNY, message extensions are added to the protocol description during protocol formalisation, so that principals can communicate their beliefs and thus reason about each other's beliefs. The use of message extensions enables the logic to deal with different levels of trust between protocol principals. As such, it is considered an improvement over BAN, which assumes that all principals are honest and competent. This development is noteworthy as many protocol attacks are performed by dishonest principals. As an example of a message extension, consider the following: $P \rightarrow Q : \{K, P\}Ks-$ is formally stated as $Q \lhd {}^*\{{}^*K, {}^*P\}Ks- \sim> S \mid\equiv P \overset{K}{\leftrightarrow} Q$. This means that principal $Q$ is told a session key, $K$, and an identity, $P$, encrypted under the private key of principal $S$. Each component is marked with a not-originated-here asterisk. Also, $Q$ is told that $S$ believes $K$ is a suitable shared secret for $P$ and $Q$.

The postulates of the GNY logic are used to deduce whether the protocol goals can be derived from the initial assumptions and protocol steps. If such a derivation exists, the protocol is successfully verified. A summary of the notation and the postulates, relevant to this work, is given in Appendix A.

### 3.3. Verification of the BCY protocol

The process of formal verification, described in Section 2, involves the following steps:

1. Formalisation of the protocol steps;
2. Specification of the initial assumptions;
3. Specification of the protocol goals;
4. Application of the logical postulates.

The verification process is now demonstrated by applying the GNY logic to the BCY protocol.

#### 3.3.1. Formalisation of the protocol steps

A formalised version of the protocol is shown in Fig. 2. The asterisks denote the ability of each principal to recognise that it did not send the received message at an earlier stage in the protocol. The message extensions in the first and second messages indicates that if $S$ signs a certificate, then it believes that the public key contained in that certificate belongs to the principal, whose name is included in the certificate.

BCY1’: $U \lhd *\{V,Kvd+,Kvm+\}Ks- \sim> S \mid\equiv \xrightarrow{Kvm+} V, S \mid\equiv \xrightarrow{Kvd+} V$

BCY2’: $V \lhd *\{*r_U\}Kvm+, *\{*\{U,Ku+\}Ks- \sim> S \mid\equiv \xrightarrow{Ku+} U\ \}r_U$

BCY3’: $U \lhd *\{dataV\}SK$

BCY4’: $V \lhd *\{dataU\}SK$

Fig. 2. The formalised BCY protocol.

*3.3.2. Specification of the initial assumptions*

The initial assumptions for the original BCY protocol are as follows:

| | | | |
|---|---|---|---|
| $U \ni Ku-;$ | $U \ni \{U,Ku+\}Ks-;$ | $U \ni dataU;$ | |
| $V \ni Kvd-;$ | $V \ni Kvm-;$ | $V \ni \{V,Kvd+,Kvm+\}Ks-;$ | $V \ni dataV;$ |
| $U \ni Ks+;$ | $V \ni Ks+;$ | $U \ni r_U;$ | $U \mid\equiv \#r_U;$ |
| $U \mid\equiv \phi(dataV);$ | $V \mid\equiv \phi(dataU);$ | $U \mid\equiv \phi(Kvd+,Kvm+)$ | $V \mid\equiv \phi(Ku+);$ |
| $U \mid\equiv \overset{Ks+}{\rightarrow} S;$ | $U \mid\equiv S \mid\Rightarrow S \mid\equiv *;$ | $U \mid\equiv (S \mid\Rightarrow \overset{Kvm+}{\rightarrow} V,\ S \mid\Rightarrow \overset{Kvd+}{\rightarrow} V);$ | |
| $V \mid\equiv \overset{Ks+}{\rightarrow} S;$ | $V \mid\equiv S \mid\Rightarrow S \mid\equiv *;$ | $V \mid\equiv S \mid\Rightarrow \overset{Ku+}{\rightarrow} S;$ | |

The first three rows state the possessions of both principals. Each principal possesses its private key(s), its certificate and its identification data. Further, both principals possess the public key of the certification authority. Also, *U* possesses a nonce which it believes to be fresh. The next row states the recognisability assumptions. Each principal recognises the other’s public keys and identification data. The final two rows concern beliefs regarding the certification authority. Both principals believe that *Ks*+ is the public key of *S*, that *S* is honest and competent, and that *S* has jurisdiction over the other principal’s public key(s).

*3.3.3. Specification of the protocol goals*

The goals of the BCY protocol are as follows:

| | | |
|---|---|---|
| $U \mid\equiv \overset{Kvd+}{\longrightarrow} V\ ;$ | $U \mid\equiv \overset{Kvm+}{\longrightarrow} V;$ | $V \mid\equiv \overset{Ku+}{\longrightarrow} U;$ |
| $U \ni SK;$ | $U \mid\equiv \#SK;$ | $U \mid\equiv U \overset{SK}{\longleftrightarrow} V;$ |
| $V \ni SK;$ | $V \mid\equiv \#SK;$ | $V \mid\equiv U \overset{SK}{\longleftrightarrow} V;$ |
| $U \mid\equiv V \mid\sim \{dataV\}SK;$ | $U \mid\equiv \#\{dataV\}SK;$ | |
| $V \mid\equiv U \mid\sim \{dataU\}SK;$ | $V \mid\equiv \#\{dataU\}SK;$ | |

The goals in the first row state that both principals can bind public keys to their owner. The next two rows describe key agreement: both principals should possess the session key and believe it to be fresh. Further, they should believe that the session key is a suitable shared secret. The goals on the remaining two rows concern authentication: each principal should believe that its counterpart conveyed the respective identification data and that these messages are fresh.

*3.3.4. Application of the logical postulates*

$BCY1' : \ U \lhd {}^*\{V,Kvd+,Kvm+\}Ks- \sim> S \mid\equiv \overset{Kvm+}{\longrightarrow} V, S \mid\equiv \overset{Kvd+}{\longrightarrow} V.$

- Applying T1 to BCY1′ yields $U \lhd \{V,Kvd+,Kvm+\}Ks-$. *U* is told *V*’s certificate without the not-originated-here asterisk.

- Appling T6 yields $U \triangleleft (V, Kvd+, Kvm+)$. $U$ is told the contents of $V$'s certificate.
- Applying T2 yields $U \triangleleft Kvd+$ and $U \triangleleft Kvm+$. $U$ is told both of $V$'s public keys.
- Applying P1 yields $U \ni Kvd+$ and $Kvm+$. The user possesses the public keys of the service provider.
- Applying P8, we obtain $U \ni KK = \{Kvd+\}Ku-$. $U$ possesses the key-encrypting key.
- Since $U$ possesses $r_U$, by applying P6, we obtain $U \ni SK = \{r_U\}KK$. $U$ possesses the session key.
- Since $U$ believes that $r_U$ is fresh, applying F1 gives $U | \equiv \#SK$. $U$ believes the session key is fresh.
- Since $U$ recognises $(Kvd+, Kvm+)$, by R1 $U | \equiv \phi(V, Kvd+, Kvm+)$. $U$ recognises the contents of $V$'s certificate.
- The prerequisites of I4 are satisfied. Thus, applying I4, we obtain $U | \equiv S | \sim (V, Kvd+, Kvm+)Ks-$ and $U | \equiv S | \sim (V, Kvd+, Kvm+)$. $U$ believes that $S$ conveyed $V$'s certificate and the contents of the certificate.
- However, $U$ cannot believe that the certificate is a valid current certificate of principal $V$. The preconditions of J2 are not achieved, as the certificate contains no expiration time. An intruder could use an old compromised certificate belonging to $V$ in order to masquerade as the service provider. Thus, $U$ cannot believe that $Kvd+$ and $Kvm+$ are $V$'s current public keys. As $Kvd+$ is used to generate the session key, $U$ cannot derive the belief that the session key, $SK$ is a shared secret with $V$.

$$\text{BCY2}': \quad V \triangleleft {}^*\{{}^*r_U\}Kvm+, {}^*\{{}^*\{U, Ku+\}Ks- \sim> S | \equiv \xrightarrow{Ku+} U\}r_U.$$

- Applying T1, T4 and P1 yields $V \ni r_U$. The service provider possesses the random number generated by the user.
- Applying T1 and T3, we obtain $V \triangleleft \{U, Ku+\}Ks-$. $V$ is told $U$'s certificate. The proof now continues along similar lines to BCY1′.
- Applying T6, T2 and P1, we obtain $V \ni Ku+$. $V$ possesses $U$'s public key.
- Applying P8, we obtain $V \ni KK = \{Ku+\}Kvd-$. $V$ possesses the key-encrypting key.
- Since $V$ possesses both $r_U$ and $KK$ by P6, we obtain $V \ni SK$. $V$ possesses the session key.
- However, $V$ cannot establish that the session key is fresh.
- Since $V$ recognises $Ku+$, by R1 $V | \equiv \phi(U, Ku+)$. $V$ recognises the contents of $U$'s certificate.
- The prerequisites of I4 are satisfied. Thus, applying I4, we obtain $V | \equiv S | \sim (U, Ku+)Ks-$ and $V | \equiv S | \sim (U, Ku+)$. $V$ believes that $S$ conveyed $U$'s certificate and the contents of the certificate.
- However, $V$ cannot believe that the received certificate is a valid current certificate of principal $U$ (for the same reasons that $U$ cannot believe in the validity of $V$'s certificate in BCY1′). Thus, $V$ cannot believe that $Ku+$ is the public key of $U$ or that the derived session key, $SK$, is a shared secret with $U$.

$$\text{BCY3}' \text{ and BCY4}': \quad U \triangleleft {}^*\{dataV\}SK, \quad V \triangleleft {}^*\{dataU\}SK.$$

- Application of F7 yields $U | \equiv \#\{dataV\}SK$. $U$ believes BCY3′ to be fresh.
- In contrast, $V$ cannot establish that BCY4′ is fresh because he does not believe that the session key is fresh.
- Further, the final message exchange fails to provide authentication. The prerequisites for I1 are not satisfied as (1) neither party believes that the session key is a shared secret and (2) $V$ does not believe that the session key is fresh. Thus, neither principal believes that the other conveyed its identification data.

### 3.4. Weaknesses in the BCY protocol

The above verification of the BCY protocol identifies the following failed goals:

1. $U$ cannot derive that $V$'s public keys are valid;
2. $V$ cannot derive that $U$'s public key is valid;
3. $U$ cannot derive that the session key is a suitable shared secret with $V$;
4. $V$ cannot derive that the session key is fresh;
5. $V$ cannot derive that the session key is a suitable shared secret with $U$;
6. $U$ cannot derive that $V$ conveyed *dataV*;
7. $V$ cannot derive that $U$ conveyed *dataU*;
8. $V$ cannot derive that BCY4′ is fresh.

These failures stem essentially from two weaknesses of the protocol:

W1. The principals cannot validate the certificates (Failures 1,2,3,5,6,7).
W2. $V$ cannot establish that the session key is fresh (Failures 4,8).

Due to these weaknesses the original BCY protocol provides neither authentication nor key agreement. Some of the security concerns associated with W1 have been identified by Carlsen [14] and other concerns have been raised by Mu and Varadharajan [15]. Carlsen proposed protocol modifications to address W1. Subsequently, Mu and Varadharajan criticised the Carlsen proposal and suggested further amendments. Carlsen also identified W2 and his proposal attempts to address this weakness.

## 4. Analysis of Carlsen BCY and Mu–Varadharajan BCY protocols

In this section, modifications to the BCY protocol by Carlsen and Mu and Varadharajan are presented. These protocols are formally verified. Our analyses demonstrate that these modifications do not adequately address the problems that they are purported to solve. Further, new security concerns arise due to the proposed modifications.

### 4.1. Verification of the Carlsen BCY protocol

Carlsen identified weaknesses in the original protocol relating to the validity of the public keys of the protocol principals. He also pointed out problems concerning the freshness of the second protocol message. In order to address these concerns, he suggested that (1) an expiration time, $TS_V$, be included in $V$'s certificate and (2) a random number, $r_V$, be included in the first and second protocol messages. However, Carlsen omitted $V$'s identity from the certificate. Thus, $V$'s certificate was changed from $\{V, Kvd+, Kvm+\}Ks-$ in the original BCY protocol to $\{Kvd+, Kvm+, TS_V\}Ks-$. The steps of the Carlsen BCY protocol are shown in Fig. 3.

#### 4.1.1. Formalisation of the protocol steps

The formalised version of the Carlsen BCY protocol is shown in Fig. 4. The message extensions to $V$'s certificate are not included because the certificate does not contain $V$'s identity. Hence, the certificate cannot bind $V$'s public keys to their owner.

C1: $V \rightarrow U$: $r_V$, {Kvd+,Kvm+,$TS_V$}Ks-

U computes Y = {$r_U$}Kvm+, KK={Kvd+}Ku-, SK = {$r_U$}KK

C2: $U \rightarrow V$: Y, {$r_V$,{U,Ku+}Ks-}$r_U$

V computes $r_U$ ={Y}Kvm-, KK={Ku+}Kvd-, SK = {$r_U$}KK

C3: $V \rightarrow U$: {dataV}SK

C4: $U \rightarrow V$: {dataU}SK

Fig. 3. The Carlsen BCY protocol.

C1’: $U \triangleleft * r_V ,*${Kvd+,Kvm+,$TS_V$}Ks-

C2’: $V \triangleleft *\{r_U\}Kvm+, *\{r_V, \{U,Ku+\}Ks- \sim> S \mid\equiv \xrightarrow{Ku+} U \}r_U$

C3’: $U \triangleleft *$ {dataV}SK

C4’: $V \triangleleft *$ {dataU}SK

Fig. 4. The formal description of the proposed protocol.

### *4.1.2. Specification of the initial assumptions*

The initial assumptions include those for the BCY protocol described in Section 3.3.1. The following additional assumptions are also required:

$V \ni r_V; \quad V \mid\equiv \#r_V.$

This states that $V$ possesses a random number $r_V$ and believes it to be fresh. It is further assumed that principals can determine whether a certificate is current by checking the included expiration time. Therefore, certificates which contain an expiration time can be treated as fresh for the purpose of establishing beliefs about public keys.

### *4.1.3. Specification of the protocol goals*

The protocol goals remain unchanged from the original BCY version.

### *4.1.4. Application of the logical postulates*

$\mathrm{C1}': \ U \triangleleft {}^*r_V, {}^*\{Kvd+, Kvm+, TS_V\}Ks-.$

- Despite the changes made by Carlsen, many of weaknesses the original BCY protocol described in Section 3 remain unresolved. Although CertV now contains an expiration time, $U$ cannot believe the keys $Kvd+$ and $Kvm+$ belong to $V$ because $V$’s identity is not included in the certificate. As a result, the prerequisites for J2 are not satisfied. $U$ cannot believe that the session key is a shared secret between $U$ and $V$.

$\mathrm{C2}': \ V \triangleleft {}^*\{r_U\}Kvm+, {}^*\{r_V, \{U, Ku+\}Ks- \sim> S \mid\equiv \xrightarrow{Ku+} U\}r_U.$

- The presence of $r_V$ in messages C2′ enables $V$ to establish the freshness of this message according to F1. This addresses Carlsen's concerns about the freshness of the second protocol message.
- Otherwise the verification of C2′ remains unchanged from the original version of the protocol (BCY2′). Thus, $V$ cannot confirm the validity of $U$'s certificate. Therefore, $V$ does not believe that $Ku+$ is the public key of $U$ or that the derived session key, $SK$, is a shared secret.

C3′ and C4′ : $U \triangleleft ^* \{dataV\}SK,\ V \triangleleft ^* \{dataU\}SK.$

- As with the original version of the protocol, the final message exchange does not provide authentication of either party, because neither can believe that the session key is a shared secret.

In summary, Carlsen's version of the BCY protocol addresses some limitations of the original protocol, but fails to address others. The introduction of $r_V$ prevents attacks based on a replay of the second protocol message and the inclusion of an expiration time in $V$'s certificate allows the user to confirm that this certificate is current. However, Carlsen's protocol fails to provide key agreement or authentication. Firstly, the introduction of $r_V$ does not enable the service provider to establish freshness of the session key. Secondly, Carlsen's proposal suffers from several new problems. Due to the removal of $V$'s identity from its certificate, the user cannot bind the contained public key to $V$. Further, the decision to leave $U$'s certificate unchanged is questionable, as it suffers from the same weakness as has been noted on $V$'s certificate. Carlsen's reason for this decision is unknown. In conclusion, limited improvement over the original protocol is achieved.

### 4.2. Verification of the Mu–Varadharajan BCY protocol

Mu and Varadharajan [15] suggested further modifications, as shown in Fig. 5, to overcome some problems associated with Carlsen's BCY protocol. Firstly, they reintroduced $V$'s identity into the certificate. Secondly, an expiration time $TS_U$ is introduced into $U$'s certificate to avoid attacks where an intruder replays a compromised certificate. Thus, the certificates were changed to $\{V, Kvd+, Kvm+, TS_V\}Ks-$ and $\{U, Ku+, TS_U\}Ks-$. Finally, Mu and Varadharajan omitted the final message exchange from the protocol.

#### 4.2.1. Formalisation of the protocol steps

The formalised version of the protocol is shown in Fig. 6.

#### 4.2.2. Specification of the initial assumptions

The initial assumptions for the Mu–Varadharajan BCY protocol are the same as those associated with the Carlsen BCY protocol.

MV1: $V \rightarrow U$: $r_V$, {V,Kvd+,Kvm+,$TS_V$}Ks-

U computes Y = {$r_U$}Kvm+, KK={Kvd+}Ku-, SK = {$r_U$}KK

MV2: $U \rightarrow V$: Y, {$r_V$,{U,Ku+,$TS_U$}Ks-}$r_U$

V computes $r_U$ ={Y}Kvm-, KK={Ku+}Kvd-, SK = {$r_U$}KK

Fig. 5. The Mu–Varadharajan BCY protocol.

$$\begin{aligned}
&\text{MV1'}: \quad U \lhd {}^{*} r_V, {}^{*}\{V, Kvd+, Kvm+, TS_V\}Ks- \\
&\qquad \sim> S \mid\equiv \xrightarrow{Kvd+} V,\ S \mid\equiv \xrightarrow{Kvm+} V \\
&\text{MV2'}: \quad V \lhd {}^{*}\{r_U\}Kvm+, {}^{*}\{r_V, \{U, Ku+, TS_U\}Ks- \sim> S \mid\equiv \xrightarrow{Ku+} U\}r_U
\end{aligned}$$

Fig. 6. The formal description of the Mu–Varadharajan BCY protocol.

### *4.2.3. Specification of the protocol goals*

The protocol goals remain unchanged from the original BCY protocol.

### *4.2.4. Application of the logical postulates*

$$\text{MV1}': \quad U \lhd {}^{*}r_V, {}^{*}\{V, Kvd+, Kvm+, TS_V\}Ks- \sim> S \mid\equiv \xrightarrow{Kvd+} V, S \mid\equiv \xrightarrow{Kvm+} V.$$

- Using identical reasoning to the original protocol we derive that $U$ possesses the session key and believes it to be fresh.
- Applying axiom J2, we obtain $U \mid\equiv S \mid\equiv \xrightarrow{Kvd+} V$ and $U \mid\equiv S \mid\equiv \xrightarrow{Kvm+} V$. $U$ believes that $S$ believes that $Kvd+$ and $Kvm+$ are $V$'s public keys.
- Applying J1 yields, $U \mid\equiv \xrightarrow{Kvd+} V$ and $U \mid\equiv \xrightarrow{Kvm+} V$. $U$ believes that $Kvd+$ and $Kvm+$ are $V$'s public keys.

$$\text{MV2}': \quad V \lhd {}^{*}\{r_U\}Kvm+, {}^{*}\{r_V, \{U, Ku+, TS_U\}Ks- \sim> S \mid\equiv \xrightarrow{Ku+} U\}r_U.$$

- Similar reasoning to that used in MV1$'$ yields $V \mid\equiv \xrightarrow{Ku+} U$. $V$ believes that $Ku+$ is $U$'s public key.
- However, $V$ does not believe in the freshness of the session key because it does not contain any component which $V$ believes to be fresh. Thus, $V$ cannot detect an attack that results in the reuse of an old session key.

Mu and Varadharajan claim that the protocol provides authentication by using certificates which can bind a principal's identity to a public key. However, Horn et al. [21] refute this claim, stating that a final message exchange, similar to that in the BCY protocol, is required to provide authentication. Our analysis supports the claim that no authentication is achieved.

## 5. A new attack on the BCY protocols

Our analysis of the BCY protocol, described in Section 3.3, revealed that the service provider cannot tell whether the session key is fresh. Despite modifications by Carlsen, intended to address this limitation, an attack can be carried out against the Carlsen BCY and the Mu–Varadharajan BCY protocols. In this attack an old session key is reused to enable the impersonation of a legitimate user.

In order to perform such an attack, an adversary must possess an old session key and the corresponding nonce generated by the user. The availability of such information to an adversary was envisaged by the authors of the BCY protocol. They refer to an insider as "someone who obtains information by theft, conspiracy or computer system intrusion or any method other than scanning the radio link." They claim that the BCY protocol is secure against a "one-time insider who can get information at some time and use it at some later time". Thus, the assumed capabilities of our attacker are reasonable.

A1: V → A(U): $r_V$, {V,Kvd+,Kvm+, $TS_V$}Ks-

A(U) computes Y = {$r_{Uo}$}Kvm+

A2: A(U) → V: Y, {$r_V$,{U,Ku+,$TS_U$}Ks-}$r_{Uo}$

V computes $r_U$ ={Y}Kvm-, KK={Ku+}Kvd-, SK = {$r_{Uo}$}KK

Fig. 7. A new attack on the Mu–Varadharajan BCY protocol.

The attack can be carried out as follows: If the attacker, $A$, is able to obtain an old value of the user's nonce, $r_{U_o}$, $A$ can replay it to the service provider $V$. This results in the inadvertent re-computation of an old session key, $SK_O$, by $V$. If $A$ also possesses the session key, $SK_O$, corresponding to $r_{U_o}$, $A$ can use this key to impersonate a legitimate user of the system. This is possible, because $V$ cannot determine freshness of the session key. Authentication and key agreement would only be achieved, if both parties were able to determine that the session key is fresh.

The message flow of this attack, mounted against the Mu–Varadharajan BCY protocol, is shown in Fig. 7. A similar attack is possible against Carlsen's version of the protocol.

A1: The service provider $V$ initiates the protocol run as normal. However, the message, intended for user $U$, is intercepted by the attacker $A$. Next, $A$ chooses the nonce $r_U$ to be equal to the old random value $r_{U_o}$ (which is known to $A$) and calculates $Y$ as the old random value $r_{U_o}$ encrypted under $V$'s MSR public key. This will cause an old session key to be used, as the session key is essentially dependent only on $r_U$.

A2: The attacker sends $r_V$ and $U$'s certificate, both encrypted under $r_{U_o}$, along with $Y$ to $V$. $V$ decrypts $Y$ with its MSR private key and thus retrieves the value $r_{U_o}$. $V$ is not able to detect that $r_{U_o}$ is a replayed value from a previous protocol run and hence calculates the session key $SK$ as $\{r_{U_o}\}KK$. This session key $SK$ has the same value as the compromised key $SK_O$, as both are based on the same random value, $r_{U_o}$. The attack is now complete. The service provider believes that the session key $SK$ is shared with the user $U$, while in fact it is shared with the attacker. According to the protocol run, $V$ believes (incorrectly) that a message encrypted with $SK$ ($V$ is not aware of the fact that $SK = SK_O$) must have been sent by $U$. However, since the attacker also knows $SK(= SK_O)$, the attacker encrypt messages with $SK$, thus impersonating the user $U$ in communication with the service provider $V$.

## 6. Proposed modification to the BCY protocol

The verification of the Mu–Varadharajan BCY protocol, as shown in Section 4.2, exposes the following weaknesses:

1. The service provider $V$ cannot derive freshness of the session key $SK$.
2. Consequently, $V$ cannot derive suitability of $SK$ as shared secret with user $U$.
3. Authentication fails, due to lack of exchange of standard messages encrypted with the session key.

In light of these failures, we propose that a number of amendments are made to the Mu–Varadharajan BCY protocol. The new protocol, which is outlined in Fig. 8 and referred to as CDF–BCY protocol, incorporates the following amendments:

- Firstly, we include $V$'s nonce, $r_V$, in the session key $SK$, i.e. $SK$ becomes $\{r_U, r_V\}KK$. As both parties now contribute towards the session key, each can confirm its freshness. In particular, $V$ is now able to confirm

CDF1: $V \rightarrow U$: $r_V$, {V,Kvd+,Kvm+,$TS_V$}Ks-

U computes Y = {$r_U$}Kvm+, KK={Kvd+}Ku-, SK = {$r_U$,$r_V$}KK

CDF2: $U \rightarrow V$: Y, {$r_V$,{U,Ku+,$TS_U$}Ks-}$r_U$

V computes $r_U$ ={Y}Kvm-, KK={Ku+}Kvd-, SK = {$r_U$,$r_V$}KK

CDF3: $V \rightarrow U$: {dataV}SK

CDF4: $U \rightarrow V$: {dataU}SK

Fig. 8. Proposed CDF–BCY protocol.

freshness of *SK* and, consequently, is able to derive that *SK* is a suitable secret shared with user *U*. Hence, weaknesses 1 and 2 above are removed.

- Secondly, we reintroduce the final message exchange CDF3/CDF4, which were omitted in [15]. These final protocol steps demonstrate possession of the session key by both principals. The exchange of recognisable data, encrypted under a fresh session key, completes authentication.

## 6.1. Verification of the proposed CDF–BCY protocol

In this section, the formal verification of the CDF–BCY protocol is outlined. The protocol is successfully verified, giving confidence in the correctness of the new protocol.

### 6.1.1. Formalisation of the protocol steps

A formal description of the proposed protocol is shown in Fig. 9.

### 6.1.2. Specification of the initial assumptions

The assumptions required for the proposed protocol are the same as those used for the Carlsen and Mu–Varadharajan versions of the protocol.

### 6.1.3. Specification of the protocol goals

The protocol goals remain unchanged from the original BCY version.

### 6.1.4. Application of the logical postulates

$$\text{CDF1}':\quad U \lhd {}^*r_V, {}^*\{V, Kvd+, Kvm+, TS_V\}Ks- \sim> S \mid\equiv \xrightarrow{Kvm+} V, S \mid\equiv \xrightarrow{Kvd+} V.$$

CDF1’: $U \lhd {}^* r_V, {}^*\{V,Kvd+,Kvm+,TS_V\}Ks-$

$\sim> S \mid\equiv \xrightarrow{Kvm+} V,\ S \mid\equiv \xrightarrow{Kvd+} V$

CDF2’: $V \lhd {}^*\{r_U\}Kvm+, {}^*\{r_V, \{U,Ku+,TS_U\}Ks- \sim> S \mid\equiv \xrightarrow{Ku+} U\ \}r_U$

CDF3’: $U \lhd {}^*\{dataV\}SK$

CDF4’: $V \lhd {}^*\{dataU\}SK$

Fig. 9. The formal description of the CDF–BCY protocol.

- The initial protocol message is unchanged from Mu and Varadharajan's version of the protocol. Thus, using identical reasoning we obtain $U \ni SK$, $U \mid\equiv \#SK$, $U \mid\equiv \xrightarrow{Kvd+} V$ and $U \mid\equiv \xrightarrow{Kvm+} V$. That is, the user possesses the session key, believes it to be fresh and believes that $Kvd+$ and $Kvm+$ are $V$'s current public keys.

$$\text{CDF2}': \quad V \lhd {}^*\{r_U\}Kvm+, {}^*\{r_V, \{U, Ku+, TS_U\}Ks- \sim> S \mid\equiv \xrightarrow{Ku+} U\}r_U.$$

- Similar reasoning to that used above allows us to derive that the service provider possesses the public key of the user and the session key. $V$ also believes that $Ku+$ is the public key of $U$.
- By application of $F1$, $V \mid\equiv \#SK$. As $r_V$ is included in the session key, $V$ believes that the session key is fresh.

$$\text{CDF3}' \text{ and CDF4}': \quad U \lhd {}^*\{dataV\}SK, \;\; V \lhd {}^*\{dataU\}SK.$$

- The session key is generated using the Diffie–Hellman technique, which is based on the properties of the underlying cryptosystem. However, the GNY logic is cryptosystem-independent and therefore cannot verify that the derived session key is a shared secret.
- To continue the proof, the following is assumed: The key-encrypting key is established by a Diffie–Hellman key exchange and therefore, can be considered a shared secret between the intended principals. Only these principals can create the session key, as knowledge of the key-encrypting key is an essential prerequisite. Hence we can assume $U \mid\equiv U \overset{SK}{\leftrightarrow} V$ and $V \mid\equiv U \overset{SK}{\leftrightarrow} V$. That it, both principals believe that $SK$ is a suitable shared secret and we continue with the proof.
- Applying T1 and I1 to CDF3′ and CDF4′ yields $V \mid\equiv U \mid\sim dataU$ and $U \mid\equiv V \mid\sim dataV$. The final message exchange using the shared session key provides authentication.

The above verification demonstrates that the protocol achieves its goals.

## 7. Concluding remarks

In this paper, the process of formal verification of cryptographic security protocols using modal logics was discussed. This verification process was demonstrated by applying a logic to the BCY protocol. All known flaws in the original BCY protocol were identified by the verification process, thus showing that the protocol fails to achieve authentication or key agreement. This highlights the importance of using formal techniques to increase confidence in protocol security.

Published modifications to the BCY protocol were also verified in this paper. In addition to discovering known flaws, the verification showed that all versions suffer from a previously undiscovered failure. An attack, which exploits this failure, has been presented. This attack, based on the reuse of old session keys, allows an adversary to impersonate a legitimate principal. Finally, a new protocol, resistant to this attack, is proposed and formally verified, giving confidence in the correctness of the protocol.

Despite the merits of formal verification described above, we note that such techniques are not without criticism. Arguably the most complex part of verification is the generation of the formalised protocol from the informal description. Formalising the protocol steps is especially difficult when using logics, as the intended message meaning must be made explicit in the formalised description. However, mistakes while formalising the protocol steps can render the verification worthless. Opportunities to make such mistakes abound as the verification process is complicated, requiring a thorough understanding of the used logic. Lack of clarity about the protocol goals and initial assumptions is a further cause for concern, as these essentially specify what is achieved by the protocol and under what circumstances it is applicable.

The success of formal verification in discovering protocol flaws shows that it offers significant advantages over informal and intuitive methods. The results of the case studies presented in this paper, demonstrate that formal verification is an imperative step in the design of security protocols.

## Appendix A. GNY notation and logical postulates

In this appendix we list the notation and the logical postulates of the GNY logic that are used throughout this paper.

| | |
|---|---|
| $(X, Y)$ | Concatenation of two formulae |
| $\{X\}K$ and $\{X\}K^{-1}$ | Symmetric encryption and decryption |
| $\{X\}K+$ | Public key encryption/decryption |
| $\{X\}K-$ | Private key encryption/decryption |
| $\#(X)$ | The formula $X$ is fresh. $X$ has not been sent in a message at any time before the current run of the protocol |
| $\phi(X)$ | Formula $X$ is recognisable |
| $P \triangleleft X$ | $P$ is told $X$. $P$ has a received a message containing $X$ and $P$ can read and repeat $X$, possibly after performing some decryption |
| $P \triangleleft {}^{*}(X)$ | $P$ is told formula $X$ which he did not convey previously during the current protocol run |
| $P \ni X$ | $P$ possesses or is capable of possessing formula $X$ |
| $P \mid\sim X$ | $P$ conveyed $X$ |
| $P \mid\equiv X$ | $P$ believes $X$. That is, the principal $P$ acts as if $X$ is true |
| $X \sim> C$ | Formula $X$ has the extension $C$. The precondition for $X$ being conveyed is represented by statement $C$ |
| $P \mid\Rightarrow X$ | $P$ has jurisdiction over $X$. The principal $P$ is an authority on $X$ and should be trusted on this matter. This construct is used when a principal has delegated authority over some statement |
| $P \overset{K}{\leftrightarrow} Q$ | $K$ is a suitable secret for $P$ and $Q$. They may use it as a key to communicate or as a proof of identity |

$$\text{T1}: \ \frac{P \triangleleft {}^{*}X}{P \triangleleft X}.$$

If a principal is told a formula marked with a not-originated-here asterisk, then the principal is told that formula.

$$\text{T3}: \ \frac{P \triangleleft \{X\}K, P \ni K}{P \triangleleft X}.$$

If a principal is told a formula encrypted with a key he possesses, then he is considered to have been told the decrypted contents of that formula.

$$\text{T4}: \ \frac{P \triangleleft \{X\}K+, P \ni K-}{P \triangleleft X}.$$

If a principal is told a formula encrypted with a public key and he possesses the corresponding private key then he is considered to have also been told the decrypted contents of that formula.

$$\text{T6}: \frac{P \lhd \{X\}K-, P \ni K+}{P \lhd X}.$$

If a principal is told a formula encrypted with a private key and he possesses the corresponding public key then he is considered to have also been told the decrypted contents of that formula.

$$\text{P1}: \frac{P \lhd X}{P \ni X}.$$

A principal is capable of possessing anything which he is told.

$$\text{P2}: \frac{P \ni X, P \ni Y}{P \ni (X,Y), P \ni F(X,Y)}.$$

If a principal possesses two formulae, then he is capable of possessing the concatenation of these formulae as well as any computationally feasible function, $F$, of them.

$$\text{P6}: \frac{P \ni K, P \ni X}{P \ni \{X\}K, P \ni \{X\}K^{-1}}.$$

If a principal possesses a formulae and a key then he is capable of possessing the encryption and decryption of that formula with this key.

$$\text{P8}: \frac{P \ni K-, P \ni X}{P \ni \{X\}K-}.$$

If a principal possesses a formulae and a private key then he is capable of possessing the decryption of that formula with this key.

$$\text{F1}: \frac{P|\equiv \#(X)}{P|\equiv \#(X,Y), P|\equiv \#(F(X)).}$$

If a principal believes that a formula $X$ is fresh, then he believes that any formula of which $X$ is a component is fresh and that a computationally feasible one-to-one function, $F$, of $X$ is fresh.

$$\text{F7}: \frac{P|\equiv \phi(X), P|\equiv \#(K), P \ni K}{P|\equiv \#(\{X\}K), P|\equiv \#(\{X\}K^{-1}).}$$

If a principal believes that a formula $X$ is recognisable, and $X$ possesses a key $K$, which he believes to be fresh, then he believes that the encryption and decryption of $X$ with the key $K$ is also fresh.

$$\text{R1}: \frac{P|\equiv \phi(X)}{P|\equiv \phi(X,Y), P|\equiv \phi(F(X)).}$$

If a principal believes that a formula $X$ is recognisable, then he believes that any formula of which $X$ is a component is recognisable and that a computationally feasible one-to-one function, $F$, of $X$ is recognisable.

$$\text{I1}: \frac{P \lhd {}^*\{X\}K, P \ni K, P|\equiv P\overset{K}{\leftrightarrow}Q, P|\equiv \phi(X), P|\equiv \#(X,K)}{P|\equiv Q| \sim X, P|\equiv Q| \sim \{X\}K, P|\equiv Q \ni K}.$$

If, for principal $P$, the following conditions hold: $P$ receives a formula $X$, encrypted under key $K$ and marked with a not-originated-here asterisk, $P$ possesses $K$ and believes that it is a suitable shared secret with $Q$, $P$ believes that the formula $X$ is recognisable and that either $X$ or $K$ is fresh. Then, $P$ believes that $Q$ once conveyed the message $X$, that $Q$ once conveyed the message $X$ encrypted under $K$ and that $Q$ possesses $K$.

$$\text{I4}: \ \frac{P \triangleleft \{X\}K-, P \ni K+, P|\equiv \overset{K+}{\rightarrow} Q, P|\equiv \phi(X), P|\equiv \#(X, K+)}{P|\equiv Q|\sim X, P|\equiv Q|\sim \{X\}K-.}$$

If, for principal $P$, the following conditions hold: $P$ receives a formula $X$, encrypted under private key, $K-$, $P$ possesses the corresponding public key, $K+$ and believes the public key belongs to $Q$, $P$ believes that the formula $X$ is recognisable and that either $X$ or $K+$ is fresh. Then, $P$ believes that $Q$ once conveyed the message $X$, that $Q$ once conveyed the message $X$ encrypted under $Q$'s private key, $K-$.

$$\text{J1}: \ \frac{P|\equiv Q|\Rightarrow C, P|\equiv Q|\equiv C}{P|\equiv C}.$$

If principal $P$ believes that $Q$ has authority over some statement $C$ and that $Q$ believes $C$, then $P$ should also believe $C$.

$$\text{J2}: \ \frac{P|\equiv Q|\Rightarrow Q|\equiv^{*}, P|\equiv Q|\sim (X \sim> C), P|\equiv \#(X)}{P|\equiv Q|\equiv C}.$$

If principal $P$ believes that $Q$ is honest and competent and $P$ receives a fresh message $X$ with the extension $C$, which he believes $Q$ conveyed, then $P$ believes that $Q$ believes $C$.

## References

[1] R.M. Needham, M.D. Schroeder, Using encryption for authentication in large networks of computers, Communications of the ACM 21 (12) (1978) 993–999.
[2] M. Burrows, M. Abadi, R. Needham, A logic of authentication, ACM Operating System Review 23 (5) (1989) 1–13.
[3] C. Meadows, The NRL protocol analyzer: an overview, Journal of Logic Programming 26 (1996) 113–131.
[4] J.K. Millen, S.C. Clark, S.B. Freedman, The interrogator: protocol security analysis, IEEE Transactions on Software Engineering SE-13 (2) (1997) 274–288.
[5] L.C. Paulson, The inductive approach to verifying cryptographic protocols, Journal of Computer Security 6 (2000) 85–128.
[6] L. Gong, R. Needham, R. Yahalom, Reasoning about belief in cryptographic protocols, in: Proceedings of 1990 IEEE Computer Society Synopsis on Research in Security and Privacy, 1990, pp. 234–248.
[7] T. Coffey, P. Saidha, A logic for verifying public key cryptographic protocols, IEE Journal of Proceedings—Computers and Digital Techniques 144 (1) (1997) 28–32.
[8] Y. Zhang, V. Varadharajan, A logic for modelling the dynamics of beliefs in cryptographic protocols, in: Proceedings of Australasian Computer Science Conference, 2001.
[9] Security Architecture, 3G TSPP 33.102 version 3.7.0.
[10] D.S. Wong, A.H. Chan, Efficient and mutually authenticated key exchange for low power computing devices, in: Proceedings of ASIACRYPT 2001.
[11] G. Horn, B. Preneel, Authentication and payment in future mobile systems, Journal of Computer Security 8 (2000) 183–207.
[12] C. Boyd, D.-G. Park, Public key protocols for wireless communications, in: Proceedings of 1998 International Conference on Information Security and Cryptology (ICISC '98), Seoul, 1998.
[13] M.J. Beller, L.-F. Chang, Y. Yacobi, Privacy and authentication on a portable communications system, IEEE Journal on Selected Areas in Communications 11 (6) (1993) 821–829.
[14] U. Carlsen, Optimal privacy and authentication on a portable communications system, ACM Operating Systems Review 28 (3) (1994) 16–23.
[15] Y. Mu, V. Varadharajan, On the design of security protocols for mobile communications, in: Information Security and Privacy, Lecture Notes in Computer Science, vol. 1172, Springer, Berlin, 1996, pp. 134–145.
[16] V.D. Gligor, R. Kailar, S. Stubblebine, L. Gong, Logics for cryptographic protocols—virtues and limitations, in: Proceedings of 4th Computer Security Foundation Workshop, 1991, pp. 219–226.
[17] W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory 22 (1976) 644–654.
[18] M.O. Rabin, Digitalized signatures and public-key functions as intractable as factorisation, Technical Report MIT/LCS/TR-212, MIT, 1979.
[19] L. Gong, Handling infeasible specifications of cryptographic protocols, in: Proceedings of 4th Computer Security Foundation Workshop, 1991, pp. 99–102.

[20] A. Mathuria, R. Safavi-Naini, P. Nickolas, Some remarks on the logic of Gong, Needham and Yahalom, in: Proceedings of the International Computer Symposium, Hsinchu, Taiwan, ROC, vol. 1, 1994, pp. 303–308.

[21] G. Horn, K. Martin, C. Mitchell, Authentication protocols for mobile network environment value-added services, IEEE Transactions on Vehicular Technology 51 (2002) 383–392.

**Tom Coffey** is Professor of Electronic and Computer Engineering at the University of Limerick, Ireland. He is director of the Data Communication Security Laboratory at the University of Limerick. He is a chartered engineer; he holds masters and doctoral degrees from City University (London) and University of Ulster (Ireland) respectively. His research interests include protocol engineering, encryption algorithms, network security, modal logics, verification of security protocols and power-line communication systems.

**Reiner Dojen** is employed as a Junior Lecturer at the Department of Electronic and Computer Engineering at the University of Limerick in Ireland. He was previously employed as a Software Design Engineer at S.u.S.E. Linux AG, Nürnberg, Germany and at the University of Applied Sciences Osnabrück, Germany. He received Diplom-Ingenieur (FH) from the University of Applied Sciences Osnabrück, Germany in 1999 and Master of Engineering from University of Limerick, Ireland in 2000. He is currently pursuing a PhD part-time at the University of Limerick. His research interests are: cryptography, security protocols, encryption algorithms, data security and network security, automated theorem proving, artificial intelligence, programming languages, operating systems, neural networks.

**Tomas Flanagan** received both a Bachelor of Engineering (B.Eng.) degree and a Master of Engineering degree (M.Eng.) from the University of Limerick, Ireland in 2000 and 2002 respectively. The M.Eng. thesis was focused on formal verification of security protocols used for mobile communications. Tomas is currently with the Engineering Design Centre at Cambridge University, UK.