Private Authentication

Martín Abadi

University of California at Santa Cruz

Cédric Fournet

Microsoft Research

Abstract

Frequently, communication between two principals reveals their identities and presence to third parties. These privacy breaches can occur even if security protocols are in use; indeed, they may even be caused by security protocols. However, with some care, security protocols can provide authentication for principals that wish to communicate while protecting them from monitoring by third parties. We discuss the problem of private authentication and present two protocols for private authentication of mobile principals. Our protocols allow two mobile principals to communicate when they meet at a location if they wish to do so, without the danger of tracking by third parties. We also present the analysis of one of the protocols in the applied pi calculus. We establish authenticity and secrecy properties. Although such properties are fairly standard, their formulation in the applied pi calculus makes an original use of process equivalences. In addition, we treat identity-protection properties, thus exploring a formal model of privacy.

1 Privacy, authenticity, and the applied pi calculus

Although privacy may coexist with communication, it often does not, and there is an intrinsic tension between them. Often, effective communication between two principals requires that they reveal their identities to each other. Still, they may wish to reveal nothing to others. Third parties should not be able to infer the identities of the two principals, nor to monitor their movements and their communication patterns. For better or for worse, they often can. In particular, a mobile principal may advertise its presence at a location in order to discover and to communicate with certain other principals at the location, thus revealing its presence also to third parties.

Authentication protocols may help in addressing these privacy breaches, as follows. When a principal A wishes to communicate with a principal B, and is willing to

To appear in TCS F-WAN special issue *Feb. 2003 (Revised: Nov. 2003)*

disclose its identity and presence to B but not to other principals, A might demand that B prove its identity before revealing anything. An authentication protocol can provide this proof. It can also serve to establish a secure channel for subsequent communication between A and B.

However, authentication protocols are not an immediate solution, and they can in fact be part of the problem. Privacy is not one of the explicit goals of common authentication protocols. These protocols often send names and credentials in cleartext, allowing any eavesdropper to see them. An eavesdropper may also learn substantial information from encrypted packets, even without knowing the corresponding decryption keys; for example, the packets may contain key identifiers that link them to other packets and to certain principals. Furthermore, in the course of authentication, a principal may reveal its identity to its interlocutor before knowing the interlocutor's identity with certainty. If A and B wish to communicate but each wants to protect its identity from third parties, who should reveal and prove theirs first?

This last difficulty is more significant in peer-to-peer communication than in client-server communication, although the desire for privacy appears in both settings.

- In client-server systems, the identity of servers is seldom protected. However, the identity of clients is not too hard to protect, and this is often deemed worthwhile. For example, in the SSL protocol [20], a client can first establish an "anonymous" connection, then authenticate with the protection of this connection, communicating its identity only in encrypted form. An eavesdropper can still obtain some addressing information, but this information may be of limited value if the client resides behind a firewall and a proxy. (Similarly, the Skeme protocol [26] provides support for protecting the identity of the initiator of a protocol session, but not the identity of the responder; the JFK protocol [8] is also asymmetric in this respect.)
- The symmetry of peer-to-peer communication makes it less plausible that one of the parties in an exchange would be willing to volunteer its identity first. Privacy may nevertheless be attractive. In particular, mobile principals may want to communicate with nearby peers without allowing others to monitor them (cf. Bluetooth [12] and its weaknesses [25]). Thus, privacy seems more problematic and potentially more interesting in the fluid setting of mobile, peer-to-peer communication.

This paper gives a definition of a privacy property (first informally, then in a process calculus). This property implies that each principal may reveal and prove its identity to certain other principals, and hide it from the rest. The definition applies even if all parties are peers and have such privacy requirements.

Standard authentication protocols do not satisfy the privacy property. However, we show two protocols that do, and undoubtedly there are others (to the extent that in-

formally described protocols can satisfy informally defined properties). In our protocols, a session between two principals A and B consists of messages encrypted under public keys and under session keys in such a way that only A and B discover each other's identity. The protocols differ from standard protocols by the absence of cleartext identity information. More subtly, they rely on some mild but non-trivial assumptions on the underlying cryptographic primitives. One of the protocols also includes a subtle "decoy" message in order to thwart certain active attacks.

Our protocols do not assume that the principals A and B have a long-term shared secret. Neither do they require an infrastructure of on-line trusted third parties, or suppose that the world is organized into domains and that each principal has a home domain. In this respect, the protocols contrast with previous ones for related purposes (see for example [30,36,11,9] and section 9). Because of their weak infrastructure needs, the protocols are consistent with ad hoc networking.

As an example, consider a mobile principal A that communicates with others when they are in the same (physical or virtual) location. In order to establish connections, A might constantly broadcast "hello, I am A, does anyone want to talk?". An eavesdropper could then detect A's presence at a particular location. An eavesdropper could even monitor A's movements without much difficulty, given sensors at sufficiently many locations. Our protocols are designed with this scenario in mind. Suppose that two principals A and B arrive anonymously at a location. Although A and B may know of each other in advance, they need not have a long-term shared key. Furthermore, neither may be certain a priori that the other one is present at this location. If they wish to communicate with one another, our protocols will enable them to do it, without the danger of being monitored by others.

This paper also presents the analysis of one of our protocols in the applied pi calculus [2], a recent variant of the pi calculus. This analysis is worthwhile for several reasons:

- As we discussed above, the protocol aims to guarantee that third parties do not learn the identity of protocol participants. Although this property and similar ones appear prominently in several recent protocol designs, they have hardly been specified and proved precisely to date. Therefore, this paper develops an approach for stating and deriving those properties.
- In addition, the protocol is for a standard purpose, namely establishing a session (with associated cryptographic keys), and it is concerned with standard security properties, such as authenticity and secrecy. Therefore, the analysis of the protocol exemplifies concepts and techniques relevant to many other protocols.
- The protocol includes some delicate features, and is not a trivial example invented in order to illustrate formal techniques. On the other hand, the protocol remains fairly simple, so we can give relatively concise treatments of its main properties.

In the applied pi calculus, the constructs of the classic pi calculus can be used to represent concurrent systems that communicate on channels, and function symbols can be used to represent cryptographic operations and other operations on data. Large classes of important attacks can also be expressed in the applied pi calculus, as contexts. These include the typical attacks for which a symbolic, mostly "black-box" view of cryptography suffices (but not for example some lower-level attacks that depend on timing behavior or on probabilities). Thus, in general, the applied pi calculus serves for describing and reasoning about many of the central aspects of security protocols. In particular, it is an appropriate setting for the analysis of the protocol for private authentication. Some of the properties of the protocol can be nicely captured in the form of equivalences between processes. Moreover, some of the properties are sensitive to the equations satisfied by the cryptographic functions upon which the protocol relies. The applied pi calculus is well-suited for expressing those equivalences and those equations.

In a sense, private authentication is about hiding the names (or identities) of protocol participants, while the applied pi calculus permits hiding the names that represent private communication channels and secret cryptographic keys (through the restriction construct ν). Despite this superficial coincidence, the name hiding of private authentication and that of the applied pi calculus are rather different. However, the name hiding of the applied pi calculus is crucial for expressing the protocol under consideration and for deriving the equivalences that express its properties.

The next section defines and discusses the privacy property sketched above. Section 3 presents the assumptions on which our protocols rely. Section 4 develops the two protocols and some optimizations and extensions. Section 5 explains the applied pi calculus. Section 6 shows how to express one of our protocols in the applied pi calculus. Section 7 treats the authenticity and secrecy properties of this protocol; section 8, its identity-protection properties. Section 9 discusses some related problems and related work (including, in particular, work on message untraceability). Section 10 concludes. An appendix contains proofs for the main claims of sections 7 and 8.

Parts of this paper have appeared in preliminary form in proceedings [1,19].

2 The problem

Although we do not aim to provide a general definition of privacy (partly because one might have to be too vague or empty), we focus on the following frequent scenario in which privacy is a central concern: two or more mobile interlocutors wish to communicate securely, protecting their messages and also their identities from third parties. This scenario arises often in mobile telephony and mobile computing [18,34,30,36,9,25]. In these contexts, roaming users may want to conceal

their identities from others and even from infrastructure providers and operators. Furthermore, identity protection is a goal of several recent protocols for communication at the IP level [26,8].

More specifically, suppose that a principal A is willing to engage in communication with some set of other principals S_A (which may change over time), and that A is willing to reveal and even prove its identity to these principals. This proof may be required, for instance if A wishes to make a sensitive request from each of these principals, or if these principals would reveal some sensitive data only to A. The problem is to enable A to authenticate to principals in S_A without requiring A to compromise its privacy by revealing its identity or S_A more broadly:

(1) A should be able to prove its identity to principals in S_A, and to establish authenticated and private communication channels with them.
(2) A should not have to indicate its identity (and presence) to any principal outside S_A.
(3) Although an individual principal may deduce whether it is in S_A from A's willingness to communicate, A should not have to reveal anything more about S_A.

Goal 1 is common; many cryptographic protocols and security infrastructures have been designed with this goal in mind.

Goal 2 is less common. As discussed above, it is seldom met with standard protocols, but it seems attractive. When C is a principal outside S_A, this goal implies that A should not have to prove its identity to C, but it also means that A should not have to give substantial hints of its identity to C.

We could consider strengthening goal 2 by saying that A should have to reveal its identity only to principals $B \in S_A$ such that $A \in S_B$, in other words, to principals with which A can actually communicate. On the other hand, if S_B is under $B's$ control, B could let $A \in S_B$, or pretend that this is the case, in order to learn A's identity. (We revisit whether S_B is under $B's$ control with the definition of compliant principal in section 6.5.)

Goal 3 concerns a further privacy guarantee. Like goal 2, it is somewhat unusual, seldom met with standard techniques, but attractive from a privacy perspective. It might be relaxed slightly, in particular allowing A to reveal the approximate size of S_A.

Note that A may be willing to engage in anonymous communication with some set of principals in addition to S_A. We expect that A is programmed and configured so that it does not spuriously reveal its identity (or other private data) to those other principals accidentally. In actual systems, however, principals may well reveal and even broadcast their names unnecessarily.

3 Assumptions

This section introduces the assumptions on which our protocols rely. They generally concern communication and cryptography, and the power of the adversary in these respects. (Menezes et al. [29] give the necessary background in cryptography; we rely only on elementary concepts.) Although the assumptions may not hold in many real systems, they are realistic enough to be implementable, and advantageously simple.

3.1 Communication

We assume that messages do not automatically reveal the identity of their senders and receivers—for example, by mentioning them in headers. When the location of the sender of a message can be obtained, for example, by triangulation, this assumption implies that the location does not reveal the sender's identity. This assumption also entails some difficulties in routing messages. Techniques for message untraceability (see for example [15,32,33] and section 9) suggest some sophisticated solutions. Focusing on a relatively simple but important case, we envision that all messages are broadcast within some small area, such as a room or a building.

We aim to protect against an adversary that can intercept any message sent on a public channel (within the small area under consideration or elsewhere). In addition, the adversary is active: it can send any message that it can compute. Thus, the adversary is essentially the standard adversary for security protocols, as described, for example, by Needham and Schroeder [31].

3.2 Cryptography

We also assume that each principal A has a public key K_A and a corresponding private key K_A^{-1}, and that the association between principals and public keys is known. This association can be implemented with the help of a mostly-off-line certification authority. In this case, some additional care is required: fetching certificates and other interactions with the certification authority should not compromise privacy goals. Alternatively, the association is trivial if we name principals by their public keys, for example as in SPKI [17]. Similarly, it is also trivial if we use ordinary principal names as public keys, with an identity-based cryptosystem [37]. Therefore, we may basically treat public keys as principal names.

When K^{-1} is a private key, we write $\{M\}_{K^{-1}}$ for M signed using K^{-1}, in such a way that M can be extracted from $\{M\}_{K^{-1}}$ and the signature verified using the corresponding public key K. As usual, we assume that signatures are unforgeable.

Similarly,[1] when K is a public key, we write $\{M\}_K$ for the encryption of M using K. We expect some properties of the encryption scheme:

(1) Only a principal that knows the corresponding private key K^{-1} should be able to recover the plaintext of a message encrypted under a public key K.
(2) Furthermore, decrypting a message with a private key K^{-1} should succeed only if the message was encrypted under the corresponding public key K, and the success or failure of a decryption should be evident to the principal who performs it.
(3) Finally, encryption should be which-key concealing [7,10,13], in the following sense. Someone who sees a message encrypted under a public key K should not be able to tell that it is under K without knowledge of the plaintext or the corresponding private key K^{-1}, even with knowledge of K and other messages under K. Similarly, someone who sees several different messages encrypted under a public key K should not be able to tell that they are under the same key without knowledge of the corresponding private key K^{-1}.

Property 1 is essential and standard. Properties 2 and 3 are not entirely standard. They are not implied by standard computational specifications of encryption (e.g., [21]) but appear in formal models (e.g., [5]). Property 2 can be implemented by including appropriate redundancy in encrypted messages, without compromising secrecy properties. It is not essential, but we find it convenient, particularly for the second protocol and its enhancements. Property 3 is satisfied with standard cryptosystems based on the discrete-logarithm problem [10,13], but it excludes implementations that tag all encryptions with key identifiers. Although the rigorous study of this property is relatively recent, it seems to be implicitly assumed in earlier work; for example, it seems to be necessary for the desired anonymity properties of the Skeme protocol [26].

4 Two protocols

This section shows two protocols that address the goals of section 2. It also discusses some variants of the protocols.

The two protocols are based on standard primitives and techniques (in particular on public-key cryptography), and resemble standard protocols. The first protocol uses digital signatures and requires that principals have loosely synchronized clocks. The second protocol uses only encryption and avoids the synchronization require-

[1] These notations are concise and fairly memorable, but perhaps somewhat misleading. In particular, they imply that the same key pair is used for both public-key signatures and encryptions, and that the underlying algorithms are similar for both kinds of operations (as in the RSA cryptosystem). We do not need to assume these properties.

ment, at the cost of an extra message. The second protocol draws attention to difficulties in achieving privacy against an active adversary.

Undoubtedly, other protocols satisfy the goals of section 2. In particular, these goals seem relatively easy to satisfy when all principals confide in on-line authentication servers. However, the existence of ubiquitous trusted servers may not be a reasonable assumption. The protocols of this section do not rely on such trusted third parties.

4.1 First protocol

In the first protocol, when a principal A wishes to talk to another principal B, and B is willing to talk to a set of principals S_B, A and B proceed as follows:

- A generates fresh key material K and a timestamp T, and sends out

  $$\text{“hello”}, \{\text{“hello”}, K_A, \{K_A, K_B, K, T\}_{K_A^{-1}}\}_{K_B}$$

  The tag “hello” indicates the type of the message; it is not essential in this particular protocol. The key material may simply be a session key, for subsequent communication; it may also consist of several session keys and identifiers for those keys. The signature means that the principal with public key K_A (that is, A) says that it has generated the key material K for communicating with the principal with public key K_B (that is, B) near time T. The explicit mention of K_B is crucial for security (see [6]).
- Upon receipt of any message that consists of “hello” and (apparently) a ciphertext, the recipient B tries to decrypt the second component using its private key. If the decryption yields a key K_A and a signed statement of the form $\{K_A, K_B, K, T\}_{K_A^{-1}}$, then B extracts K_A and K, verifies the signature using K_A, ensures that the message is not a replay using the timestamp T, and checks that $A \in S_B$. If the plaintext is not of the expected form, if the message is a replay, or if $A \notin S_B$, then B does nothing.
- A and B may use K for encrypting subsequent messages. Each of these messages may be tagged with a key identifier, derived from K but independent of A and B. When A or B receives a tagged message, the key identifier suggests the use of K for decrypting the message.

This protocol is based on the Denning-Sacco public-key protocol and its corrected version [16,6]. Noticeably, however, this protocol does not include any identities in cleartext. In addition, the protocol requires stronger assumptions on encryption, specifically that public-key encryption under K_B be which-key concealing. This property is needed so that A's encrypted message does not reveal the identity of its (intended) recipient B.

When A wishes to communicate with several principals $B_1, \ldots, B_n$ at the same time (for example, when A arrives at a new location), A may simply start n instances of the protocol in parallel, sending different key material to each of $B_1, \ldots, B_n$. Those of $B_1, \ldots, B_n$ who are present and willing to communicate with A will be able to do so using the key material. (Section 4.4 describes optimizations of the second protocol for this situation.)

4.2 Second protocol

In the second protocol, when a principal A wishes to talk to another principal B, and B is willing to talk to a set of principals S_B, A and B proceed as follows:

- A generates a fresh, unpredictable nonce N_A, and sends out

  $$\text{“hello”}, \{\text{“hello”}, N_A, K_A\}_{K_B}$$

  (In security protocols, nonces are quantities generated for the purpose of being recent; they are typically used in challenge-response exchanges.)
- Upon receipt of a message that consists of “hello” and (apparently) a ciphertext, the recipient B checks that it is not a replay [2] and tries to decrypt the second component using its private key. If the decryption succeeds, then B extracts the corresponding nonce N_A and key K_A, checks that $A \in S_B$, generates a fresh, unpredictable nonce N_B, and sends out

  $$\text{“ack”}, \{\text{“ack”}, N_A, N_B, K_B\}_{K_A}$$

  If the message is a replay, if the decryption fails, if the plaintext is not of the expected form, or if $A \notin S_B$, then B sends out a “decoy” message. This message should basically look like B’s other message. In particular, it may have the form

  $$\text{“ack”}, \{N\}_K$$

  where N is a fresh nonce (with padding, as needed) and only B knows K^{-1}, or it may be indistinguishable from a message of this form.
- Upon receipt of a message that consists of “ack” and (apparently) a ciphertext, A tries to decrypt the second component using its private key. If the decryption succeeds, then A extracts the corresponding nonces N_A and N_B and key K_B, and checks that it has recently sent N_A encrypted under K_B. If the decryption or the checks fail, then A does nothing.

[2] The filtering of replays by B is not in the original description of the protocol [1], and may be avoided under certain conditions on B’s behavior, but we believe that it is a reasonable refinement, with useful consequences. We omit the details of how to implement the filtering, which are fairly standard; as usual, some but not all implementations preserve security properties.

- Subsequently, A and B may use N_A and N_B as shared secrets. In particular, A and B may use N_B as a session key, or they may compute session keys by concatenating and hashing the two nonces. They may also derive key identifiers, much as in the first protocol.

In summary, the message flow of a successful exchange is:

$$A \to B : \text{“hello”}, \{\text{“hello”}, N_A, K_A\}_{K_B}$$
$$B \to A : \text{“ack”}, \{\text{“ack”}, N_A, N_B, K_B\}_{K_A}$$

Section 4.4 describes variants of this basic pattern, for example (as mentioned above) for the case where A wishes to communicate with n principals $B_1, \ldots, B_n$.

This protocol has some similarities with the Needham-Schroeder public-key protocol [31] and others [27,26]. However, like the first protocol, this one does not include any identities in cleartext, and again that is not quite enough for privacy. As in the first protocol, public-key encryption should be which-key concealing so that encrypted messages do not reveal the identities of their (intended) recipients. Furthermore, the delicate use of the decoy message is important:

- B's decoy message is unfortunately necessary in order to prevent an attack where a malicious principal $C \notin S_B$ computes and sends

  $$\text{“hello”}, \{\text{“hello”}, N_C, K_A\}_{K_B}$$

  and then deduces B's presence and $A \in S_B$ by noticing a response. In order to prevent this attack, the decoy message should look to C like it has the form

  $$\text{“ack”}, \{\text{“ack”}, N_C, N_B, K_B\}_{K_A}$$

- B's response to A when $A \notin S_B$ should look as though B was someone else, lest A infer B's presence. Since B sends a decoy message when its decryption fails, it should also send one when $A \notin S_B$.

The decoy message “ack”, $\{N\}_K$ is intended to address both of these requirements.

4.3 Properties and limitations

Intuitively, the protocols are supposed to establish shared secrets between A and B. At the very least, we would expect that A and B, and only them, can obtain a session key from these secrets. We would expect, moreover, that this key be essentially independent of any other data. For example, it should not be possible for an attacker without access to the key to compute a ciphertext under the key from a record of

the protocol messages. In short, the key should behave much like a pre-established shared key. The only observable differences between running the protocol and having a pre-established shared key should be that an attacker can disrupt a protocol run, making it fail, and that an attacker can notice that the protocol generates some opaque messages. Our results of section 7 provide a more precise statement of this comparison, in the form of an equivalence, for the second protocol.

The protocols are also supposed to assure A and B of each other's identity. However, the two participants have somewhat different states in this respect at the conclusion of a key exchange.

- With the first protocol, after receiving and checking A's message, B has evidence that A is attempting to establish a session. On the other hand, A knows nothing about B's presence and interest in a session until receiving messages under the session key.
- With the second protocol, after receiving and checking B's message, A has evidence that it shares the session key with the principal B that responded. On the other hand, B has evidence that it shares the session key at most with A, but cannot be certain that A initiated the protocol run. Any other principal C might have contacted B pretending to be A, but then C will not obtain the key. Only after further communication can B be sure of A's participation in the session.

In addition, the protocols are supposed to protect the identity of the participants. This should mean, in particular, that an attacker cannot learn anything when A wishes to communicate with B but not vice versa. It should also mean that an attacker cannot distinguish a run between A and B from a run between two other principals A' and B', under appropriate hypotheses. The hypotheses should say, for example, that B is not the attacker, since B learns A's identity. The hypotheses should also consider what the participants can do besides running the protocol. For example, if A were to broadcast "A has a secret!" after every protocol run, then A's identity would clearly not be protected. Similarly, if A would only contact C after sessions with B, then C could infer B's recent presence from A's behavior. In general, the hypotheses need to address possible leaks not caused by the protocol itself. Section 8 develops these hypotheses and gives our privacy results, also relying on equivalences.

The protocols do not provide location information, so they do not guarantee that two principals A and B that establish a session are necessarily in the same location. In a distributed system, a relay could allow A and B to establish a session remotely, perhaps with the intention of misleading A and B. Assuming that each principal can name its own location, the protocols can easily be extended with location indicators in order to detect relays across locations.

4.4 Efficiency considerations

Both protocols can be rather inefficient in some respects. These inefficiencies are largely unavoidable consequences of the goals of private authentication.

- A generates its message and sends it before having any indication that B is present and willing to communicate. In other situations, A might have first engaged in a lightweight handshake with B, sending the names A and B and waiting for an acknowledgment. Alternatively, both A and B might have broadcast their names and their interest in communicating with nearby principals. Here, these preliminary messages are in conflict with the privacy goals, even though they do not absolutely prove the presence of A and B to an eavesdropper. Some compromises may be possible; for example, A and B may publish some bits of information about their identities if those bits are not deemed too sensitive. In addition, in the second protocol, A may precompute its message.
- Following the protocols, B may examine many messages that were encrypted under the public keys of other principals. This examination may be costly, perhaps opening the door to a denial-of-service attack against B. In other situations, A might have included the name B, the key K_B, or some identifier for K_B in clear in its message, as a hint for B. Here, again, the optimization is in conflict with the privacy goals, and some compromises may be possible.

The second protocol introduces some further inefficiencies, but those can be addressed as follows:

- In the second protocol, A may process many acknowledgments that were encrypted under the public keys of other principals. This problem can be solved through the use of a connection identifier: A can create a fresh identifier I, send it to B, and B can return I in clear as a hint that A should decrypt its message.

  $$A \to B : \text{“hello”}, I, \{\text{“hello”}, N_A, K_A\}_{K_B}$$
  $$B \to A : \text{“ack”}, I, \{\text{“ack”}, N_A, N_B, K_B\}_{K_A}$$

  The identifier I should also appear in B's decoy message. Third parties may deduce that the messages are linked, because I is outside the encryptions, but cannot relate the messages to A and B.
- Suppose that A wishes to communicate with several principals, $B_1, \ldots, B_n$. It could initiate n instances of the protocol. However, combining the messages from all the instances can be faster. In particular, although each of $B_1, \ldots, B_n$ should receive a different nonce, they can all share a connection identifier. Moreover, when K_A is long, its public-key encryption may be implemented as a public-key encryption of a shorter symmetric key K plus an encryption of K_A using K; the key K and the latter encryption may be the same for $B_1, \ldots, B_n$.

Thus, A may send:

$$\text{“hello”},\ I,\ \{K_A\}_K,\ \{\text{“hello”}, H(K_A),\ N_{A1},\ K\}_{K_{B_1}}, \ldots,$$
$$\{\text{“hello”}, H(K_A),\ N_{An},\ K\}_{K_{B_n}}$$

where H is a one-way hash function. Most importantly, the need for decoy messages is drastically reduced. A principal that plays the role of B need not produce n true or decoy acknowledgments, but only one. Specifically, B should reply to a ciphertext encrypted under K_B, if A included one in its message, and send a decoy message otherwise. This last optimization depends on our assumption that B can recognize whether a ciphertext was produced by encryption under K_B.

We have not attempted a careful analysis of these variants, or a thorough study of alternative designs (for instance, with other treatments of identifiers). There are opportunities for further work in these directions.

With these and other improvements, both protocols are practical enough in certain systems, although they do not scale well. Suppose that principals wish to communicate with few other principals at a time, and that any one message reaches few principals, for instance because messages are broadcast within small locations; then it should be possible for principals that come into contact to establish private, authenticated connections (or fail to do so) within seconds. What is “few”? A simple calculation indicates that 10 is few, and maybe 100 is few, but 1000 is probably not few. Typically, the limiting performance factor will be public-key cryptography, rather than communications: each public-key operation takes a few milliseconds or tens of milliseconds in software on modern processors (e.g., [28]). Perhaps the development of custom cryptographic techniques (flavors of broadcast encryption) can lead to further efficiency gains.

4.5 Groups

In the problem described above, the set of principals S_A and S_B with which A and B wish to communicate, respectively, are essentially presented as sets of public keys. In variants of the problem, S_A, S_B, or both may be presented in other ways. The protocols can be extended to some situations where a principal wants to deal with others not because of their identities but because of their attributes or memberships in groups, such as “ACME printers” or “Italians”. These extensions are not all completely satisfactory.

- Suppose that B is willing to communicate with any principal in a certain group, without having a full list of those principals. However, let us still assume that S_A is presented as a set of public keys. In this case, we can extend our protocols without much trouble: A can include certificates in its encrypted message to B,

proving its membership in groups.

- Suppose that, instead, A wants to communicate with any principal in a certain group, and S_B is presented as a set of public keys. The roles in the protocols may be reversed to handle this case.
- However, the protocols do not address the case in which neither S_A nor S_B is presented as a set of public keys, for example when both are presented as groups. Introducing group keys should reduce this case to familiar ones, but group keys are harder to manage and protect.

5 The applied pi calculus (overview)

The applied pi calculus is a simple, general extension of the pi calculus with value passing, primitive function symbols, and equations between terms. In [2], we introduce this calculus, develop semantics and proof techniques, and apply those techniques in reasoning about some security protocols. This section gives only a brief overview. Later sections return to private authentication, relying on the applied pi calculus.

5.1 Syntax and informal semantics

A *signature* Σ consists of a finite set of function symbols, such as h and decrypt, each with an integer arity. Given a signature Σ, an infinite set of names, and an infinite set of variables, the set of *terms* is defined by the grammar:

$U, V, W ::=$	terms
$a, n, \ldots$	name
$x, y, \ldots$	variable
$f(U_1, \ldots, U_l)$	function application

where f ranges over the function symbols of Σ and l matches the arity of f. We use meta-variables u and v to range over both names and variables. We write $U = V$ to indicate that U and V are equal in an underlying equational theory associated with Σ.

The grammar for *processes* is similar to the one in the pi calculus, except that here messages can contain terms (rather than only names) and that names need not be just channel names:

$P, Q, R ::=$	processes (or plain processes)
$\mathbf{0}$	null process
$P \mid Q$	parallel composition

$!P$	replication
$\nu n.P$	name restriction (“new”)
$\mathit{if}\ U = V\ \mathit{then}\ P\ \mathit{else}\ Q$	conditional
$u(x).P$	message input
$\overline{u}\langle V\rangle.P$	message output

The null process $\mathbf{0}$ does nothing; $P \mid Q$ is the parallel composition of P and Q; the replication $!P$ behaves as an infinite number of copies of P running in parallel. The process $\nu n.P$ makes a new name n then behaves as P. The conditional construct $\mathit{if}\ U = V\ \mathit{then}\ P\ \mathit{else}\ Q$ is standard, but we should stress that $U = V$ represents equality, rather than strict syntactic identity. We abbreviate it $\mathit{if}\ U = V\ \mathit{then}\ P$ when Q is $\mathbf{0}$. Finally, the input process $u(x).P$ is ready to input from channel u, then to run P with the actual message replaced for the formal parameter x, while the output process $\overline{u}\langle V\rangle.P$ is ready to output message V on channel u, then to run P. In both of these, we may omit P when it is $\mathbf{0}$.

Further, we extend processes with *active substitutions*:

$A, B, C ::=$	extended processes
P	plain process
$A \mid B$	parallel composition
$\nu n.A$	name restriction
$\nu x.A$	variable restriction
$\{x = V\}$	active substitution

We write $\{x = V\}$ for the substitution that replaces the variable x with the term V. The substitution $\{x = V\}$ typically appears when the term V has been sent to the environment, but the environment may not have the atomic names that appear in V; the variable x is just a way to refer to V in this situation. The substitution $\{x = V\}$ is active in the sense that it “floats” and applies to any process that comes into contact with it. In order to control this contact, we may add a variable restriction: $\nu x.(\{x = V\} \mid P)$ corresponds exactly to $\mathit{let}\ x = V\ \mathit{in}\ P$. Although the substitution $\{x = V\}$ concerns only one variable, we can build bigger substitutions by parallel composition. We always assume that our substitutions are cycle-free. We also assume that, in an extended process, there is at most one substitution for each variable, and there is exactly one when the variable is restricted.

A *frame* is an extended process built up from active substitutions by parallel composition and restriction. Informally, frames represent the static knowledge gathered by the environment after communications with an extended process. We let φ range over frames, and let $\varphi(A)$ be the frame obtained from the extended process A by erasing all plain subprocesses of A. We let $dom(\varphi)$ be the set of variables defined by substitutions in φ and not restricted in φ. As usual, names and variables have scopes, which are delimited by restrictions and by inputs. When E is any expression, $fv(E)$, $dv(E)$, $bv(E)$, $fn(E)$, and $bn(E)$ are the sets of free, defined, and

bound variables and free and bound names of E, respectively; E is closed when every variable is either bound or defined by an active substitution. An *evaluation context* $C[\_]$ is an extended process with a hole in the place of an extended process. The context $C[\_]$ *closes* A when $C[A]$ is closed.

We rely on a sort system for terms and extended processes [2, section 2]. We always assume that terms and extended processes are well-sorted and that substitutions and context applications preserve sorts.

5.2 Examples

We further explain the applied pi calculus with examples motivated by our second protocol. We start with formatted messages. We then discuss one-way hash functions and encryption functions.

In that protocol, we use two kinds of formated messages ("hello" and "ack") with two and three variable fields, respectively. Accordingly, we introduce binary and ternary function symbols $\mathsf{hello}(\_,\_)$ and $\mathsf{ack}(\_,\_,\_)$ in the signature Σ; these symbols represent the message constructors. In addition, we introduce inverse, unary function symbols $\mathsf{hello.0}\,(\_)$, $\mathsf{hello.1}\,(\_)$, $\mathsf{ack.0}\,(\_)$, $\mathsf{ack.1}\,(\_)$, and $\mathsf{ack.2}\,(\_)$ in order to select particular fields in messages. Finally, we describe the intended behavior of formatted messages with the evident equations:

$$\begin{aligned}
\mathsf{hello.0}\,(\mathsf{hello}(x_0, x_1)) &= x_0 \\
\mathsf{hello.1}\,(\mathsf{hello}(x_0, x_1)) &= x_1 \\
\mathsf{ack.0}\,(\mathsf{ack}(y_0, y_1, y_2)) &= y_0 \\
\mathsf{ack.1}\,(\mathsf{ack}(y_0, y_1, y_2)) &= y_1 \\
\mathsf{ack.2}\,(\mathsf{ack}(y_0, y_1, y_2)) &= y_2
\end{aligned}$$

A first equational theory may consist of these equations, and all equations obtained by reflexivity, symmetry, and transitivity and by substituting terms for the variables $x_0, \ldots, y_2$.

In order to model the one-way hash computation of a session key out of the nonces N_A and N_B, we introduce a binary function symbol $\mathsf{h}(\_,\_)$ with no equations. The fact that $\mathsf{h}(N_A, N_B) = \mathsf{h}(N'_A, N'_B)$ only when $N_A = N'_A$ and $N_B = N'_B$ models that h is collision-free. The absence of an inverse for h models the one-wayness of h. In our protocol, these properties are important to guarantee that $\mathsf{h}(N_A, N_B)$ is indeed secret (as long as N_A or N_B is) and, further, that the attacker cannot recover N_A or N_B even if it obtains $\mathsf{h}(N_A, N_B)$.

In order to model symmetric cryptography (that is, shared-key cryptography), we

may introduce binary function symbols $\mathsf{encrypt}(\_,\_)$ and $\mathsf{decrypt}(\_,\_)$ for encryption and decryption, respectively, with the equation:

$$\mathsf{decrypt}(\mathsf{encrypt}(x,y),y) = x \tag{1}$$

Here x represents the plaintext and y the key. We often use the notation $\{U\}_V$ instead of $\mathsf{encrypt}(U,V)$. For instance, the (useless) process $\nu K.\bar{c}\langle\{U\}_K\rangle$ sends the term U encrypted under a fresh key K on channel c. It is only slightly harder to model asymmetric (public-key) cryptography, where the keys for encryption and decryption are different. In addition to $\mathsf{encrypt}(\_,\_)$ and $\mathsf{decrypt}(\_,\_)$, we introduce the unary function symbol $\mathsf{pk}(\_)$ for deriving a public key from a private key. Instead of (1), we use the equation:

$$\mathsf{decrypt}(\mathsf{encrypt}(x,\mathsf{pk}(y)),y) = x \tag{2}$$

Since there is no inverse for $\mathsf{pk}(\_)$, a public key $\mathsf{pk}(s)$ can be passed to the environment without giving away the capability to decrypt messages encrypted under $\mathsf{pk}(s)$.

For instance, a principal B with public key K_B can be represented as a process in a context $P_B[\_] \stackrel{\text{def}}{=} \nu s.(\{K_B = \mathsf{pk}(s)\} \mid [\_])$ that binds a decryption key s and exports the associated encryption key as a variable K_B. As this example indicates, we essentially view ν as a generator of unguessable seeds. In some cases, those seeds may be directly used as passwords or keys; in others, some transformations are needed.

5.3 Operational semantics

Given a signature Σ, we equip it with an equational theory (that is, with an equivalence relation on terms with certain closure properties). We write $\Sigma \vdash U = V$ when the equation $U = V$ is in the theory associated with Σ. We usually keep the theory implicit, and abbreviate $\Sigma \vdash U = V$ to $U = V$ when Σ is clear from context or unimportant. We write $(U = V)\varphi$ when U and V are equal after applying φ, with α-conversion on names and variables bound in φ and free in U or V [2, section 4.2].

Structural equivalences, written $A \equiv B$, relate extended processes that are equal by rearrangements of parallel compositions, restrictions, and active substitutions, and by equational rewriting of terms. Formally, structural equivalence is defined as the smallest equivalence relation on extended processes that is closed by α-conversion

on both names and variables, by application of evaluation contexts, and such that:

$$\begin{array}{lrl}
\textsc{Par-0} & A & \equiv A \mid \mathbf{0} \\
\textsc{Par-A} & A \mid (B \mid C) & \equiv (A \mid B) \mid C \\
\textsc{Par-C} & A \mid B & \equiv B \mid A \\
\textsc{Repl} & !P & \equiv P \mid !P \\
\textsc{New-0} & \nu n.\mathbf{0} & \equiv \mathbf{0} \\
\textsc{New-C} & \nu u.\nu v.A & \equiv \nu v.\nu u.A \\
\textsc{New-Par} & A \mid \nu u.B & \equiv \nu u.(A \mid B) \quad \text{when } u \notin \mathit{fv}(A) \cup \mathit{fn}(A) \\
\textsc{Alias} & \nu x.\{x = V\} & \equiv \mathbf{0} \\
\textsc{Subst} & \{x = V\} \mid A & \equiv \{x = V\} \mid A\{x = V\} \\
\textsc{Rewrite} & \{x = U\} & \equiv \{x = V\} \quad \text{when } \Sigma \vdash U = V
\end{array}$$

We say that a variable x can be *derived* from the extended process A when, for some term V and extended process A', we have $A \equiv \{x = V\} \mid A'$. Intuitively, if x can be derived from A, then A does not reveal more information than $\nu x.A$, because the context can build the term V and use it instead of x.

Reductions, written $A \to B$, represent silent steps of computation. Reduction is defined as the smallest relation on extended processes closed by structural equivalence and application of evaluation contexts such that:

$$\begin{array}{lrl}
\textsc{Comm} & \overline{a}\langle x\rangle.P \mid a(x).Q & \to P \mid Q \\
\textsc{Then} & \mathit{if}\ U = U\ \mathit{then}\ P\ \mathit{else}\ Q & \to P \\
\textsc{Else} & \mathit{if}\ U = V\ \mathit{then}\ P\ \mathit{else}\ Q & \to Q \\
 & \multicolumn{2}{l}{\text{for any ground terms } U \text{ and } V \text{ such that } \Sigma \nvdash U = V}
\end{array}$$

Labelled transitions, written $A \xrightarrow{\alpha} B$, represent interactions with the environment. They consist of message inputs and message outputs, respectively written $A \xrightarrow{a(V)} B$ and $A \xrightarrow{\nu\widetilde{u}.\overline{a}\langle V\rangle} B$, with $\{\widetilde{u}\} \subseteq \mathit{fv}(V) \cup \mathit{fn}(V) \setminus \{a\}$. In both, a represents a communication channel and V a message. In outputs, $\widetilde{u}$ collects the names and variables revealed by the message. The labelled transition relation is defined as the smallest relation indexed by labels α that is closed by structural equivalence and

such that:

IN $\quad a(x).P \xrightarrow{a(V)} P\{x = V\}$ $\qquad$ OUT $\quad \overline{a}\langle V\rangle.P \xrightarrow{\overline{a}\langle V\rangle} P$

OPEN-CHANNEL
$$\frac{A \xrightarrow{\overline{a}\langle b\rangle} A' \qquad b \neq a}{\nu b.A \xrightarrow{\nu b.\overline{a}\langle b\rangle} A'}$$

OPEN-VARIABLE
$$\frac{A \xrightarrow{\nu \widetilde{u}.\overline{a}\langle V\rangle} A' \qquad x \in fv(V) \setminus \{\widetilde{u}\}, z \text{ fresh} \qquad x \text{ can be derived from } \nu\widetilde{u}.(\{z = V\} \mid A')}{\nu x.A \xrightarrow{\nu x,\widetilde{u}.\overline{a}\langle V\rangle} A'}$$

SCOPE
$$\frac{A \xrightarrow{\alpha} A' \qquad \alpha \text{ is } \overline{a}\langle V\rangle \text{ or } a(V), u \text{ does not occur in } \alpha}{\nu u.A \xrightarrow{\alpha} \nu u.A'}$$

PAR
$$\frac{A \xrightarrow{\alpha} A' \qquad bv(\alpha) \cap fv(B) = bn(\alpha) \cap fn(B) = \emptyset}{A \mid B \xrightarrow{\alpha} A' \mid B}$$

In contrast with some other process calculi, output transitions $A \xrightarrow{\nu \widetilde{u}.\overline{a}\langle V\rangle} B$ are enabled only for messages V that effectively reveal the names and variables in $\widetilde{u}$. Typically, the transition is just of the form $A \xrightarrow{\nu x.\overline{a}\langle x\rangle} B$ for some fresh variable x, and B contains an active substitution that associates x with a more complex message. Input transitions $A \xrightarrow{a(V)} B$ may use variables defined in A (typically from previous message outputs) to form the message V.

5.4 Observational equivalences

In the analysis of protocols, we frequently argue that two given processes cannot be distinguished by any context, that is, that the processes are observationally equivalent. As in the spi calculus, the context represents an active attacker, and equivalences capture security properties in the presence of the attacker. The applied pi calculus has a useful, general theory of observational equivalence parameterized by Σ and its equational theory [2]. Specifically, the following three relations are defined for any Σ and equational theory:

- *Static equivalence*, written $\varphi \approx_s \psi$, relates frames with the same domain that cannot be distinguished by any term comparison: $dom(\varphi) = dom(\psi)$ and, for all terms U and V, we have $(U = V)\varphi$ if and only if $(U = V)\psi$. Static equivalence is closed by structural equivalence, by reduction, and by application of closing evaluation contexts. In the presence of the ν construct, this relation is somewhat

delicate and interesting. For instance, we have

$$\nu N.\{x = \mathsf{h}(N, K_B)\} \approx_s \nu N.\{x = \mathsf{h}(N, K_C)\}$$

for any K_B and K_C, since the nonce N guarantees that both terms substituted for x have the same (null) equational properties, but

$$\nu N.\{x = \mathsf{hello}(N, K_B)\} \not\approx_s \nu N.\{x = \mathsf{hello}(N, K_C)\}$$

as soon as K_B and K_C differ, since the comparison $\mathsf{hello}.1\,(x) = K_B$ succeeds only with the first frame.

- More generally, *contextual equivalence* relates extended processes that cannot be distinguished by any evaluation context in the applied pi calculus, with any combination of messaging and term comparisons. Observational equivalence coincides with static equivalence on frames, but is strictly finer on extended processes.
- *Labelled bisimilarity*, written $\approx_l$, coincides with contextual equivalence, but it is defined in terms of labelled transitions instead of arbitrary evaluation contexts, and it is the basis for standard, powerful proof techniques. We state our main results in terms of $\approx_l$. We recall its definition below.

Definition 1 Labelled bisimilarity *($\approx_l$) is the largest symmetric relation $\mathcal{R}$ on closed extended processes such that $A\ \mathcal{R}\ B$ implies:*

(1) $A \approx_s B$,
(2) if $A \to A'$, *then* $B \to^* B'$ *and* $A'\ \mathcal{R}\ B'$ *for some* B',
(3) if $A \xrightarrow{\alpha} A'$ *and* $fv(\alpha) \subseteq dom(A)$ *and* $bn(\alpha) \cap fn(B) = \emptyset$, *then* $B \to^* \xrightarrow{\alpha} \to^* B'$ *and* $A'\ \mathcal{R}\ B'$ *for some* B'.

As usual, strong labelled bisimilarity ($\sim_l$) is defined analogously, requiring $B \to B'$ and $B \xrightarrow{\alpha} B'$ instead of $B \to^* B'$ and $B \to^* \xrightarrow{\alpha} \to^* B'$, respectively, in the bisimulation clauses.

6 The second protocol in the applied pi calculus

In this section we give a precise model for our second protocol (described in section 4.2) in the applied pi calculus: we first choose an adequate equational theory, then detail our representation of principals and attackers, and finally give processes that express the protocol.

We believe that the first protocol could be studied along similar lines. It introduces one complication (the modeling of timestamps), but is otherwise much simpler.

The following grammar of terms indicates the function symbols and notation conventions that we use:

$T, U, V, V_0, W, \ldots ::=$	terms
$A, B, K, x_1, x_2, \ldots$	variable
$c_1, c_2, \mathit{init}_A, \mathit{accept}_B, \mathit{connect}_A, \ldots$	name (for a channel)
$N, N_A, K_A^{-1}, \ldots$	name (typically for a nonce or a key)
$\mathsf{h}(U, V)$	cryptographic hash
$\mathsf{pk}(U)$	public-key derivation
$\{T\}_V$	public-key encryption
$\mathsf{decrypt}(W, U)$	private-key decryption
$\mathsf{hello}(U_0, U_1), \mathsf{ack}(V_0, V_1, V_2)$	constructor for protocol message
$\mathsf{hello.0}\,(U), \ldots, \mathsf{ack.2}\,(V)$	field selector for protocol message
$\emptyset$	empty set
$U.V$	set extension

This grammar includes primitives for constructing sets ($\emptyset$ and .) but not a set membership relation. We write $V \in W$ as an abbreviation for $W.V = W$.

Our equational theory is fairly standard. The equations on terms are:

$\mathsf{decrypt}(\{x\}_{\mathsf{pk}(z)}, z) = x$	private-key decryption
$\mathsf{hello.j}\,(\mathsf{hello}(x_0, x_1)) = x_j$	field selection in "hello" message
$\mathsf{ack.j}\,(\mathsf{ack}(x_0, x_1, x_2)) = x_j$	field selection in "ack" message
$(\emptyset.x).x = \emptyset.x$	idempotence of set extension
$(x.y).z = (x.z).y$	commutativity of set extension

The equational theory implicitly assumes that encryption is which-key concealing, in the sense that someone who sees a message encrypted under a public key K should not be able to tell that it is under K without knowledge of the corresponding private key K^{-1} (see section 3.2). On the other hand, it would be easy to add functions and equations that negate this property, in order to model additional capabilities of an attacker. In particular, for the benefit of the attacker, we could add the function symbols get-key, test-key, or same-key, with respective equations:

$$\mathsf{get\text{-}key}(\{x\}_z) = z$$
$$\mathsf{test\text{-}key}(\{x\}_z, z) = \mathsf{true}$$
$$\mathsf{same\text{-}key}(\{x\}_z, \{y\}_z) = \mathsf{true}$$

These additions would not affect authentication and secrecy properties, but they would compromise privacy properties.

6.2 The network and the attacker

In our model of the protocol, network messages are transmitted (asynchronously) on the channels named c_1 and c_2. These represent two public communication channels, or a single public channel, perhaps the ether, in which tags serve for differentiating traffic flows.

As explained in section 3, we assume that an attacker can interpose itself on all public communication channels. In our model, an arbitrary environment (an arbitrary evaluation context) represents the attacker. This environment can interact with the configuration of principals using labelled transitions on any free channel name. We obtain an attractively simple representation of broadcast communication: each message is simply made available to the attacker, on a public channel, and the attacker may then decide to transmit the message, again on a public channel, to one or more principals.

As a special case, we sometimes model a weaker, passive attacker that only eavesdrops on messages. An attack step—that is, eavesdropping on a message—amounts to a message interception (formally, with an output label) followed by a re-emission of the same message (with an input label). We write $A \xrightarrow{\nu\widetilde{u}.c[\widetilde{V}]} A'$ as a shorthand for the sequence of two transitions $A \xrightarrow{\nu\widetilde{u}.\overline{c}\langle\widetilde{V}\rangle} \xrightarrow{c(\widetilde{V})} A'$. Here, $\xrightarrow{\nu\widetilde{u}.\overline{c}\langle\widetilde{V}\rangle}$ shows an output of the protocol and $\xrightarrow{c(\widetilde{V})}$ shows the same message being input.

6.3 The principals

We model arbitrary configurations of principals. Each principal may run any number of sessions and may perform other operations after session establishment or even independently of the protocol. Only some of these principals are trustworthy. We are interested in the security properties that hold for them.

Our model of a principal A has two parts: an implementation of the protocol, written $\mathcal{P}_A$, and a "user process" (or "user protocol"), written U_A. The user process defines any additional behavior, such as when protocol runs are initiated and what happens after each session establishment. It consumes the shared secrets produced during the establishment of sessions and uses these secrets. According to the user process, each principal may run several sessions of the protocol, possibly playing both the role of initiator and that of responder. Of course, security properties depend on both $\mathcal{P}_A$ and U_A. We define $\mathcal{P}_A$ below in section 6.4; on the other hand, we

treat U_A as a parameter.

We use the following control interface between the (abstract) user process and the (specific) session-establishment protocol. The interface concerns both the roles of session initiator and responder.

***init*:** U_A sends $\overline{\mathit{init}_A}\langle B\rangle$ to trigger a session-establishment attempt with principal B.

***accept*:** $\mathcal{P}_B$ sends $\overline{\mathit{accept}_B}\langle A, K\rangle$ to notify U_B that it has accepted a session apparently from principal A, with session key K.

***connect*:** $\mathcal{P}_A$ sends $\overline{\mathit{connect}_A}\langle B, K\rangle$ to notify U_A that its attempt to contact principal B succeeded, with session key K.

In addition, for each principal B, the set S_B represents all acceptable interlocutors for B. For simplicity, we do not provide an interface for updating this set, so it remains constant (and therefore not under the control of U_B). Thus, the interface between the session-establishment protocol and the user process for each principal X consists of the communication channels $\mathcal{V}_X \stackrel{\text{def}}{=} \{\mathit{init}_X, \mathit{accept}_X, \mathit{connect}_X\}$ plus a (constant) set of principals S_X. These channels can be restricted (with ν) in order to hide the interface from the environment.

Note that the interface provides a key K to the user process, rather than nonces N_A and N_B. We prefer to define K in such a way that N_A and N_B cannot be computed from K (for example, $K = \mathsf{h}(N_A, N_B)$). Our results can thus be independent of how the user process applies K.

As suggested in the informal description of the protocol, we represent the identity of each principal as its public key, using variables A, B, ... for both identities and public keys. For the present purposes, the essence of a principal lies in its ability to decrypt any message encrypted under its public key. Accordingly, we associate a context of the form

$$\mathit{PK}_A\,[\_] \stackrel{\text{def}}{=} \nu K_A^{-1}.\Big(\{A = \mathsf{pk}(K_A^{-1})\} \mid [\_]\Big)$$

with every principal identity A. This context restricts the use of the decryption key K_A^{-1} to the process in the context and it exports the corresponding public key. Whenever we put a process R in this context, our intent is that R never communicates K_A^{-1} to the environment.

By definition of well-formed configurations in the applied pi calculus, a process of the form $C[\mathit{PK}_A\,[R]]$ exports A, only R can access K_A^{-1}, and we cannot apply a context that would redefine A. On the other hand, $C[\_]$ can define any number of other principals. Thus, we obtain a fairly generous and convenient model when we represent an attacker by an arbitrary context.

For example, the process $\mathit{PK}_A\,[\mathbf{0}]$ indicates that A is a principal whose decryption

key is never used. This process concisely models an absent principal.

6.4 The protocol

In this section we give a formal counterpart to the description of message flows of section 4.2.

Messages We rely on substitutions in order to define the protocol messages and the key derivation, as follows.

$$\sigma_1 \stackrel{\text{def}}{=} \{x_1 = \{\mathsf{hello}(N_A, A)\}_B\}$$
$$\sigma_2 \stackrel{\text{def}}{=} \{x_2 = \{\mathsf{ack}(N_A, N_B, B)\}_A\}$$
$$\sigma_2^\circ \stackrel{\text{def}}{=} \{x_2 = N_B\}$$
$$\sigma_K \stackrel{\text{def}}{=} \{K = \mathsf{h}(N_A, N_B)\}$$

Although N_A and N_B are free here, they represent fresh nonces. They will be bound in any process that introduces these substitutions. The substitution σ_2° corresponds to the responder's decoy message, in which here we use a name rather than a ciphertext, for simplicity.

Syntactic sugar We sometimes use the following abbreviations.

For testing, we write $\mathit{if}\ U_1 = V_1\ \mathit{and}\ U_2 = V_2\ \mathit{then}\ P\ \mathit{else}\ Q$ for the process $\mathit{if}\ U_1 = V_1\ \mathit{then}\ (\mathit{if}\ U_2 = V_2\ \mathit{then}\ P\ \mathit{else}\ Q)\ \mathit{else}\ Q$, and rely on other similar abbreviations.

For decryption, we use pattern matching on message contents. Specifically, we write

$$\mathit{if}\ x = \{\mathsf{ack}(N_A, \nu N_B, B)\}_A\ \mathit{using}\ K_A^{-1}\ \mathit{then}\ P\ \mathit{else}\ Q$$

for the process

$$\nu N_B.\begin{pmatrix}\{N_B = \mathsf{ack}.1\left(\mathsf{decrypt}(x, K_A^{-1})\right)\} \mid \\ \mathit{if}\ x = \{\mathsf{ack}(N_A, N_B, B)\}_A\ \mathit{then}\ P\ \mathit{else}\ Q\end{pmatrix}$$

with the assumption that $N_B \notin \mathit{fv}(Q)$, and we use analogous abbreviations with νA and νN_A. Here, we use the identifiers N_A and N_B as variables rather than names, locally.

For filtering duplicate messages, we write

$$!c_1(x \setminus V).\mathit{if}\ x\ \mathit{fresh}\ \mathit{then}\ P\ \mathit{else}\ Q$$

for the process

$$\nu c.(\overline{c}\langle V\rangle \mid !c_1(x).c(s).(\overline{c}\langle s.x\rangle \mid \mathit{if}\ x \in s\ \mathit{then}\ Q\ \mathit{else}\ P))$$

where c is a fresh channel name and s is a fresh variable. We use channel c for maintaining a set V of previously received messages; Q is triggered instead of P when one of those messages is received again.

Processes The following code represents the protocol. It includes definitions of processes for the initiator role and for the responder role. We write A for the initiator and B for the responder, but the definitions apply to every principal by renaming.

$$\begin{aligned}
\mathcal{P}_A &\stackrel{\text{def}}{=} I_A \mid R_A \\
I_A &\stackrel{\text{def}}{=} !\mathit{init}_A(B).\nu N_A.\,(\overline{c}_1\langle x_1\sigma_1\rangle \mid I'_A) \\
I'_A &\stackrel{\text{def}}{=} c_2(x_2). \\
&\qquad \mathit{if}\ x_2 = \{\mathsf{ack}(N_A, \nu N_B, B)\}_A\ \mathit{using}\ K_A^{-1}\ \mathit{then}\ \overline{\mathit{connect}}_A\langle B, K\sigma_K\rangle \\
R_B &\stackrel{\text{def}}{=} !c_1(x_1 \setminus \emptyset). \\
&\qquad \mathit{if}\ x_1\ \mathit{fresh}\ \mathit{and}\ x_1 = \{\mathsf{hello}(\nu N_A, \nu A)\}_B\ \mathit{using}\ K_B^{-1}\ \mathit{and}\ A \in S_B \\
&\qquad \mathit{then}\ \nu N_B.\,(\overline{c}_2\langle x_2\sigma_2\rangle \mid \overline{\mathit{accept}}_B\langle A, K\sigma_K\rangle)\ \mathit{else}\ \nu N_B.\overline{c}_2\langle x_2\sigma_2^\circ\rangle
\end{aligned}$$

Here, I_A shows the initiator receiving a session request on channel init_A and sending the first protocol message; I'_A then shows the initiator receiving and checking a response, and passing a session key on channel $\mathit{connect}_A$ if the response is satisfactory. On the other hand, R_B shows the responder receiving a message, processing it, responding, and in some cases passing a session key on channel accept_B. Both I_A and R_B are replicated processes.

As coded, the protocol has little resistance to multiplexing errors. In particular, the initiator fails if the first response that it receives is not the expected one. We could add retries without much difficulty, but this aspect of the protocol is mostly irrelevant in the study of safety properties.

In our statements of security properties (not in the definition of the protocol itself), we distinguish a particular finite, non-empty set $\mathcal{C}$ of compliant principals A, B, ... A compliant principal A is one in which the decryption key K_A^{-1} is used exclusively in the session-establishment protocol. The initial configuration of a single compliant principal A with user process U_A is therefore an extended process of the form:

$$\mathcal{Q}_A \stackrel{\text{def}}{=} \nu\mathcal{V}_A.\big(U_A \mid \mathit{PK}_A\,[\mathcal{P}_A]\big)$$

This extended process is parameterized by the set S_A, and (at least) exports the variable A and has free channels c_1 and c_2. In $\mathcal{Q}_A$, by definition, U_A does not have access to K_A^{-1}.

Combining several such extended processes, we obtain a global configuration of the form $\prod_{A\in\mathcal{C}} \mathcal{Q}_A$ for any set of compliant principals $\mathcal{C}$. Sometimes, however, we do not need to distinguish the user processes of several compliant principals. We can instead group them in a single (compound) user process U, letting $U = \prod_{A\in\mathcal{C}} U_A$. Then, letting $\mathcal{V} = \bigcup_{A\in\mathcal{C}} \mathcal{V}_A$, we consider configurations of the form:

$$\mathcal{P} \stackrel{\text{def}}{=} \prod_{A\in\mathcal{C}} \mathit{PK}_A\,[\mathcal{P}_A]$$
$$\mathcal{Q} \stackrel{\text{def}}{=} \nu\mathcal{V}.\big(U \mid \mathcal{P}\big)$$

We assume that the user processes of compliant principals (U_A and U) never communicate control channels ($\mathcal{V}$) in messages. For instance, the process $\overline{c}_1\langle \mathit{connect}_A\rangle$ cannot be the user process of a compliant principal. This assumption can easily be enforced by the sort system.

We use $\mathcal{P}$ in section 7 when we establish security properties that do not depend on U, thus effectively regarding U as part of the attacker. We use $\mathcal{Q}$ in section 8, with additional hypotheses on U, when we study privacy.

7 Authentication and secrecy properties

We begin our analysis of the protocol with traditional properties, namely responder authentication and session-key secrecy. We state and discuss the properties, leaving proofs for an appendix. Such standard properties are important, and often a prerequisite for privacy properties. Moreover, their formulation in the applied pi calculus illustrates the use of observational equivalence for expressing security properties. In contrast, many other formalisms for similar purposes rely only on properties of traces, rather than on equivalences.

For a given set of compliant principals $\mathcal{C}$, we study runs of the protocol in the presence of an active attacker, by examining transitions $\mathcal{P} \xrightarrow{\eta} P'$ from the configuration $\mathcal{P}$ defined above to some configuration P', where η is an arbitrary sequence of labels.

In our statements, we let ω and φ abbreviate the series of actions and the equational "net effect", respectively, of a successful run of the protocol:

$$\xrightarrow{\omega} \stackrel{\text{def}}{=} \xrightarrow{\mathit{init}_A(B)} \xrightarrow{\nu x_1.c_1[x_1]} \longrightarrow^* \xrightarrow{\nu x_2.c_2[x_2]} \longrightarrow \xrightarrow{\nu K.\overline{\mathit{accept}}_B\langle A,K\rangle} \xrightarrow{\overline{\mathit{connect}}_A\langle B,K\rangle}$$

$$\varphi \stackrel{\text{def}}{=} \nu N_A.\,(\sigma_1 \mid \nu N_B.(\sigma_2 \mid \sigma_K))$$

Thus, ω shows a message that initiates a session-establishment attempt from A to B, then two messages x_1 and x_2 on channels c_1 and c_2, respectively, then some internal steps, and finally two messages that represent the establishment of a session with a key K at B and A, respectively. The environment learns x_1 and x_2 by eavesdropping. According to the frame φ, x_1 represents the "hello" message and x_2 represents the "ack" message; in addition, φ binds K to its value $\mathsf{h}(N_A, N_B)$. Similarly, we let ω^- and φ^- abbreviate the series of actions and the equational "net effect", respectively, of a failed (rejected) run of the protocol:

$$\xrightarrow{\omega^-} \stackrel{\text{def}}{=} \xrightarrow{\mathit{init}_A(B)} \xrightarrow{\nu x_1.c_1[x_1]} \longrightarrow^* \xrightarrow{\nu x_2.c_2[x_2]} \longrightarrow$$

$$\varphi^- \stackrel{\text{def}}{=} (\nu N_A.\sigma_1) \mid (\nu N_B.\sigma_2^\circ)$$

We have that if $A \in S_B$ then

$$\mathcal{P} \xrightarrow{\omega} \mathcal{P}_{x_1} \mid \varphi$$

else

$$\mathcal{P} \xrightarrow{\omega^-} \mathcal{P}_{x_1} \mid \varphi^-$$

where $\mathcal{P}_{x_1}$ is $\mathcal{P}$ updated so that R_B holds an element x_1 in the set of messages it has received. Thus, $\mathcal{P}$ may perform a complete run of the protocol, and this run succeeds if authorized by the responder and fails otherwise. More generally (in part because of the replications in $\mathcal{P}$), for any P' such that $\mathcal{P} \xrightarrow{\eta} P'$, we have that if $A \in S_B$ then

$$P' \xrightarrow{\omega} P'_{x_1} \mid \varphi$$

else

$$P' \xrightarrow{\omega^-} P'_{x_1} \mid \varphi^-$$

where P'_{x_1} is a corresponding update of P'. These results express the functional correctness of the protocol. They hold independently of whether encryption is which-key concealing.

The first theorem relates the two possible outcomes of an actual run to a "magical" outcome $\varphi^\circ \stackrel{\text{def}}{=} \nu N_1.\{x_1 = N_1\} \mid \nu N_2.\{x_2 = N_2\}$ where the two intercepted messages are trivially independent of the principals A and B and of the established key.

Theorem 2 (Key freshness for complete runs) *Let $A, B \in \mathcal{C}$.*

(1) (Success:) If $\mathcal{P} \xrightarrow{\eta} P'$ and $A \in S_B$, then $P' \xrightarrow{\omega} \approx_l P' \mid \varphi^\circ \mid \nu N.\{K = N\}$.
(Failure:) If $\mathcal{P} \xrightarrow{\eta} P'$ and $A \notin S_B$, then $P' \xrightarrow{\omega^-} \approx_l P' \mid \varphi^\circ$.
(2) Conversely, if $\mathcal{P} \xrightarrow{\omega} P''$, then $A \in S_B$ and $P'' \approx_l \mathcal{P} \mid \varphi^\circ \mid \nu N.\{K = N\}$.

For instance, if $A \in S_B$ then $\mathcal{P} \xrightarrow{\omega} \mathcal{P}_{x_1} \mid \varphi$, as explained above; in this case the theorem yields $\mathcal{P}_{x_1} \mid \varphi \approx_l \mathcal{P} \mid \varphi^\circ \mid \nu N.\{K = N\}$, so the environment cannot distinguish the actual messages and key (on the left-hand side) from fresh, independent names (on the right-hand side). The active substitution $\nu N.\{K = N\}$ exports the simplest definition of a fresh secret key, a fresh name, rather than an expression computed from x_1 and x_2.

Interestingly, φ° and $\nu N.\{K = N\}$ do not depend on A and B at all, so this theorem implies a first privacy guarantee: one does not learn anything about A and B from $\varphi^\circ \mid \nu N.\{K = N\}$, and hence from φ. The equivalences $\approx_l$ are used for rewriting $P'_{x_1} \mid \varphi$ and $P'_{x_1} \mid \varphi^-$, by simplifying φ and φ^- and by erasing x_1 from the set of messages that R_B has received, returning to the process P' and hiding that a run has occurred. These equivalences hold only if encryption is which-key-concealing. Otherwise, we would obtain only:

$$P'_{x_1} \mid \varphi \approx_l P'_{x_1} \mid (\nu N_A.\sigma_1) \mid (\nu N_A N_B.\sigma_2) \mid (\nu N.\{K = N\})$$

On the right-hand side, we are left with messages x_1 and x_2 that contain the public keys of A and B. Nonetheless, N_A and N_B are bound around σ_1 and σ_2, so the independence of the session key is still guaranteed.

A direct corollary concerns two instances $\mathcal{P}_A$ and $\mathcal{P}_B$ of the protocol in the initial state. This corollary emphasizes the transitions observed by an environment with no access to the control channels.

$$\mathcal{P}_A \mid \mathcal{P}_B \mid \overline{\mathit{init}}_A\langle B\rangle \;\longrightarrow \xrightarrow{\nu x_1.c_1[x_1]} \longrightarrow^* \xrightarrow{\nu x_2.c_2[x_2]} \longrightarrow \approx_l$$

$$\mathcal{P}_A \mid \mathcal{P}_B \mid \varphi^\circ \mid \begin{cases} \nu N.(\overline{\mathit{accept}}_B\langle A, N\rangle \mid \overline{\mathit{connect}}_A\langle B, N\rangle) & \text{if } A \in S_B \\ \mathbf{0} & \text{if } A \notin S_B \end{cases}$$

Intuitively, when we erase control messages, we obtain the same trace and equational effect whether or not $A \in S_B$.

We also obtain a complementary authentication property:

Theorem 3 (Responder authentication) *Suppose that* $\mathcal{P} \xrightarrow{\eta} P'$ *and (1)* $\mathcal{P} \xrightarrow{\eta} P'$ *has no internal communication step on* c_1 *and* c_2*; (2)* P' *has no immediate output on channel* $accept_B$*.*

If $\overline{connect}_A\langle B, K\rangle$ *occurs in* η*, then* $\mathcal{P} \xrightarrow{\omega}\xrightarrow{\eta'} P'$ *for some permutation* $\omega\eta'$ *of* η*.*

In the statement of the theorem, we rely on α-conversion and assume that the names and variables in processes and labels never clash. With this standard assumption, the commutation of two transition steps (when enabled) can be written simply as the commutation of their labels. Conditions 1 and 2 are technically convenient for avoiding special cases in the statement of the theorem, but they are not essential. Condition 1 rules out traces where a message on c_1 or c_2 is not intercepted by the attacker, and is instead transmitted internally. (Formally, any internal communication $A \rightarrow A'$ on channel c_i implies that $A \xrightarrow{\nu x_i.c_i[x_i]} A''$ with $A' \equiv \nu x_i.A''$.) Condition 2 rules out traces where the transition $accept_B$ in ω has not occurred and is enabled in P'.

In light of the results above, we can interpret this theorem partly as a correspondence assertion: whenever A receives a connection message after a protocol run, apparently with B, we have that

(1) A initiated the session with B;
(2) B accepted the session with A;
(3) both parties are now sharing a fresh key K, as good as a fresh shared name; and
(4) intercepted messages x_1 and x_2 are seemingly unrelated to A, B, and K.

8 Privacy properties

In this section, we focus on privacy properties. For this purpose, we need to consider the behavior of user processes, not just the protocol itself (see section 4.3). For a given set of compliant principals $\mathcal{C}$, we address the question of whether an attacker can distinguish two (compound) user processes U_1 and U_2 when we place these processes in the context $\nu\mathcal{V}.([\_]|\mathcal{P})$ that provides local access to the session-establishment protocol. Therefore, indistinguishability for user processes depends on the identity-protection features of the protocol, and it is coarser than ordinary observational equivalence $\approx_l$ (that is, indistinguishability in all evaluation contexts).

For instance, if U_1 and U_2 each contain a message $\overline{init}_{A_1}\langle B_1\rangle$ and $\overline{init}_{A_2}\langle B_2\rangle$, and if U_1 and U_2 "behave similarly" once a session is established, then U_1 and U_2 are indistinguishable in this specific sense. On the other hand, we have $\overline{init}_{A_1}\langle B_1\rangle \approx_l \overline{init}_{A_2}\langle B_2\rangle$ only if $A_1 = A_2$ and $B_1 = B_2$.

In order to capture this notion of indistinguishability without having to pick particular user processes, we introduce a special labelled transition system and a notion of bisimulation. We obtain a general result in terms of that notion of bisimulation, then derive some privacy properties as corollaries. Thus, for the study of a particular protocol, we develop a special notion of observation of user processes. In contrast, in recent, related work [4,3], we take a standard notion of observation, and develop communication protocols that are secure with respect to it (and which, for instance, rely on "noise" messages in order to hide communication patterns between compliant principals).

We adopt the following notation convention. We write A, B for principals in the set of compliant principals $\mathcal{C}$, and E for a principal not in $\mathcal{C}$.

8.1 A labelled transition system

Next we define labelled transitions for user processes with control state. The control state records the sets S_B of acceptable interlocutors and abstractly keeps track of the sessions being negotiated. The labelled transitions reflect only what the environment can observe about these sessions, filtering out identity information.

Formally, a control state ρ consists of two functions, one that maps each principal $B \in \mathcal{C}$ to a set S_B, and the other a finite map from integers to entries t. The entries are of four kinds:

- $A\ B$: a session offer from A to B not yet considered by B.
- $A\ B\ K_i$: a session offer from A to B accepted by B with key K_i (when $A \in S_B$).
- $A\ B\ -$: a session offer from A to B rejected by B (when $A \notin S_B$).
- $A\ E$: a session offer from A to some non-compliant principal E.

For any ρ and any integer i not in ρ's domain, we let $\rho[i \mapsto t]$ be the control state that extends ρ by mapping i to t. We assume that the keys K_i are all distinct. We let $\mathcal{V}_\rho$ be the union of $\mathcal{V}$ with the keys K_i for all integers i in the domain of ρ.

We pair a process with a control state, with the notation $\rho : U$. We assume that K_i is free in U only if ρ maps i to an entry of the form $A\ B\ K_i$. (In $\mathcal{Q}$, the user process U may have free variables defined by $\mathcal{P}$, such as variables A and B that represent compliant principals, or K_i for a computed key. When we consider transitions of U or $\rho : U$, we treat these variables as names.)

Such a pair $\rho : U$ may have the three sorts of transitions $\rho : U \xrightarrow{\gamma} \rho' : U'$ that we define next: ordinary transitions, blinded transitions, and external transitions.

- Ordinary transitions are essentially those of the process U. Let $\xrightarrow{\lambda}$ range over $\rightarrow$ and $\xrightarrow{\alpha}$ for all labels α that do not contain control channels or bind keys K_i (that

is, $fn(\alpha) \cap \mathcal{V}_\rho = \emptyset$ and $bn(\alpha) \cap (\mathcal{C} \cup \mathcal{V}_\rho) = \emptyset$). We have:

$$\textsc{Lift}\ \frac{U \xrightarrow{\lambda} U'}{\rho : U \xrightarrow{\lambda} \rho : U'}$$

- The attacker can blindly intercept all messages sent on public channels by the principals in $\mathcal{C}$ and resend any of these messages later. Specifically, the attacker can notice new session attempts, make responders consider session offers (either genuine or fake), and make initiators consider intercepted "ack" messages. These attacker actions are correlated with messages on restricted control channels, which the attacker cannot observe directly. Accordingly, we reflect these actions using blinded transitions $\xrightarrow{\overline{init}\,\nu i}$, $\xrightarrow{accept\ i}$, $\xrightarrow{accept_B(A)}$, and $\xrightarrow{connect\ i}$.

$$\textsc{Init}\ \frac{U \xrightarrow{\overline{init}_A\langle B\rangle} U'}{\rho : U \xrightarrow{\overline{init}\,\nu i} \rho[i \mapsto A\ B] : U'}$$

ACCEPT

$$\rho[i \mapsto A\ B] : U \xrightarrow{accept\ i} \begin{cases} \rho[i \mapsto A\ B\ K_i] : U \mid \overline{accept}_B\langle A, K_i\rangle & \text{if } A \in S_B \\ \rho[i \mapsto A\ B\ -\ ] : U & \text{if } A \notin S_B \end{cases}$$

ACCEPT-FAKE

$$\rho : U \xrightarrow{accept_B(A)} \begin{cases} \rho : U \mid \nu N.\overline{accept}_B\langle A, N\rangle & \text{if } A \in S_B \\ \rho : U & \text{if } A \notin S_B \end{cases}$$

CONNECT

$$\rho[i \mapsto A\ B\ K_i] : U \xrightarrow{connect\ i} \rho : \nu K_i.(U \mid \overline{connect}_A\langle B, K_i\rangle)$$

$$\rho[i \mapsto A\ B\ -\ ] : U \xrightarrow{connect\ i} \rho : U$$

- In addition, compliant principals may be willing to open sessions with non-compliant ones. These sessions are also mediated by the protocol, even if they are transparent to the attacker who can in principle decrypt all messages in these sessions. We reflect these actions using external transitions $\xrightarrow{\nu iE.\overline{init}_A\langle i,E\rangle}$, $\xrightarrow{accept_B(W,V)}$, $\xrightarrow{connect_A(i,E,V)}$, where E is a variable and V and W are terms such that $fn(V) \cap \mathcal{V}_\rho = fn(W) \cap \mathcal{V}_\rho = \emptyset$.

$$\text{INIT-E}\ \frac{U \xrightarrow{\nu E.\overline{init}_A\langle E\rangle} U'}{\rho : U \xrightarrow{\nu i E.\overline{init}_A\langle i,E\rangle} \rho[i \mapsto A\ E] : U'}\ \text{when } (E \neq B)\varphi(U') \text{ for all } B \in \mathcal{C}$$

$$\text{ACCEPT-E}\quad \rho : U \xrightarrow{accept_B(W,V)} \rho : U \mid \overline{accept}_B\langle W, V\rangle \quad \begin{array}{l}\text{when } (W = A)\varphi(U)\\ \text{for some } A \in S_B \setminus \mathcal{C}\end{array}$$

$$\text{CONNECT-E}\quad \rho[i \mapsto A\ E] : U \xrightarrow{connect_A(i,E,V)} \rho : U \mid \overline{connect}_A\langle E, V\rangle$$

8.2 Private bisimulation

In order to express hypotheses on the observable properties of user processes, we define an ad hoc notion of bisimulation:

Definition 4 Private bisimilarity *($\approx_{\mathcal{C}}$) is the largest symmetric relation $\mathcal{R}$ on extended processes with control state such that, whenever $T_1\ \mathcal{R}\ T_2$ with $T_1 = \rho_1 : U_1$ and $T_2 = \rho_2 : U_2$, we have:*

(1) $\nu\mathcal{V}_{\rho_1}.U_1 \approx_s \nu\mathcal{V}_{\rho_2}.U_2$,
(2) if $T_1 \to T_1'$, *then* $T_2 \to^* T_2'$ *and* $T_1'\ \mathcal{R}\ T_2'$ *for some* T_2',
(3) if $T_1 \xrightarrow{\gamma} T_1'$ *and* $fv(\gamma) \subseteq dom(\nu\mathcal{V}_{\rho_1}.U_1)$ *and* $bn(\gamma) \cap fn(\nu\mathcal{V}_{\rho_2}.U_2) = \emptyset$, *then* $T_2 \to^* \xrightarrow{\gamma} \to^* T_2'$ *and* $T_1'\ \mathcal{R}\ T_2'$ *for some* T_2'.

This definition is an adaptation of that of weak labelled bisimilarity for the applied pi calculus (Definition 1 in section 5.4). The three clauses are analogous to those for the applied pi calculus; the main novelty here is that $\xrightarrow{\gamma}$ ranges over different transitions in clause 3.

We also let ε range over initial control states, that is, control states that have no session entries and only define sets S_B for $B \in \mathcal{C}$. We write $\mathcal{P}(\varepsilon)$ for the protocol $\mathcal{P}$ with these sets S_B. When ε is clear from context, we may write (as usual) $\mathcal{P}$ instead of $\mathcal{P}(\varepsilon)$.

Our main privacy result states that, if two user processes are privately bisimilar (under our new notion of bisimulation), then the two corresponding configurations are observationally equivalent from the environment's point of view. As we show below, this result provides an effective proof technique for privacy properties.

Lemma 5 (Privacy) *If* $\varepsilon_1 : U_1 \approx_{\mathcal{C}} \varepsilon_2 : U_2$, *then*

$$\nu\mathcal{V}.\big(U_1 \mid \mathcal{P}(\varepsilon_1)\big) \approx_l \nu\mathcal{V}.\big(U_2 \mid \mathcal{P}(\varepsilon_2)\big)$$

The hypothesis $\varepsilon_1 : U_1 \approx_{\mathcal{C}} \varepsilon_2 : U_2$ deals with arbitrary user processes and sets S_B, and is typically not difficult to establish in particular cases. Importantly, its statement does not depend on any detail of the session-establishment protocol, only on its control interface. The conclusion $\nu\mathcal{V}.(U_1 \mid \mathcal{P}(\varepsilon_1)) \approx_l \nu\mathcal{V}.(U_2 \mid \mathcal{P}(\varepsilon_2))$ then says that two composite systems, each with a user process, are indistinguishable.

The converse of Lemma 5 does not quite hold, at least because the definition of labelled transitions is conservative in some respects. (For instance, in that definition, we safely presume that the attacker has a private key associated with any value E that U employs to identify a non-compliant principal.) Thus, user processes that are not privately bisimilar may still be part of indistinguishable systems. Such user processes can be excluded with additional hypotheses.

8.3 Applications of the Privacy Lemma

One may formulate and prove many specific privacy properties for the protocol. The various properties may differ, in particular, on which user processes and sets S_B they consider. We give a series of simple examples of such properties. In the examples, the hypotheses can usually be made less demanding, and more specific and complicated. The proofs follow directly from Lemma 5.

We begin with a basic example that concerns the anonymity of failed sessions. Provided that U never inputs on channel init_X for any $X \in \mathcal{C}$, if $A \notin S_B$ and $A' \notin S_{B'}$, then replacing $\overline{\mathit{init}}_A\langle B\rangle$ with $\overline{\mathit{init}}_{A'}\langle B'\rangle$ in U does not affect $\mathcal{Q}$ up to observational equivalence.

The next result deals with a single initial session attempt, and states that the session attempt may not compromise any private bisimilarity that would hold after establishing the session.

Theorem 6 (Equivalent sessions) *For $j = 1, 2$, let*

$$U_j \stackrel{\mathit{def}}{=} \overline{\mathit{init}}_{A_j}\langle B_j\rangle \mid \mathit{connect}_{A_j}(B_j, K).V_j$$
$$U'_j \stackrel{\mathit{def}}{=} \nu K.\left(\overline{\mathit{accept}}_{B_j}\langle A_j, K\rangle \mid V_j\right)$$

with $A_j, B_j \in \mathcal{C}$ and $A_j \in S_{B_j}$ in ε_j. If $\varepsilon_1 : U'_1 \approx_{\mathcal{C}} \varepsilon_2 : U'_2$, then $\varepsilon_1 : U_1 \approx_{\mathcal{C}} \varepsilon_2 : U_2$.

For any user processes V_1 and V_2 that do not use the control channels, the private bisimilarity hypothesis holds as soon as $\nu K.V_1 \approx_l \nu K.V_2$. With this additional assumption and Lemma 5, we have a corollary expressed in terms of standard labelled bisimilarity: we obtain that if $\nu K.V_1 \approx_l \nu K.V_2$ then $\nu\mathcal{V}.(U_1 \mid \mathcal{P}(\varepsilon_1)) \approx_l \nu\mathcal{V}.(U_2 \mid \mathcal{P}(\varepsilon_2))$.

A further privacy property concerns compliant principals that attempt to open ses-

sions with one another but do not perform any action observable by the attacker after establishing a session. (They may for instance use private channels, or public channels with adequate decoys.) We may describe any such configuration of principals in $\mathcal{C}$ by a parallel compositions of init_A messages with $A \in \mathcal{C}$, plus the sets $(S_B)_{B \in \mathcal{C}}$. In this special case, we easily characterize the equivalence of two configurations:

Theorem 7 (Silent sessions) *Let U_1 and U_2 be parallel compositions of messages $\overline{\mathit{init}}_A\langle X \rangle$ with $A \in \mathcal{C}$. If*

(1) U_1 and U_2 contain the same number of messages,
(2) U_1 and U_2 contain exactly the same messages $\overline{\mathit{init}}_A\langle W \rangle$ for $W \notin \mathcal{C}$, and
(3) the sets $S_B \setminus \mathcal{C}$ are identical in ε_1 and ε_2,

then $\nu\mathcal{V}.(U_1 \mid \mathcal{P}(\varepsilon_1)) \approx_l \nu\mathcal{V}.(U_2 \mid \mathcal{P}(\varepsilon_2))$.

In order to prove the theorem, we may establish $\varepsilon_1 : U_1 \approx_{\mathcal{C}} \varepsilon_2 : U_2$ by enumerating a few blinded and external transitions, then apply Lemma 5. Conversely, the three hypotheses seem necessary for the conclusion, since the attacker can count the number of "hello" messages, can decrypt "hello" messages sent to principals outside $\mathcal{C}$ (as long as W is a public key not in $\mathcal{C}$), and can attempt to establish a session with any $B \in \mathcal{C}$.

We can derive other similar privacy results for uniform families of user processes, such as processes that do not use any principal identity after establishing sessions.

Our final result relates a configuration with a present but silent principal to a configuration with an absent principal. (This theorem does not require Lemma 5; it has a simple, direct bisimulation proof.)

Theorem 8 (Absent principal) *Assume that $|\mathcal{C}| > 1$, and let $X \notin \mathcal{C}$ and $S_X = \emptyset$. We have:*

$$\mathcal{Q} \mid \nu\mathcal{V}_X.\mathit{PK}_X\,[\mathcal{P}_X] \approx_l \mathcal{Q} \mid \mathit{PK}_X\,[\mathbf{0}]$$

The process on the left-hand side is structurally equivalent to a configuration $\mathcal{Q}'$ with compliant principals $\mathcal{C} \cup \{X\}$; the process on the right-hand side includes an absent principal (a principal X whose decryption key is never used). Hence, one may first use private bisimilarity to show that X is apparently not involved in any session in $\mathcal{Q}'$, then apply Theorem 8 to substitute an absent principal for X. (Conversely, if $\mathcal{C} = \{\}$ or $\mathcal{C} = \{A\}$, then the addition of any instance of the protocol is observable.)

9 Related problems and related work

The questions treated here are related to traffic analysis, and how to prevent it. This subject is not new, of course. In particular, work on message untraceability has dealt with the question of hiding (unlinking) the origins and destinations of messages (e.g., [15,32,33]). It has produced techniques that allow a principal A to send messages to a principal B in such a way that an adversary may know the identities of A and B and their locations, but not that they are communicating with one another. Those techniques address how to route a message from A to B without leaking information. In the case of cellular networks, those techniques can be adapted to hide the locations of principals [18,34]. In contrast, here we envision that all messages are broadcast within a location, simplifying routing issues, and focus on hiding the identities of principals that meet and communicate at the location. Other interesting work on untraceability in mobile networks has addressed some important authentication problems under substantial infrastructure assumptions, for instance that each principal has a home domain and that an authentication server runs in each domain [30,36,9]. That work focuses on the interaction between a mobile client and an authentication server of a domain that the client visits, typically with some privacy guarantees for the former but not for the latter. In contrast, we do not rely on those infrastructure assumptions and we focus on the interaction between two mobile principals with potentially similar privacy requirements.

There has been other research on various aspects of security in systems with mobility (e.g., [14,40,39] in addition to [18,25,30,36,11,9], cited above). Some of that work touches on privacy issues. In particular, the work of Jakobsson and Wetzel points out some privacy problems in Bluetooth. The protocols of this paper are designed to address such problems.

The questions treated here are also related to the delicate balance between privacy and authenticity in other contexts. This balance plays an important role in electronic cash systems (e.g., [23]). It can also appear in traditional access control. Specifically, suppose that A makes a request to B, and that A is member of a group that appears in the access control list that B consults for the request. In order to conceal its identity, A might use a ring signature [35] for the request, establishing that the request is from a member of the group without letting B discover that A produced the signature. However, it may not be obvious to A that showing its membership could help, and B may not wish to publish the access control list. Furthermore, A may not wish to show all its memberships to B. Thus, there is a conflict between privacy and authenticity in the communication between A and B. No third parties need be involved. In contrast, we do not guarantee the privacy of A and B with respect to each other, and focus on protecting them against third parties.

Designated verifier proofs address another trade-off between confidentiality and authenticity [24]. They allow a principal A to construct a proof that will convince

only a designated principal B. For instance, only B may be convinced of A's identity. Designated verifier proofs differ from the protocols of this paper in their set-up and applications (e.g., for fair exchange). Moreover, in general, they may leak information about A and B to third parties, without necessarily convincing them. Therefore, at least in general, they need not provide a solution to the problem of private authentication treated in this paper.

More broadly, this paper is partly a contribution to the formal study of security protocols and of their properties. In recent years, the understanding of basic security properties such as integrity and confidentiality has become both deeper and wider. There has also been substantial progress in the design and verification of protocols that aim to guarantee these properties. On the other hand, fundamental tasks such as secure session establishment remain the subject of active, productive research. Moreover, properties beyond integrity and confidentiality have been studied rather lightly to date. These properties include, for example, protection of identity information and protection against denial-of-service attacks. They may seem secondary but they are sometimes important.

The literature contains many other formal treatments of protocols. We will not attempt to survey that work here, but mention only the two most relevant papers. One of them is our original paper on the applied pi calculus [2], which considers session establishment and some of its properties, and which includes additional references. The other is a recent paper by Hughes and Shmatikov that defines several notions of anonymity and privacy [22]. A preliminary version of that paper [38] sketches—in just a few sentences—an analysis of the protocol that is the subject of this paper. Hughes and Shmatikov develop a special formal framework for protocols, communication graphs. Despite some thematic overlap, the applied pi calculus appears to be richer than communication graphs. In particular, communication graphs do not include an account of user processes. While the definitions of anonymity and privacy seem appropriate and useful for communication graphs, it is not yet entirely clear whether and how they would carry over to the applied pi calculus and other settings.

10 Conclusions

Security protocols can contribute to the tension between communication and privacy, but they can also help resolve it. In this paper, we construct two protocols that allow principals to authenticate with chosen interlocutors while hiding their identities from others. In particular, the protocols allow mobile principals to communicate when they meet, without being monitored by third parties. The protocols resemble standard ones, but interestingly they rely on some non-standard assumptions and messages to pursue non-standard objectives. As virtually all protocols, however, they are only meaningful in the context of complete systems. They are

part of a growing suite of technical and non-technical approaches to privacy.

We also analyze one of the protocols in the applied pi calculus. We cover standard authenticity and secrecy properties and also privacy (identity protection) properties. The formulation of these properties mainly relies on equivalences, which express indistinguishability in an arbitrary context. Our analysis concerns not only the core of the protocol but also its composition with a user process, since this composition may endanger privacy properties. Thus, we examine the protocol under several hypotheses on user processes. We obtain several related results that transfer hypotheses on user processes to security properties of complete systems.

Acknowledgements

Markus Jakobsson and Mike Reiter provided encouragement and useful references. The access-control scenario sketched in section 9 arose in the context of SPKI [17] during discussions with Carl Ellison, Alan Kotok, and Andrew Palka in 1997. Discussions with Vitaly Shmatikov were helpful in thinking about section 4.4 (though this section does not report on Vitaly's ideas). Hubert Comon and Véronique Cortier started to study the protocols in the spi calculus. Mike Burrows suggested an interesting variant of the second protocol with timestamps and without decoy messages. Hugo Krawczyk confirmed points related to the Skeme protocol. Anand Desai made available information on his unpublished work. Mary Baker suggested many improvements in the presentation of this paper.

Martín Abadi's work was started at Bell Labs Research, Lucent Technologies, and at InterTrust's Strategic Technologies and Architectural Research Laboratory, and was partly done at Microsoft Research, Silicon Valley. Martín Abadi's work was also partly supported by the National Science Foundation under Grants CCR-0204162 and CCR-0208800.

References

[1] Martín Abadi. Private authentication. In *Proceedings of the Workshop on Privacy Enhancing Technologies (PET 2002)*, volume 2482 of *LNCS*, pages 27–40. Springer-Verlag, 2003.

[2] Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In *Proceedings of the 28th ACM Symposium on Principles of Programming Languages (POPL 2001)*, pages 104–115. ACM, January 2001.

[3] Martín Abadi, Cédric Fournet, and Georges Gonthier. Authentication primitives and their compilation. In *Proceedings of the 27th ACM Symposium on Principles of Programming Languages (POPL 2000)*, pages 302–315. ACM, January 2000.

[4] Martín Abadi, Cédric Fournet, and Georges Gonthier. Secure implementation of channel abstractions. *Information and Computation*, 174(1):37–83, April 2002.

[5] Martín Abadi and Andrew D. Gordon. A calculus for cryptographic protocols: The spi calculus. *Information and Computation*, 148(1):1–70, January 1999. An extended version appeared as Digital Equipment Corporation Systems Research Center report No. 149, January 1998.

[6] Martín Abadi and Roger Needham. Prudent engineering practice for cryptographic protocols. *IEEE Transactions on Software Engineering*, 22(1):6–15, January 1996.

[7] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (The computational soundness of formal encryption). In *Proceedings of the First IFIP International Conference on Theoretical Computer Science*, volume 1872 of *LNCS*, pages 3–22. Springer-Verlag, August 2000.

[8] William Aiello, Steven M. Bellovin, Matt Blaze, Ran Canetti, John Ionnidis, Angelos D. Keromytis, and Omer Reingold. Efficient, DoS-resistant, secure key exchange for internet protocols. In Vijay Atluri, editor, *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, pages 48–58. ACM, November 2002.

[9] Giuseppe Ateniese, Amir Herzberg, Hugo Krawczyk, and Gene Tsudik. On traveling incognito. *Computer Networks*, 31(8):871–884, 1999.

[10] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *Advances in Cryptology—ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 566–582. Springer-Verlag, 2001.

[11] V. Bharghavan and C. V. Ramamoorthy. Security issues in mobile communications. In *Proceedings of the Second International Symposium on Autonomous Decentralized Systems*, pages 19–24, 1995.

[12] Specification of the Bluetooth system (core, v1.0b). On the Web at `http://www.bluetooth.com`, December 1999.

[13] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *Advances in Cryptology—EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer-Verlag, 2001.

[14] Luca Cardelli. Mobility and security. In F.L. Bauer and R. Steinbrueggen, editors, *Foundations of Secure Computation*, NATO Science Series, pages 1–37. IOS Press, 2000. Volume for the 20th International Summer School on Foundations of Secure Computation, held in Marktoberdorf, Germany (1999).

[15] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the Association for Computing Machinery*, 24(2):84–88, February 1981.

[16] Dorothy E. Denning and Giovanni Maria Sacco. Timestamps in key distribution protocols. *Communications of the ACM*, 24(7):533–535, August 1981.

[17] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. SPKI certificate theory. On the Web at http://www.ietf.cnri.reston.va.us/rfc/rfc2693.txt, September 1999.

[18] Hannes Federrath, Anja Jerichow, and Andreas Pfitzmann. MIXes in mobile communication systems: Location management with privacy. In Ross J. Anderson, editor, *Information hiding: First international workshop*, volume 1174 of *LNCS*, pages 121–135. Springer-Verlag, 1996.

[19] Cédric Fournet and Martín Abadi. Hiding names: Private authentication in the applied pi calculus. In *Software Security – Theories and Systems. Mext-NSF-JSPS International Symposium (ISSS'02)*, volume 2609 of *LNCS*, pages 317–338. Springer-Verlag, 2003.

[20] Alan O. Freier, Philip Karlton, and Paul C. Kocher. The SSL protocol: Version 3.0. Available at http://wp.netscape.com/eng/ssl3/, March 1996.

[21] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, April 1984.

[22] Dominic Hughes and Vitaly Shmatikov. Information hiding, anonymity, and privacy: a modular approach. *Journal of Computer Security*, 2003. To appear.

[23] Markus Jakobsson. *Privacy vs. Authenticity*. PhD thesis, University of California, San Diego, 1997.

[24] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In Ueli Maurer, editor, *Advances in Cryptology—EUROCRYPT 96*, volume 1070 of *LNCS*, pages 143–154. Springer-Verlag, 1996.

[25] Markus Jakobsson and Susanne Wetzel. Security weaknesses in Bluetooth. In *Topics in Cryptology - CT-RSA 2001, Proceedings of the Cryptographer's Track at RSA Conference 2001*, volume 2020 of *LNCS*, pages 176–191. Springer-Verlag, 2001.

[26] Hugo Krawczyk. SKEME: A versatile secure key exchange mechanism for internet. In *Proceedings of the Internet Society Symposium on Network and Distributed Systems Security*, February 1996. Available at http://bilbo.isu.edu/sndss/sndss96.html.

[27] Butler Lampson, Martín Abadi, Michael Burrows, and Edward Wobber. Authentication in distributed systems: Theory and practice. *ACM Transactions on Computer Systems*, 10(4):265–310, November 1992.

[28] Arjen K. Lenstra and Eric R. Verheul. The XTR public key system. In Mihir Bellare, editor, *Advances in Cryptology—CRYPT0 2000*, volume 1880 of *LNCS*, pages 1–19. Springer-Verlag, 2000.

[29] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[30] Refik Molva, Didier Samfat, and Gene Tsudik. Authentication of mobile users. *IEEE Network*, 8(2):26–35, March/April 1994.

[31] Roger M. Needham and Michael D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, December 1978.

[32] Andreas Pfitzmann and Michael Waidner. Networks without user observability. *Computers and Security*, 6(2):158–166, April 1987.

[33] Charles Rackoff and Daniel R. Simon. Cryptographic defense against traffic analysis. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on the Theory of Computing*, pages 672–681, 1993.

[34] Michael G. Reed, Paul F. Syverson, and David M. Goldschlag. Protocols using anonymous connections: Mobile applications. In B. Christianson, B. Crispo, M. Lomas, and M. Roe, editors, *Security Protocols: 5th International Workshop*, volume 1361 of *LNCS*, pages 13–23. Springer-Verlag, 1997.

[35] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *Advances in Cryptology—ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565. Springer-Verlag, 2001.

[36] Didier Samfat, Refik Molva, and N. Asokan. Untraceability in mobile networks. In *Proceedings of the First Annual International Conference on Mobile Computing and Networking (MobiCom 1995)*, pages 26–36, 1995.

[37] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology—CRYPTO 84*, volume 196 of *LNCS*, pages 47–53. Springer-Verlag, 1984.

[38] Vitaly Shmatikov and Dominic Hughes. Defining anonymity and privacy (extended abstract). In *Workshop on Issues in the Theory of Security (WITS' 02)*, January 2002.

[39] Alex C. Snoeren and Hari Balakrishnan. An end-to-end approach to host mobility. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom 2000)*, pages 155–166, 2000.

[40] Yongguang Zhang and Wenke Lee. Intrusion detection in wireless ad-hoc networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, pages 275–283, 2000.

A Appendix: Proofs

This appendix contains the proofs for the results of sections 7 and 8 about the second protocol. It partly relies on definitions and proof techniques for the applied pi calculus [2]. As could be expected, the proofs require the consideration of many details (sometimes abbreviated in this presentation); mechanical support for such proofs may be useful in the future.

We first give a co-inductive proof technique for establishing labelled bisimilarity in the applied pi calculus. Recall that $\sim_l$ is the strong variant of labelled bisimilarity.

We write $\rightarrow_d$ for the subset of $\rightarrow$ that corresponds to term-comparison steps and inputs on filter channels in the protocol—these steps are deterministic and commute with any other step, so they can almost be considered part of structural equivalence in weak bisimulation proofs.

Lemma 9 (Bisimulation proofs up to context, deterministic steps, and strong bisimilarity) *To establish that $\mathcal{R} \subseteq \approx_l$, it suffices to show that $\mathcal{R}$ meets the conditions in the definition of $\approx_l$ (Definition 1) modified as follows: In conditions 2 and 3, instead of $A' \mathcal{R} B'$, we have $A' \rightarrow_d^* \sim_l C[A'']$, $B' \rightarrow_d^* \sim_l C[B'']$, and $A'' \mathcal{R} B''$ for some extended processes A'' and B'', and some evaluation context $C[\_]$.*

The proof is a standard variation of the proof that $\approx_l$ is closed by application of closing evaluation contexts (see [2]).

A.1 State translation

For a given set of compliant principals $\mathcal{C}$, we translate (that is, we compile) each abstract control state to a specific state of the process that implements the session-establishment protocol. We first refine the abstract state and define auxiliary substitutions, then give the translation, and finally state lemmas on the frames that appear in the translation.

We refine the abstract state ρ so that it keeps track of additional transient states for the protocol $\mathcal{P}$. (Intuitively, the attacker can do less in the refined states ρ, so these states need to appear only in transition invariants of the proofs.)

- We supplement ρ with a third map from $B \in \mathcal{C}$ to finite sets of messages F_B already received in R_B (and terms representing those sets). This map is not modified in transitions between processes with control state.
- For each entry t, we introduce another entry $\star t$ to represent the same session state as t but with no subprocess I'_A (typically a state after I'_A received a wrong message). We write $?t$ for t or $\star t$.

In extended processes with control states $\rho : U$, whenever ρ maps i to an entry $?t$ with target B (that is, $t = A\,B$, $t = A\,B\,K_i$, and $t = A\,B-$), and ρ maps B to F_B, we assume that $(x_{1i} \notin F_B)\varphi(U)$—in the translation below, x_{1i} is selectively added to F_B. We also assume that $?A\;B\;K_i$ and $?A\;B-$ occur in ρ only if $A \in S_B$ in ρ and $A \notin S_B$ in ρ, respectively, and that $A\;E$ occurs in ρ only if $E \notin \mathcal{C}$.

We let $\sigma_K^\circ \stackrel{\text{def}}{=} \nu N.\{K = N\}$ and $\sigma_1^\circ \stackrel{\text{def}}{=} \{x_1 = N_A\}$. We use indexed substitutions σ_{1i}, σ_{1i}°, σ_{2i}, σ_{2i}°, σ_{Ki}, and σ_{Ki}° instead of those defined in section 6.4 to represent multiple instances of the substitutions with distinct free names and variables. (For instance, σ_{2i} is σ_2 with defined variable x_{2i} and free nonces N_{Ai}, N_{Bi} instead of x_2 and N_A, N_B.)

We translate $\rho : U$ into the extended process $\mathcal{Q}(\rho : U)$ defined as follows:

$$\begin{aligned}
\mathcal{Q}(\rho : U) &\stackrel{\text{def}}{=} \nu \mathcal{V}_\rho.(U \mid \mathcal{P}(\rho)) \\
\mathcal{P}(\rho) &\stackrel{\text{def}}{=} \textstyle\prod_{A \in \mathcal{C}} \boldsymbol{PK}_A \left[I_A \mid R_A(S_A, F'_A) \mid \prod_{(i \mapsto ?t) \in \rho,\, t = A \ldots} S(i \mapsto ?t) \right] \\
F'_B &\stackrel{\text{def}}{=} F_B \uplus \{ x_{1i} \mid \rho = \rho'[i \mapsto ?A\ B\ (K_i \text{ or } -)] \} \\
S(i \mapsto ?A\ B) &\stackrel{\text{def}}{=} \nu N_{Ai}.(\sigma_{1i} \mid ?I'_{Ai}) \\
S(i \mapsto ?A\ B\ K_i) &\stackrel{\text{def}}{=} \nu N_{Ai}.(\sigma_{1i} \mid ?I'_{Ai} \mid \nu N_{Bi}.(\sigma_{2i} \mid \sigma_{Ki})) \\
S(i \mapsto ?A\ B\ -\ ) &\stackrel{\text{def}}{=} \nu N_{Ai}.(\sigma_{1i} \mid ?I'_{Ai}) \mid \nu N_{Bi}.\sigma^\circ_{2i} \mid \sigma^\circ_{Ki} \\
S(i \mapsto A\ E) &\stackrel{\text{def}}{=} I'_{Ai}\{B = E\}
\end{aligned}$$

where $?I'_{Ai} \stackrel{\text{def}}{=} I'_{Ai}$ when $?$ is nil and $?I'_{Ai} \stackrel{\text{def}}{=} \mathbf{0}$ when $?$ is $\star$, and where $R_A(S_A, F'_A)$ is R_A with sets S_A of acceptable interlocutors and F'_A of messages in the cache (instead of $\emptyset$). In particular, we have $\mathcal{P}(\varepsilon) \equiv \mathcal{P}$ and $\mathcal{Q}(\varepsilon : U) \equiv \mathcal{Q}$ as defined in section 6.5.

The state translation $\mathcal{P}(\rho)$ defines the variables

$$dv(\mathcal{P}(\rho)) = \mathcal{C} \uplus \textstyle\bigcup_{(i \mapsto ?t) \in \rho} \begin{cases} \{x_{1i}\} & \text{when } t = A\ B \\ \{x_{1i}, x_{2ij}, K_i\} & \text{when } t = A\ B\ K_i \text{ or } t = A\ B\ - \\ \emptyset & \text{when } t = A\ E \end{cases}$$

We let $\mathcal{D} \stackrel{\text{def}}{=} \bigcup_\rho dv(\mathcal{P}(\rho))$—this co-infinite set gathers all variables potentially exported in $\mathcal{P}(\rho)$. When we write $\rho : U$, we always assume that the variables in $\mathcal{D} \setminus dv(\mathcal{P}(\rho))$ do not occur in ρ and U.

At each stage of a session between compliant principals A and B, the corresponding frame in the translation is given by $\psi(i \mapsto ?t) \stackrel{\text{def}}{=} S(i \mapsto \star t)$. Except for the indexing on defined variables, $\psi(i \mapsto ABK_i)$ coincides with φ and $\psi(i \mapsto AB-\ )$ coincides with $\varphi^- \mid \nu N.\{K = N\}$ as defined for the theorems of section 7. We also define auxiliary frames for fake messages to B with terms V instead of a nonce and W instead of a public key:

$$\begin{aligned}
\chi(V, W\ B) &\stackrel{\text{def}}{=} \nu N_B.(\{x_2 = \{\mathsf{ack}(V, N_B, B)\}_W\} \mid \{K = \mathsf{h}(V, N_B)\}) \\
\chi^\circ &\stackrel{\text{def}}{=} \nu N_B.\sigma^\circ_2 \mid \sigma^\circ_K
\end{aligned}$$

A.2 Invariant lemma

Next, we systematically write down the protocol states and their transitions, using the distinguished states $\mathcal{P}(\rho)$.

In the lemma below, we rely on the following notation conventions. Equality on terms is to be interpreted in the frame associated with $\mathcal{P}(\rho)$ (so $U = V$ stands for $(U = V)\varphi(\mathcal{P}(\rho))$). When we use structural equivalence to make explicit some restrictions within P', we always assume that the bound names and variables do not clash with $\mathcal{P}(\rho)$ and α.

Lemma 10 *The transitions $\mathcal{P}(\rho) \xrightarrow{\alpha} P'$ are those enumerated below, with the following properties of P'. (We also mention the corresponding transition rules of section 8.1, if any.)*

- $\mathcal{P}(\rho) \xrightarrow{init_A(X)} P'$ *for any $A \in \mathcal{C}$. For each fresh index i, we have subcases depending on X:*

(1) $P' \equiv \nu x_{1i}.(\overline{c}_1\langle x_{1i}\rangle \mid \mathcal{P}(\rho[i \mapsto A\ B]))$ *if* $X = B \in \mathcal{C}$.

(2) $P' \equiv \nu x_{1i}.(\overline{c}_1\langle x_{1i}\rangle \mid \nu N_{Ai}.(\sigma_{1i}\{B = E\} \mid \mathcal{P}(\rho[i \mapsto A\ E])))$ *if* $X = E \notin \mathcal{C}$.

(These cases correspond to special transitions INIT *and* INIT-E.*)*

- $\mathcal{P}(\rho) \xrightarrow{c_1(X_1)} P'$. *For each $B \in \mathcal{C}$, we have subcases depending on ρ and X_1:*

(3) If $\rho = \rho'[i \mapsto ?A\ B]$ *and* $X_1 = x_{1i}$ *for some i, then we have*

$$P' \rightarrow_d^+ \nu x_{2i} K_i.\left(\overline{c}_2\langle x_{2i}\rangle \mid \begin{cases} \mathcal{P}(\rho'[i \mapsto ?A\ B\ K_i]) | \overline{accept}_B\langle A, K_i\rangle & \text{if } A \in S_B \\ \mathcal{P}(\rho'[i \mapsto ?A\ B -]) & \text{if } A \notin S_B \end{cases}\right)$$

For the other subcases, let ρ' be ρ with X_1 added to F_B.

(4) If $X_1 \notin F'_B$ *and* $X_1 = \{\mathsf{hello}(V, W)\}_B$ *for some terms V and W with $W \in S_B$, we have* $P' \rightarrow_d^+ \mathcal{P}(\rho') \mid \nu x_2, K.(\overline{c}_2\langle x_2\rangle \mid \overline{accept}_B\langle V, K\rangle \mid \chi(V, W\ B))$.

(5) Otherwise, we have $P' \rightarrow_d^+ \mathcal{P}(\rho') \mid \nu N_B.\overline{c}_2\langle x_2\sigma_2^\circ\rangle$.

(These transitions do not depend on "?", and always add X_1 to B's filter. Case 3 corresponds to the two branches of ACCEPT. *Case 4 covers both the first branch of* ACCEPT-FAKE *and* ACCEPT-E. *Case 5 covers the second branch of* ACCEPT-FAKE.*)*

- $\mathcal{P}(\rho) \xrightarrow{c_2(X_2)} P'$. *For each i such that $\rho = \rho'[i \mapsto t]$, we have subcases depending on t and X_2:*

(6) $t = A\ B\ K_i$, $X_2 = x_{2i}$, *and we have* $P' \rightarrow_d \mathcal{P}(\rho'[i \mapsto \star t]) \mid \overline{connect}_A\langle B, K_i\rangle$; $t = A\ B - $, $X_2 = x_{2i}$, *and we have* $P' \rightarrow_d \mathcal{P}(\rho'[i \mapsto \star t])$.

(7) $t = A\ B$ *or, for any* $X_2 \neq x_{2i}$, $t = A\ B\ K_i$ *or* $t = A\ B - $. *Then, we have* $P' \rightarrow_d \mathcal{P}(\rho'[i \mapsto \star t])$.

(8) $t = A\ E$, $X_2 = \{\mathsf{ack}(N_{Ai}, V, E)\}_A$ *for some term V. Then, we have* $P' \rightarrow_d \mathcal{P}(\rho') \mid \overline{connect}_A\langle E, \mathsf{h}(N_{Ai}, V)\rangle$.

(9) $t = A\ E$ *for any other X_2, and we have* $P' \rightarrow_d \mathcal{P}(\rho')$.

(Cases 6 and 8 correspond to rules CONNECT *and* CONNECT-E.*)*

Proof: The proof follows from our definition of translated states, and is by case analysis on the input prefixes in $\mathcal{P}(\rho)$. ($\mathcal{P}(\rho)$ has neither internal steps nor outputs.) We detail the following cases:

- $\mathcal{P}(\rho) \xrightarrow{\mathit{init}_A(X)} P'$ is a replicated input of I_A.
- $\mathcal{P}(\rho) \xrightarrow{c_1(X_1)} P'$ is a replicated input of R_B for some $B \in \mathcal{C}$. The deterministic steps $\rightarrow_d^+$ consist of a communication on the local channel c to read F'_B followed by a series of tests on X_1 in R_B: a test for freshness $X_1 \notin F'_B$, one for pattern matching $X_1 = \{\mathsf{hello}(N_A, A)\}_B$ in a context that defines $\{N_A = \mathsf{hello.0}(\mathsf{decrypt}(X_1, K_B^{-1}))\}$ and $\{A = \mathsf{hello.1}(\mathsf{decrypt}(X_1, K_B^{-1}))\}$, and one for authorization $A \in S_B$.
  3. The freshness test succeeds by hypothesis $x_{1i} \notin F'_B$ when $i \mapsto ?A\ B$ (with X_1 added to F'_B in the resulting state). By equational rewriting and structural equivalence, the pattern matching succeeds with A bound in $\mathit{PK}_A\,[\_]$ and $N_A = N_{Ai}$ bound in $S(i \mapsto ?A\ B)$. If $A \in B$, the resulting subprocess is:

$$\nu x_{2i} K_i.(\bar{c}_2\langle x_{2i}\rangle \mid \nu N_{Bi}.(\sigma_{2i} \mid \sigma_{Ki} \mid \overline{\mathit{accept}}_B\langle A, K_i\rangle))$$

  If $A \notin B$, we use $\mathbf{0} \equiv \nu K_i.\sigma^\circ_{Ki}$ and obtain by structural equivalence

$$\nu x_{2i} K_i.(\bar{c}_2\langle x_{2i}\rangle \mid \nu N_{Bi}.\sigma^\circ_{2i} \mid \sigma^\circ_{Ki})$$

  In any case, we rely on the hypothesis on $\mathcal{D}$ and structural equivalence to lift the restriction on x_{2i} and K_i to the top level of the translation.
  4. The three tests succeed, each using a hypothesis in the case definition, with X_1 added to F_B in the resulting state. (The hypothesis $X_1 \notin F'_B$ implies $X_1 \neq x_{1i}$ for any i with target B in the domain of ρ.)
  5. If the freshness test fails, then $\rho = \rho'$. Otherwise, $X_1 \neq x_{1i}$ for any i with target B in the domain of ρ, and X_1 is added to F_B. In any case, a fresh decoy message is sent.
- $\mathcal{P}(\rho) \xrightarrow{c_2(X_2)} P'$ is a single input in a subprocess I'_{Ai} of $\mathcal{P}(\rho)$, which corresponds to some $A \in \mathcal{C}$ and entry $i \mapsto t$ in ρ (not $i \mapsto \star t$). After the test, I'_{Ai} is replaced with either a message on $\mathit{connect}_A$ or the null process $\mathbf{0}$ and we conclude by structural equivalence.

  We detail the test in the pattern matching of I'_A in the cases $i \mapsto A\ B\ K_i$ and $i \mapsto A\ B\ -$ with two subcases depending on $A \in S_B$. (The cases $i \mapsto A\ B$ and $i \mapsto A\ E$ are similar.) Since N_{Ai} and K_A^{-1} are bound in $\mathcal{P}(\rho)$, we can assume that they do not syntactically occur in X_2. Let X be a term such that $\mathit{fn}(X) \cap \{N_{Ai}, K_A^{-1}\} = \emptyset$, and V and W be any terms. We have:
  - $X\left(\sigma_{1i} \mid \sigma_{2i} \mid \sigma_{Ki} \mid \{A = \mathsf{pk}(K_A^{-1})\}\right) = \{\mathsf{ack}(N_{Ai}, V, W)\}_A$ succeeds if and only if $X = x_{2i}$.
  - $X\left(\sigma_{1i} \mid \sigma^\circ_{2i} \mid \sigma^\circ_{Ki}) \mid \{A = \mathsf{pk}(K_A^{-1})\}\right) = \{\mathsf{ack}(N_{Ai}, V, W)\}_A$ always fails.
  6. The test in I'_A succeeds or fails according to t, as detailed above. When the test succeeds, we rely on structural equivalence and the active substitution σ_{Ki} in $\psi(i \mapsto A\ B\ K_i)$ to replace the key computation triggered in I'_{Ai} by the defined variable K_i.
  7. The test fails and yields $\mathbf{0} = \star I'_{Ai}$.
  8. The test succeeds and yields a connect message.
  9. The test fails and yields $\mathbf{0}$. □

The next lemmas relate frames appearing in the protocol implementation; they crucially rely on which-key concealment.

Lemma 11 *We have the following static equivalences:*

$$\mathit{PK}_A\,[\mathbf{0}] \mid \chi(V, A\ B) \approx_s \mathit{PK}_A\,[\mathbf{0}] \mid \chi^\circ \tag{A.1}$$
$$\mathit{PK}_B\,[\mathbf{0}] \mid \nu N_{Ai}.\sigma_{1i} \approx_s \mathit{PK}_B\,[\mathbf{0}] \mid \nu N_{Ai}.\sigma^\circ_{1i} \tag{A.2}$$
$$\begin{aligned} &\mathit{PK}_A\,[\mathbf{0}] \mid \mathit{PK}_B\,[\mathbf{0}] \mid \psi(i \mapsto A\ B\ K_i) \\ \approx_s\ &\mathit{PK}_A\,[\mathbf{0}] \mid \mathit{PK}_B\,[\mathbf{0}] \mid \psi(i \mapsto A\ B\ -\ ) \end{aligned} \tag{A.3}$$

Proof: Within our equational theory, we check that, for all terms with free variables in the domain of the related frames, the substituted terms are "equationally inert", that is, do not enable any additional rewrite step.

Equivalence A.1 is an instance of:

$$\nu s, N_B.\begin{pmatrix} \{A = \mathsf{pk}(s)\} \mid \\ \{x_2 = \{\mathsf{ack}(V, N_B, W)\}_{\mathsf{pk}(s)}\} \mid \\ \{K = \mathsf{h}(V, N_B)\} \end{pmatrix} \approx_s \nu s, M, N.\begin{pmatrix} \{A = \mathsf{pk}(s)\} \mid \\ \{x_2 = M\} \mid \\ \{K = N\} \end{pmatrix}$$

where V and W range over arbitrary terms (up to $\equiv$ and supposing $s \notin \mathit{fn}(V, W)$). Consider two terms V_1, V_2 with $\mathit{fv}(V_i) \subseteq \{A, x_2, K\}$ and $\mathit{fn}(V_i) \cap \{s, N_B, M, N\} = \emptyset$. Let σ and σ° be the two plain substitutions obtained from the frames above by discarding restrictions. We show that $V_1\sigma = V_2\sigma$ iff $V_1\sigma^\circ = V_2\sigma^\circ$ by structural induction on V_1 and V_2. For each axiom in the equational theory, we check the correspondence of rewrite steps after applying either substitution: as regards x_2, for instance, the rule for decryption does not apply to x_2 because the key term is not equal to s; the rule for field selection does not apply to x_2 because the encrypted message is not a plain message constructor.

Equivalence A.3 is obtained from equivalence A.1 (with A and B swapped) by indexing K and x_2 with i and applying the context $\mathit{PK}_A\,[\mathbf{0}] \mid \nu N_{Ai}.(\sigma_{1i} \mid \_)$.

Equivalence A.2 follows from a more general static equivalence:

$$\nu s, N_A.\begin{pmatrix} \{B = \mathsf{pk}(s)\} \mid \\ \{x_{1i} = \{\mathsf{hello}(N_A, V)\}_{\mathsf{pk}(s)}\} \end{pmatrix} \approx_s \nu s, M.\begin{pmatrix} \{B = \mathsf{pk}(s)\} \mid \\ \{x_{1i} = M\} \end{pmatrix}$$

where V is an arbitrary term, with a similar proof. □

By composing these static equivalences in evaluation contexts, we obtain that the frame associated with any state of the protocol is equivalent to a frame that defines all its variables as distinct fresh names, and is thus equationally inert:

Lemma 12 $\varphi(\mathcal{P}(\rho)) \approx_s \prod_{x \in dv(\mathcal{P}(\rho))} \nu N.\{x = N\}$.

Proof: For a given ρ, let $Is = \{i \mid (i \mapsto ?A\ B\ s_i) \in \rho\}$ where s is nil, K, or $-$. We have:

$$\begin{aligned}
&\mathcal{P}(\rho) \stackrel{\text{def}}{=} \prod_{A \in \mathcal{C}} PK_A \left[\mathcal{P}_A \mid \prod_{(i \mapsto ?t) \in \rho,\, t = A\ldots} S(i \mapsto ?t)\right] \\
\approx_s &\prod_{A \in \mathcal{C}} PK_A\,[\mathbf{0}] \mid \prod_{I} \psi(i \mapsto ?A\ B) \mid \prod_{IK} \psi(i \mapsto ?A\ B\ K_i) \mid \prod_{I-} \psi(i \mapsto ?A\ B\ -) && \text{(A.4)} \\
\approx_s &\prod_{A \in \mathcal{C}} PK_A\,[\mathbf{0}] \mid \prod_{I} \psi(i \mapsto A\ B) \mid \prod_{IK \cup I-} \psi(i \mapsto A\ B\ -) && \text{(A.5)} \\
\equiv &\prod_{A \in \mathcal{C}} PK_A\,[\mathbf{0}] \mid \prod_{I \cup IK \cup I-} \nu N_{Ai}.\sigma_{1i} \mid \prod_{IK \cup I-} (\nu N_{Bi}.\sigma^{\circ}_{2i} \mid \sigma^{\circ}_{Ki}) \\
\approx_s &\prod_{A \in \mathcal{C}} PK_A\,[\mathbf{0}] \mid \prod_{I \cup IK \cup I-} \nu N_{Ai}.\sigma^{\circ}_{1i} \mid \prod_{IK \cup I-} (\nu N_{Bi}.\sigma^{\circ}_{2i} \mid \sigma^{\circ}_{Ki}) && \text{(A.6)} \\
\approx_s &\prod_{A \in \mathcal{C}} \nu N.\{A = N\} \mid \prod_{I \cup IK \cup I-} \nu N_{Ai}.\sigma^{\circ}_{1i} \mid \prod_{IK \cup I-} (\nu N_{Bi}.\sigma^{\circ}_{2i} \mid \sigma^{\circ}_{Ki}) && \text{(A.7)} \\
\equiv &\prod_{x \in dv(\mathcal{P}(\rho))} \nu N.\{x = N\} && \text{(A.8)}
\end{aligned}$$

where (A.4) is obtained by erasure of plain subprocesses $\mathcal{P}_A$ and I'_A followed by structural equivalence (since the decryption key does not occur anywhere except in its definition); (A.5) follows from Lemma 11(A.3) for each $i \in IK$; (A.6) follows from Lemma 11(A.2) for each $i \in I \cup IK \cup I-$; (A.7) follows from $\nu s.\{A = \mathsf{pk}(s)\} \approx_s \nu N.\{A = N\}$. (A.8) is a renaming of bound names. □

From Lemma 12, we obtain that, for any compliant user processes U and U' and any label α such that $bv(\alpha) \cap \mathcal{D} = \emptyset$, we have:

(1) if $U \xrightarrow{\alpha} U'$ considering the variables $dv(\mathcal{P}(\rho))$ as distinct fresh names, then $U \mid \mathcal{P}(\rho) \xrightarrow{\alpha} U' \mid \mathcal{P}(\rho)$.
(2) if $U \mid \varphi(\mathcal{P}(\rho)) \xrightarrow{\alpha} U'' \mid \varphi(\mathcal{P}(\rho))$, then $U \xrightarrow{\alpha} U'$ considering the variables $dv(\mathcal{P}(\rho))$ as distinct fresh names, with $U' \mid \varphi(\mathcal{P}(\rho)) \equiv U'' \mid \varphi(\mathcal{P}(\rho))$.

The next lemma lifts Lemma 11 from frames to translated states:

Lemma 13 *For any extended control state ρ and $A, B \in \mathcal{C}$, we have:*

$$\mathcal{P}(\rho) \mid \chi(V, A\ B) \sim_l \mathcal{P}(\rho) \mid \chi^{\circ} \qquad \text{(A.9)}$$

Let $\rho = \rho'[i \mapsto A\ B\ K_i]$ and ρ_{1i} be ρ' with x_{1i} added to F_B. We have:

$$\mathcal{P}(\rho_{1i}) \mid \nu N_A.\sigma_{1i} \approx_l \mathcal{P}(\rho') \mid \nu N_A.\sigma^\circ_{1i} \tag{A.10}$$
$$\mathcal{P}(\rho_{1i}) \mid \psi(i \mapsto A\ B\ K_i) \approx_l \mathcal{P}(\rho_{1i}) \mid \psi(i \mapsto A\ B\ -\ ) \tag{A.11}$$

Proof:

(A.9): Let χ abbreviate $\chi(V, A\ B)$. By definition, we have:

$$\chi \stackrel{\text{def}}{=} \nu N_B.(\{x_2 = \{\mathsf{ack}(V, N_B, B)\}_A\} \mid \{K = \mathsf{h}(V, N_B)\})$$
$$\chi^\circ \stackrel{\text{def}}{=} \nu N.\{x_2 = N\} \mid \nu N.\{K = N\}$$

For a given χ, let $\mathcal{R}$ be the relation that contains $(A.9)$ for all ρ. We show that $\mathcal{R}$ is a strong bisimulation up to context and conclude using (a strong variant of) Lemma 9.

The static equivalence requirement is Lemma 11(A.1) in context. The strong bisimulation requirements are easily established using the case analysis of Lemma 10: in each case, it suffices to check that all tests in $\mathcal{P}(\rho)$ yield the same results when placed in parallel with χ and with χ°.

We detail the cases 6–9 of Lemma 10 when $\rho = \rho'[i \mapsto t]$, which cover all transitions leading to a decryption attempt of x_2 with a decryption key that matches the encryption key A used in χ. (For all other transitions, the static equivalence of 11(A.1) suffices to conclude.) In the frames of $\mathcal{P}(\rho) \mid \chi$ and $\mathcal{P}(\rho) \mid \chi^\circ$, we have $x_2 \neq x_{2i}$ by Lemma 12, and thus case 6 is excluded. Similarly, in both frames, $x_2 \neq \{\mathsf{ack}(N_A, V, E)\}_A$ for any $E \notin \mathcal{C}$, since $x_2\chi^\circ$ is not an encrypted message and $x_2\chi$ has a third field $B \neq E$, and thus case 8 is excluded. In case 7, we obtain processes related by $\mathcal{R}$ for the control state $\rho'[i \mapsto \star t]$. In case 9, we obtain processes related by $\mathcal{R}$ for the control state ρ'.

(A.10): The proof similarly relies on Lemmas 9, 11, and 10. We use a candidate relation $\mathcal{R}$ that contains all pairs

$$\mathcal{P}(\rho_{1i}) \mid \nu N_A.\sigma_{1i}\ \mathcal{R}\ \mathcal{P}(\rho') \mid \nu N_A.\sigma^\circ_{1i} \tag{A.12}$$
$$\mathcal{P}(\rho_{1i}) \mid \nu N_A.\sigma_{1i}\ \mathcal{R}\ \mathcal{P}(\rho_{1i}) \mid \nu N_A.\sigma^\circ_{1i} \tag{A.13}$$

The cases 3–5 of Lemma 10 cover all potential decryption attempts of x_{1i} as a "hello" message with decryption key K_B^{-1}. In $\mathcal{P}(\rho') \mid \nu N_A.\sigma^\circ_{1i}$, the message x_{1i} passes the freshness test but fails the pattern matching (the message is a nonce, not an encrypted message), a decoy is generated, and x_{1i} is added to F_B. For all other processes related by $\mathcal{R}$, we have $x_{1i} \in F'_B$, so the message fails the freshness test, a decoy is generated, and the protocol state is left unchanged. Thus, we are always in case 5 and obtain on each side the processes related on line (A.13) in the evaluation context $[\_] \mid \nu N_B.\bar{c}_2\langle x_2\sigma^\circ_2\rangle$. Other transitions are handled using Lemma 11(A.2).

(A.11): Similarly, $x_{1i} \in F_B$ always excludes the decryption of x_{1i}, and the test in pattern matching of cases 6 and 8 always fails on x_{2i}, either because the first nonce is different from the expected one or because the message is not encrypted under A. Other transitions are handled using Lemma 11(A.3). □

A.4 Proofs of section 7

While its statement is optimized for the proof of Lemma 5, Lemma 10 also provides precise syntactic support for establishing the theorems of section 7. We first relate the results of arbitrary transitions of $\mathcal{P}$ to state translations in context:

Lemma 14 *If $\mathcal{P} \xrightarrow{\eta} P'$, then $P' \rightarrow_d^* \equiv C[\mathcal{P}(\rho)]$ for some control state ρ and evaluation context $C[\_]$, where $C[\_]$ is obtained by composing the evaluation contexts appearing in Lemma 10 and deleting their messages as they are consumed by output transitions and internal communication steps on c_1 and c_2.*

Proof: By induction on η, definition of (ordinary) labelled transitions, Lemma 10, and subcommutation of $\rightarrow_d$ with any other transition: if $P_1 \xrightarrow{\eta} P'$ and $P_1 \rightarrow_d^* \equiv C[\mathcal{P}(\rho)]$, then for some P'' and η' obtained from η by deleting $\rightarrow_d$-steps, we have $P' \rightarrow_d^* \equiv P''$ and $C[\mathcal{P}(\rho)] \xrightarrow{\eta'} \equiv P''$. The transition label (or, in case of an internal communication on channels c_1 or c_2, the message consumed in $C[\_]$) and the input prefix in $\mathcal{P}(\rho)$ determine the case in Lemma 10. □

The next theorem corresponds to the discussion before Theorem 2; it uses the same notation conventions.

Theorem 15 (Complete runs) *Let $A, B \in \mathcal{C}$.*

(1) (Success:) If $\mathcal{P} \xrightarrow{\eta} P'$ and $A \in S_B$, then $P' \xrightarrow{\omega} P'_{x_1} \mid \varphi$.
(Failure:) If $\mathcal{P} \xrightarrow{\eta} P'$ and $A \notin S_B$, then $P' \xrightarrow{\omega^-} P'_{x_1} \mid \varphi^-$.
(2) Conversely, if $\mathcal{P} \xrightarrow{\omega} P''$, then $A \in S_B$ and $P'' \equiv \mathcal{P}_{x_1} \mid \varphi$.

Proof of Theorem 15: We first apply Lemma 14 to obtain $P' \rightarrow_d^* \equiv C[\mathcal{P}(\rho)]$. From the translation state $\mathcal{P}(\rho)$, we exhibit a particular trace $\xrightarrow{\omega}$ (or $\xrightarrow{\omega^-}$), up to α-conversion to erase indices in bound variables in the trace and avoid clashes with $C[\_]$. We then check that $C[\mathcal{P}(\rho)]$ (by construction) and finally P' (by commutation of each $\rightarrow_d$ step occurring in $P' \rightarrow_d \equiv C[\mathcal{P}(\rho)]$ with $\xrightarrow{\omega}$) have the same trace.

The trace is obtained by composing the following transitions:

- transition 1 of Lemma 10 for some index i fresh in ρ, leading to $\mathcal{P}(\rho[i \mapsto A\ B])$ in evaluation context $\nu x_{1i}.(\overline{c}_1\langle x_{1i}\rangle \mid \_)$;
- $\xrightarrow{\nu x_{1i}.\overline{c}_1\langle x_{1i}\rangle}$ that discards this evaluation context;
- transition 3 with $X_1 = x_{1i}$, leading (after $\rightarrow_d^+$) to $\mathcal{P}(\rho[i \mapsto A\,B\,K_i])$ in evaluation context $\nu x_{2i} K_i.(\overline{c}_2\langle x_{2i}\rangle \mid \_ \mid \overline{\mathit{accept}}_B\langle A, K_i\rangle)$;
- $\xrightarrow{\nu x_{2i}.\overline{c}_2\langle x_{2i}\rangle}$ that discards the evaluation context $\nu x_{2i}.(\overline{c}_2\langle x_{2i}\rangle \mid \_)$ and leaves $\mathcal{P}(\rho[i \mapsto A\ B\ K_i])$ in evaluation context $\nu K_i.(\_ \mid \overline{\mathit{accept}}_B\langle A, K_i\rangle)$;
- transition 6 for $i \mapsto A\ B\ K_i$ leading (after $\rightarrow_d^+$) to $\mathcal{P}(\rho[i \mapsto \star A\ B\ K_i])$ in evaluation context $\nu K_i.(\_ \mid \overline{\mathit{accept}}_B\langle A, K_i\rangle \mid \overline{\mathit{connect}}_A\langle B, K_i\rangle)$;
- $\xrightarrow{\nu K.\overline{\mathit{accept}}_B\langle A,K\rangle}\xrightarrow{\overline{\mathit{connect}}_A\langle B,K\rangle}$ (after α-converting K_i to K) that discard the evaluation context given above and leave just $\mathcal{P}(\rho[i \mapsto i \mapsto \star A\ B\ K_i]) = \mathcal{P}(\rho)_{x_1} \mid \varphi$;

and, when $A \notin B$, similar initial five transitions leading to

$$\nu K_i.(\mathcal{P}(\rho[i \mapsto \star A\ B\ -\ ]) \equiv \mathcal{P}(\rho)_{x_1} \mid \varphi^-$$

To prove the second part of the theorem, we apply Lemma 14 for the labels ω and check that, after each labelled transition, there is a unique reachable state translation up to $\equiv$ that enables the rest of the trace. □

Proof of Theorem 2: It suffices to relate the processes obtained by Theorem 15 and Lemma 14 to those appearing in the statement of Theorem 2, that is, to show that

$$\begin{aligned} C[\mathcal{P}(\rho)_{x_1}] \mid \varphi &\approx_l C[\mathcal{P}(\rho)] \mid \varphi^\circ \mid \nu N.\{K = N\} \\ &\approx_l C[\mathcal{P}(\rho)_{x_1}] \mid \varphi^- \mid \nu N.\{K = N\} \end{aligned}$$

Moreover, for some evaluation context $C'[\_]$, we have

$$C[\mathcal{P}(\rho)_{x_1}] \mid \varphi \equiv C'[\mathcal{P}(\rho)_{x_1} \mid \varphi]$$

and similarly for the other frames. Since $\approx_l$ is closed by application of evaluation contexts, it suffices to show that

$$\begin{aligned} P_1 &= \mathcal{P}(\rho)_{x_1} \mid \varphi \\ \approx_l P_2 &= \mathcal{P}(\rho) \mid \varphi^\circ \mid \nu N.\{K = N\} \\ \approx_l P_3 &= \mathcal{P}(\rho)_{x_1} \mid \varphi^- \mid \nu N.\{K = N\} \end{aligned}$$

Finally, $P_1 \approx_l P_3$ is Lemma 13(A.11) and $P_2 \approx_l P_3$ is Lemma 13(A.10) in evaluation context $[\_] \mid \nu N_B.\{x_2 = N_B\} \mid \nu N.\{K = N\}$. □

Proof of corollary after Theorem 2: For all processes A, we have that $A \xrightarrow{a(V)} A'$ implies $A \mid \overline{a}\langle V\rangle \rightarrow A'$ and (for asynchronous outputs) $A \xrightarrow{\nu x.\overline{a}\langle x\rangle} A'$ implies

$A \equiv \nu x.(A' \mid \overline{a}\langle x\rangle)$. We apply Theorem 2 (Success), then use these remarks and the context-closure property of $\approx_l$ for the evaluation context $\nu K.(\overline{\mathit{connect}}_A\langle B, K\rangle \mid \overline{\mathit{accept}}_B\langle A, K\rangle \mid \_)$. We finally discard $\nu N.\{K = N\}$ by structural equivalence. □

Proof of Theorem 3: We apply Lemma 14 to obtain $\mathcal{P}(\varepsilon) \xrightarrow{\eta} \rightarrow_d^* \equiv C[\mathcal{P}(\rho)]$ and use the case analysis of Lemma 10. Starting from the first transition $\xrightarrow{\overline{\mathit{connect}}_A\langle B,W\rangle}$ that occurs in η (for any term W), and going backwards, we successively identify preliminary input transitions that must appear in the trace and correspond to the first branch of case 6, case 3 with $A \in S_B$, and case 1 of Lemma 10. Hypothesis (1) in the theorem guarantees that no input transition described in Lemma 10 depends on $C[\_]$.

- This first *connect* transition commutes with any preceding transition (as given by Lemma 10) that does not introduce the message $\overline{\mathit{connect}}_A\langle B, W\rangle$ in $C[\_]$. The only transitions that introduce such message are described in Lemma 10, case 6, and enabled only by an input of x_{2i} for some index i with state $t = A\ B\ K$.
- For this index i, we identify the two other input transitions in η that yield the states $t = A\ B$ and $t = A\ B\ K$ at index i.
- The outputs of the messages x_{2i} and x_{1i} introduced by these transitions necessarily precede their input in η.
- Hypothesis (2) ensures that the message $\overline{\mathit{accept}}_B\langle A, K\rangle$ introduced by case 3 with $A \in S_B$ yields an output transition $\xrightarrow{\overline{\mathit{accept}}_B\langle A,K\rangle}$ in η.

Once we have identified ω as a subtrace of η, we easily check that each of these transitions commute with any other preceding transition in η, using again Lemma 10. □

A.5 Proofs of section 8.2

We first refine our notion of private bisimilarity to deal with refined control states and give some basic properties as regards failed sessions, then we prove (a generalization of) our main result.

So far, private bisimilarity is defined only for processes with control states as defined in section 8.2. The next lemma relates the control states of bisimilar processes:

Lemma 16 (Related control states) *If $\rho_1 : U_1 \approx_{\mathcal{C}} \rho_2 : U_2$, then ρ_1 and ρ_2 have identical domain, and yield session states t_z of the same kind: either both $A_z\ B_z$, or both $A_z\ B_z\ K_i$ or $A_z\ B_z\ -$, or both $A\ E$ with the same $A \in \mathcal{C}$ and $E \notin \mathcal{C}$.*

Proof: This property follows from the definition of transitions that operate on ρ_z. We apply the simulation hypothesis of private bisimilarity (Definition 4(3)) to specific transitions $\xrightarrow{\gamma}$ that characterize the structure of ρ. For instance, for any index i, we have $\rho : U \xrightarrow{\textit{accept } i} \rho' : U'$ if and only if $(i \mapsto A\ B) \in \rho$ for some $A, B \in \mathcal{C}$. □

We now extend our definitions of labelled transitions and private bisimilarity to user processes with refined control state.

- $T \xrightarrow{\gamma} T'$ is defined as in section 8.1 (and leaves the sets F_B unchanged), except that rule ACCEPT is extended to operate on failed states $\star t$:

  ACCEPT

  $$\rho[i \mapsto ?\ A\ B] : U \xrightarrow{\textit{accept } i} \begin{cases} \rho[i \mapsto ?\ A\ B\ K_i] : U \mid \overline{\textit{accept}}_B\langle A, K_i\rangle & \text{if } A \in S_B \\ \rho[i \mapsto ?\ A\ B\ -\ ] : U & \text{if } A \notin S_B \end{cases}$$

  Conversely, "initiator" rules CONNECT and CONNECT-E are defined as before, and do not operate on entries $\star t$.
- $\rho_1 : U_1 \approx_{\mathcal{C}} \rho_2 : U_2$ is defined as in Definition 4 with two additional requirements:
  4. For all $B \in \mathcal{C}$, the sets F_B in ρ_1 and ρ_2 are syntactically identical.
  5. ρ_1 and ρ_2 have identical domain and yield entries of the same kind (as defined in Lemma 16) with $\star$ at the same indices.

User processes with unrefined control states are closed under transitions, so our extension of $\approx_{\mathcal{C}}$ coincides with Definition 4 for such processes.

The next lemma describes how to change parts of the refined control state while preserving private bisimilarity. These changes will be convenient to reflect changes in the state of the protocol translation.

Lemma 17 (Control changes) *For all well-formed extended processes with control state, we have:*

(1) For some $B \in \mathcal{C}$ and $z = 1, 2$, let T_{zx} be T_z with the same term X_1 added to F_B. If $T_1 \approx_{\mathcal{C}} T_2$, then $T_{1x} \approx_{\mathcal{C}} T_{2x}$.

(2) Let t_1 and t_2 be control states of the kind $A\ B\ K_i$ or $A\ B\ -$.
If $\rho_1 : \nu K_i.U_1 \approx_{\mathcal{C}} \rho_2 : \nu K_i.U_2$, then $\rho_1[i \mapsto \star t_1] : U_1 \approx_{\mathcal{C}} \rho_2[i \mapsto \star t_2] : U_2$.
If $\rho_1[i \mapsto t_1] : U_1 \approx_{\mathcal{C}} \rho_2[i \mapsto t_2] : U_2$, then $\rho_1[i \mapsto \star t_1] : U_1 \approx_{\mathcal{C}} \rho_2[i \mapsto \star t_2] : U_2$.

(3) If $\rho_1[i \mapsto A\ E] : U_1 \approx_{\mathcal{C}} \rho_2[i \mapsto A\ E] : U_2$, then $\rho_1 : U_1 \approx_{\mathcal{C}} \rho_2 : U_2$.

Proof: For each private bisimilarity claim in the lemma, we easily show that the relation $\mathcal{R}$ containing all processes that meet the hypothesis is a private bisimulation, up to an injective re-indexing on the domain of ρ_1 and ρ_2 for the proof of 3.

(1) Our transitions are independent of F_B.
(2) Conditions 1 in Definition 4 is structurally equivalent to condition 1 for both private bisimilarity hypotheses. Conditions 2 and 3 follow from the direct correspondence between the transitions of $\rho[i \mapsto \star t] : U$ and those of $\rho : \nu K_i.U$ and $\rho[i \mapsto t] : U$ (although the latter processes have additional labelled transitions).
(3) The proof is immediate, except for transitions with label νi that "reuse" the index of the discarded session (rules INIT and INIT-E). For those transition, we choose another fresh index i' and conclude up to injective re-indexing after the transitions. □

Next, we relate transitions of translated protocol configurations to those of user processes. In the lemma, we write $C_z[\_]$ for the evaluation context around $\mathcal{P}(\_)$ in Lemma 10(z).

Lemma 18 *Let $T = \rho : U$. If $\mathcal{Q}(T) \xrightarrow{\alpha} Q'$ with $fn(\alpha) \cap \mathcal{V}_\rho = \emptyset$ and $bn(\alpha) \cap (\mathcal{C} \cup \mathcal{V}_\rho)$, then one of the following holds:*

(1) $U \xrightarrow{\alpha} U'$, with $T \xrightarrow{\alpha} T' = \rho : U'$ and $Q' \equiv \mathcal{Q}(T')$.
(2) $\alpha = \tau$ and $\mathcal{P}(\rho)$ receives a message on $init_A$ for some $A \in \mathcal{C}$, with two subcases:
  (a) $U \xrightarrow{\overline{init}_A\langle B\rangle} U'$ for some $B \in \mathcal{C}$ and, for any fresh index i,
  $T \xrightarrow{\overline{init}\,\nu i} T' = \rho[i \mapsto A\ B] : U'$ and $Q' \equiv C_1[\mathcal{Q}(T')]$.
  (b) $U \xrightarrow{\nu E.\overline{init}_A\langle E\rangle} U'$ and, for any fresh index i,
  $T \xrightarrow{\nu i E.\overline{init}_A\langle i,E\rangle} T' = \rho[i \mapsto A\ E] : U'$ and $Q' \equiv \nu E.C_2[\mathcal{Q}(T')]$.
(3) $\alpha = \tau$ and $\mathcal{P}(\rho)$ receives a message on c_1 or c_2.
(4) α is an input on c_1 with $\varphi(U) \mid \mathcal{P}(\rho) \xrightarrow{\alpha} P' \mid \varphi(U)$ and $Q' \equiv \nu\mathcal{V}_\rho.(U \mid P')$.
(5) α is an input on c_2 with $\varphi(U) \mid \mathcal{P}(\rho) \xrightarrow{\alpha} P' \mid \varphi(U)$ and $Q' \equiv \nu\mathcal{V}_\rho.(U \mid P')$.

Proof: By definition of (ordinary) transitions and Lemma 10, $\mathcal{P}(\rho)$ can at most input on control channels (when U outputs on those channels) and network channels c_1 and c_2 (when either U or the environment output on those channels). For all other transitions, we also have $U \mid \varphi(\mathcal{P}(\rho)) \xrightarrow{\alpha} U'' \mid \varphi(\mathcal{P}(\rho))$. By Lemma 12, this implies $U \xrightarrow{\alpha} U'$ (treating variables defined in $\varphi(\mathcal{P}(\rho))$ as distinct names) for some U' such that $Q' \equiv U' \mid \mathcal{P}(\rho)$.

Case 2 of the lemma details an input on channel $init_A$ for some $A \in \mathcal{C}$, corresponding to an output in U. For any such output, we can introduce a fresh variable, E, use structural equivalence to introduce an active substitution that defines E, and write the output $U \xrightarrow{\nu E.\overline{init}_A\langle E\rangle} U'$. There are two subcases:

- If $(E = B)\varphi(U')$ for some $B \in \mathcal{C}$, then we also have the free variable output

$U \xrightarrow{\overline{init}_A\langle B\rangle} U''$. Let $\rho' = \rho[i \mapsto A\ B]$ for some fresh i. By rule INIT, we have $\rho : U \xrightarrow{\overline{init}\,\nu i} \rho' : U''$. Using Lemma 10(1), we have $\mathcal{P}(\rho) \xrightarrow{init_A(B)} C_1[\mathcal{P}(\rho')]$ and finally $Q' \equiv C_1[\nu\mathcal{V}_{\rho'}.(U'' \mid \mathcal{P}(\rho'))]$.

- Otherwise, let $\rho' = \rho[i \mapsto A\ E]$. By rule INIT-E, we have $U : \rho \xrightarrow{\nu iE.\overline{init}_A\langle i,E\rangle} \rho' : U'$. Using Lemma 10(2), we have $\mathcal{P}(\rho) \xrightarrow{init_A(E)} C_2[\mathcal{P}(\rho')]$ and finally $Q' \equiv \nu E.C_2[\nu\mathcal{V}_{\rho'}.(U' \mid \mathcal{P}(\rho'))]$. □

We are now ready to prove a privacy lemma that generalizes Lemma 5 to arbitrary user processes with refined control states.

Lemma 19 (Privacy with state) *If* $T_1 \approx_{\mathcal{C}} T_2$, *then* $\mathcal{Q}(T_1) \approx_l \mathcal{Q}(T_2)$.

Proof: Our proof relies on the technique detailed in Lemma 9: we show that the relation

$$\mathcal{R} \stackrel{\text{def}}{=} \{(\mathcal{Q}(T_1), \mathcal{Q}(T_2)) \mid T_1 \approx_{\mathcal{C}} T_2\}$$

where $T_z = \rho_z : U_z$ range over processes with refined control states is a weak bisimulation up to context, $\rightarrow_d$, and strong bisimilarity.

In order to establish the static equivalence requirement $\mathcal{Q}(T_1) \approx_s \mathcal{Q}(T_2)$, we use

$$\begin{aligned}\mathcal{Q}(T_z) &\stackrel{\text{def}}{=} \nu\mathcal{V}_{\rho_z}.(U_z \mid \mathcal{P}(\rho_z)) \\ &\approx_s \nu\mathcal{V}_{\rho_z}.\Big(U_z \mid \nu(N_x)_{x\in D}.\{\widetilde{x} = \widetilde{N_x}\}\Big) \\ &\equiv \nu(N_x)_{x\in D}.\Big(\{\widetilde{x} = \widetilde{N_x}\} \mid \nu\mathcal{V}_{\rho z}.U_z\Big)\end{aligned}$$

where $D \stackrel{\text{def}}{=} dv(\mathcal{P}(\rho_z))$ is the (identical) set of variables defined in $\mathcal{P}(\rho_1)$ and $\mathcal{P}(\rho_2)$ and D' is the subset of D without the variables K_i—these key variables K_i appear in $\mathcal{V}_{\rho z}$. The equivalences above follow from the definition of the translation, Lemma 12, and structural rearrangement. Finally, we use the static equivalence requirement of our private bisimilarity hypothesis, $\nu\mathcal{V}_{\rho 1}.U_1 \approx_s \nu\mathcal{V}_{\rho 2}.U_2$, in the common context $\nu(N_x)_{x\in D}.(\{\widetilde{x} = \widetilde{N_x}\} \mid [\_])$.

The proof of the two weak bisimulation properties is by case analysis of the transitions $\mathcal{Q}(T_1) \xrightarrow{\alpha} Q_1'$ and their relation to the transitions of T_1 (and U_1) and those of $\mathcal{P}(\rho_1)$, using the cases of Lemma 18 then Lemma 10.

(1) We detail the case $\alpha \neq \tau$. (The case $Q_1 \rightarrow Q_1'$ is essentially the same.)
By Lemma 18, we have $T_1 \xrightarrow{\alpha} T_1'$ and $Q_1' \equiv \mathcal{Q}(T_1')$.
By private bisimilarity (Definition 4(3)), we have $T_2 \rightarrow^* \xrightarrow{\alpha} \rightarrow^* T_2'$ with $T_1' \approx_{\mathcal{C}} T_2'$.

Using rule LIFT, we carry over this series of transitions to U_2 in the evaluation context $\nu\mathcal{V}_{\rho 2}.(\_ \mid \mathcal{P}(\rho_2))$, and obtain $\mathcal{Q}(T_2) \rightarrow^* \xrightarrow{\alpha} \rightarrow^* \mathcal{Q}(T_2')$ with $\mathcal{Q}(T_1') \mathcal{R} \mathcal{Q}(T_2')$.

(2) We use the subcases and notations of Lemma 18:

(a) By Lemma 18, we have $T_1 \xrightarrow{\overline{init}\,\nu i} T_1'$ and $Q_1' \equiv C_1[\mathcal{Q}(T_1')]$. By private bisimilarity for the transition $T_1 \xrightarrow{\overline{init}\,\nu i} T_1'$, we have $T_2 \rightarrow^* \xrightarrow{\overline{init}\,\nu i} \rightarrow^* T_2'$ with $T_1' \approx_{\mathcal{C}} T_2'$, for some $T_2' = \rho_2[i \mapsto A_2\ B_2] : U_2'$.

By rules INIT and LIFT, we obtain transitions $U_2 \rightarrow^* \xrightarrow{\overline{init}_{A_2}\langle B_2\rangle} \rightarrow^* U_2'$ and finally $\mathcal{Q}(T_2) \rightarrow^* C_1[\mathcal{Q}(T_2')]$. Relying on Lemma 9 (bisimulation up to context), we discard the common evaluation context $C_1[\_]$ and conclude with $\mathcal{Q}(T_1') \mathcal{R} \mathcal{Q}(T_2')$.

(b) By Lemma 18, we have $T_1 \xrightarrow{\nu i E.\overline{init}_A\langle i,E\rangle} T_1'$ and $Q_1' \equiv \nu E.C_2[\mathcal{Q}(T_1')]$.

We use private bisimilarity for the transition $T_1 \xrightarrow{\nu i E.\overline{init}_A\langle i,E\rangle} T_1'$, apply rules INIT-E and LIFT to the resulting transitions, discard $\nu E.C_2[\_]$, and conclude similarly.

(3) We decompose internal communication steps on c_1 or c_2 into an output followed by an input on that channel, up to a variable restriction. We rely on other cases (twice) for simulating these transitions, remark that the resulting pair of labelled transitions can be composed to form an internal step, and conclude up to context for the variable restriction.

(4) We use the cases of Lemma 10 for input on c_1:

Case 3:

We have $X_1 = x_{1i}$ with $\rho_1 = \rho_1'[i \mapsto ?A_1\ B_1]$.

Let $C_3[\_] \stackrel{\text{def}}{=} \nu x_{2i}.(\overline{c}_2\langle x_{2i} \mid [\_]\rangle)$. Lemma 10 yields $Q_1' \rightarrow_d^+ C_3[\mathcal{Q}(T_1')]$ with two cases

$T_1' = \rho_1'[i \mapsto ?A_1\ B_1\ K_i] : U_1 \mid \overline{accept}_{B_1}\langle A_1, K_i\rangle$ or

$T_1' = \rho_1'[i \mapsto ?A_1\ B_1\ -] : U_1$ depending on $A_1 \in S_{B_1}$ in ρ_1.

Rule ACCEPT applies in both cases and yields $T_1 \xrightarrow{accept\ i} T_1'$.

By private bisimilarity, $\rho_2 = \rho_2'[i \mapsto ?A_2\ B_2]$ for some $A_2, B_2 \in \mathcal{C}$, and we have $T_2 \rightarrow^* \xrightarrow{accept\ i} \rightarrow^* T_2'$ with $T_1' \approx_{\mathcal{C}} T_2'$ and two cases

$T_2' = \rho_2'[i \mapsto ?A_2\ B_2\ K_i] : U_2'$ with $U_2 \rightarrow^* U_2'$, or

$T_2' = \rho_2'[i \mapsto ?A_2\ B_2\ -] : U_2'$ with $U_2 \mid \overline{accept}_{B_2}\langle A_2, K_i\rangle \rightarrow^* U_2'$ depending on $A_2 \in S_{B_2}$ in ρ_2 (but not on $A_1 \in S_{B_1}$ in ρ_1).

Using rule ACCEPT, rule LIFT, and Lemma 10, we build transitions $\mathcal{Q}_2 \rightarrow^* \xrightarrow{\alpha} \rightarrow_d^* \rightarrow^* C_3[\mathcal{Q}(T_2')]$. We discard $C_3[\_]$ and conclude.

Case 4 when $W = A$, and

Case 5 with the same hypotheses except that $A \notin S_B$ in ρ_1:

Let $U_1' = U_1 \mid \nu N.\overline{accept}_{B_1}\langle A_1, N\rangle$ if $A \in S_B$ in ρ_1 and $U_1' = U_1$ otherwise. By rule ACCEPT-FAKE, we have $T_1 \xrightarrow{accept_B(A)} T_1' \stackrel{\text{def}}{=} \rho_1 : U_1'$.

By private bisimilarity, we obtain $T_2 \rightarrow^* \xrightarrow{accept_B(A)} \rightarrow^* T_2'$ and $T_1' \approx_{\mathcal{C}} T_2'$, with two cases in the application of ACCEPT-FAKE, depending on $A \in S_B$

in ρ_2.
For $z = 1, 2$, let T'_{zx} be T'_z with the additional message X_1 in F_B. Let $M \stackrel{\text{def}}{=} \nu N.\bar{c}_2\langle N\rangle$. In case 4 ($A \in S_B$ in ρ_1), we have

$$\begin{aligned} Q'_1 &\to_d^+ \nu x_2, K.(\bar{c}_2\langle x_2\rangle \mid \mathcal{Q}(\rho_{1x} : U_1 \mid \overline{\mathit{accept}}_B\langle A, K\rangle) \mid \chi(V, A\ B)) \\ &\sim_l \nu x_2, K.(\bar{c}_2\langle x_2\rangle \mid \mathcal{Q}(\rho_{1x} : U_1 \mid \overline{\mathit{accept}}_B\langle A, K\rangle) \mid \chi^\circ) \\ &\equiv M \mid \mathcal{Q}(T'_{1x}) \end{aligned}$$

using equivalence (A.9) in Lemma 13. In case 5 ($A \notin S_B$ in ρ_1), we simply have

$$Q'_1 \to_d^+ M \mid \mathcal{Q}(\rho_{1x} : U_1) = M \mid \mathcal{Q}(T'_{1x})$$

For each of the two cases of $A \in S_B$ in ρ_2, we use rules LIFT and ACCEPT-FAKE to build transitions $\mathcal{Q}_2 \to^* \stackrel{\alpha}{\to} \to_d^+ \sim_l \to^* M \mid \mathcal{Q}(T'_{2x})$, which implies $\mathcal{Q}_2 \to^* \stackrel{\alpha}{\to} \to^* \sim_l M \mid \mathcal{Q}(T'_{2x})$.
By Lemma 17(1), we obtain $T'_{1x} \approx_{\mathcal{C}} T'_{2x}$. We discard M to conclude.

Case 4 when $W \in S_B \setminus \mathcal{C}$ in ρ_1: By rule ACCEPT-E, we have

$$T_1 \xrightarrow{\mathit{accept}_B(W,V)} T'_1 = \rho_1 : U_1 \mid \overline{\mathit{accept}}_B\langle W, V\rangle$$

By private bisimilarity, $T_2 \to^* \xrightarrow{\mathit{accept}_B(W,V)} \to^* T'_2$ with $T'_1 \approx_{\mathcal{C}} T'_2$. Moreover, the condition of ACCEPT-E ensures that $W \in S_B \setminus \mathcal{C}$ in ρ_2.
Let T'_{1x}, T'_{2x} be obtained from T'_1, T'_2 by adding the message X_1 to F_B. By Lemma 17(1), we also have $T'_{1x} \approx_{\mathcal{C}} T'_{2x}$.
Let $C_4[\_] \stackrel{\text{def}}{=} \nu x_2 K.([\_] \mid \bar{c}_2\langle x_2\rangle \mid \chi(V, W\ V))$. By Lemma 10, we have $Q'_1 \to_d^+ C_4[\mathcal{Q}(T'_{1x})]$. Using rule LIFT, rule ACCEPT-E, and Lemma 10, we build transitions $\mathcal{Q}(T_2) \to^* \stackrel{\alpha}{\to} \to^* C_4[\mathcal{Q}(T'_{2x})]$. We discard $C_4[\_]$ and conclude using $T'_{1x} \approx_{\mathcal{C}} T'_{2x}$.

Case 5 except as above: Let T_{1x}, T_{2x} be obtained from T_1, T_2 by adding the message X_1 to F_B (with no effect if $X_1 \in F_B$ already).
By Lemma 17(1), we obtain $T_{1x} \approx_{\mathcal{C}} T_{2x}$.
Let $C_r[\_] \stackrel{\text{def}}{=} \nu N_B.\bar{c}_2\langle N_B\rangle \mid [\_]$. By Lemma 10, we have

$$Q'_1 \to_d^+ C_r[\mathcal{Q}(T_{1x})] \quad \text{and} \quad \mathcal{Q}(T_2) \xrightarrow{c_1(X_1)} \to_d^+ C_r[\mathcal{Q}(T_{2x})]$$

We discard $C_r[\_]$ and conclude using $T_{1x} \approx_{\mathcal{C}} T_{2x}$.

(5) We use the cases of Lemma 10 for input on c_2. Let $\rho'_z[i \mapsto t_z] = \rho_z$ for $z = 1, 2$.
Case 6: By rule CONNECT, we have $T_1 = \rho'_1[i \mapsto t_1] : U_1$ and $T_1 \xrightarrow{\mathit{connect}\ i} T'_1 = \rho'_1 : \nu K_i.U'_1$, with two cases for U'_1 depending on t_1: either $t_1 = A_1\ B_1\ K_i$ and $U'_1 = U_1 \mid \overline{\mathit{connect}}_{A_z}\langle B_z, K_i\rangle$, or $t_1 = A_1\ B_1 -$ and $U'_1 = U_1$.
By private bisimilarity, we have $T_2 \to^* \xrightarrow{\mathit{connect}\ i} \to^* T'_2$ with $T'_1 \approx_{\mathcal{C}} T'_2$ and

moreover $T_2 = \rho_2'[i \mapsto t_2] : U_2$ and $T_2' \equiv \rho_2' : \nu K_i.U_2'$.
Let $T_z'' = \rho_z'[i \mapsto \star t_z] : U_z'$. By Lemma 17(2), $T_1' \approx_{\mathcal{C}} T_2'$ implies $T_1'' \approx_{\mathcal{C}} T_2''$.
By Lemma 10, $Q_1' \to_d \mathcal{Q}(T_1'')$. By rule Lemma 10 and rules LIFT and CONNECT, we obtain $\mathcal{Q}(T_2) \to^* \xrightarrow{\alpha} \to^* \mathcal{Q}(T_2'')$. We conclude using $T_1'' \approx_{\mathcal{C}} T_2''$.

Case 7: Let T_z' be T_z with a $\star$ at index i. We have $Q_1' \to_d \mathcal{Q}(T_1')$ and $\mathcal{Q}_2 \xrightarrow{\alpha} \to_d \mathcal{Q}(T_2')$. By Lemma 17(2), $T_1 \approx_{\mathcal{C}} T_2$ implies $T_1' \approx_{\mathcal{C}} T_2'$.

Case 8: By rule CONNECT-E, we have

$$T_1 = \rho_1'[i \mapsto A\ E] : U_1 \xrightarrow{\mathit{connect}_A(i,E,V)} T_1' = \rho_1' : U_1'$$

with $U_1' = U_1 \mid \overline{\mathit{connect}}_A\langle E, V\rangle$.
By private bisimilarity and rule CONNECT-E,

$$T_2 = \rho_2'[i \mapsto A\ E] : U_2 \to^* \xrightarrow{\mathit{connect}_A(i,E,V)} \to^* T_2' = \rho_2' : U_2'$$

and $T_1' \approx_{\mathcal{C}} T_2'$ for some U_2'. Let $T_z'' \stackrel{\text{def}}{=} \rho_z'[i \mapsto \star A\ E] : U_z'$. By Lemma 10, we have $Q_1' \to_d \mathcal{Q}(T_1'')$ and we build $\mathcal{Q}(T_2) \to^* \xrightarrow{\alpha} \to^* \mathcal{Q}(T_2'')$. By Lemma 17(3), we obtain $T_1'' \approx_{\mathcal{C}} T_2''$ and conclude.

Case 9: Let T_z' be T_z without the session at index i. We have $Q_1' \to_d \mathcal{Q}(T_1')$ and $\mathcal{Q}_2 \xrightarrow{\alpha} \to_d \mathcal{Q}(T_2')$. By Lemma 17(3), $T_1 \approx_{\mathcal{C}} T_2$ implies $T_1' \approx_{\mathcal{C}} T_2'$. □

Proof of Lemma 5: This is a special case of Lemma 19, with initial control states ε_z instead of arbitrary control states ρ_z in T_1 and T_2. □

A.6 Proofs of section 8.3

Proof of the basic example: For any process with control state $\rho : U$ such that $A \notin S_B$ and i is fresh, we have the blinded transitions

$$\begin{aligned} T \stackrel{\text{def}}{=} \rho : U \mid \overline{\mathit{init}}_A\langle B\rangle \quad &\xrightarrow{\overline{\mathit{init}}\,\nu i} \quad T_i = \rho[i \mapsto A\ B] : U \\ &\xrightarrow{\mathit{accept}\ i} \quad T_r = \rho[i \mapsto A\ B\ -\ ] : U \\ &\xrightarrow{\mathit{connect}\ i} \quad T' = \rho : U \end{aligned}$$

For $z = 1, 2$, assume $A_z, B_z \in \mathcal{C}$ with $A_z \notin S_{B_z}$, and let T_z be T with A_z, B_z instead of A, B. In order to show that $T_1 \approx_{\mathcal{C}} T_2$, we establish that the relation

$$\mathcal{R} \stackrel{\text{def}}{=} \bigcup_{\rho:U} \{(T_1, T_2), (T_{1i}, T_{2i}), (T_{1r}, T_{2r})\} \cup \approx_{\mathcal{C}}$$

is a private bisimulation. By construction, $\mathcal{R}$ is closed by the transitions detailed above. Any other transition does not depend on the *init* message and leads to related processes with control state, in the same case of the definition of $\mathcal{R}$. □

Proof of Theorem 6: Although each user process U_j initially attempts a single session, the environment can trigger *accept* messages using transition rules ACCEPT-FAKE or ACCEPT-E. For any given series of transitions derived from these rules, let U_{aj} be the resulting user subprocess—this subprocess consists of *accept* messages and depends only on these transitions and ε_j. For any fresh index i, we have

$$\begin{aligned}
& T_j \stackrel{\text{def}}{=} \varepsilon_j : U_{aj} \mid U_j \\
\xrightarrow{\overline{\mathit{init}}\,\nu i} \quad & \varepsilon_j[i \mapsto A_j\ B_j] : U_{aj} \mid \mathit{connect}_{A_j}(B_j, K).V_j \\
\xrightarrow{\mathit{accept}\ i} \quad & \varepsilon_j[i \mapsto A_j\ B_j\ K_i] : U_{aj} \mid (\overline{\mathit{accept}}_{B_j}\langle A_j, K_i\rangle \mid \mathit{connect}_{A_j}(B_j, K).V_j \\
\xrightarrow{\mathit{connect}\ i} \quad & \varepsilon_j : U_a \mid \nu K_i. \begin{pmatrix} \overline{\mathit{accept}}_{B_j}\langle A_j, K_i\rangle \mid \\ \overline{\mathit{connect}}_{A_j}\langle B_j, K_i\rangle \mid \mathit{connect}_{A_j}(B_j, K).V_j \end{pmatrix} \\
\rightarrow \quad & T'_j \stackrel{\text{def}}{=} \varepsilon_j : U_{aj} \mid U'_j
\end{aligned}$$

We omit other, uniform transitions that extend U_{aj} or lead to the failure of the session.

Let $\mathcal{R}$ relate these extended processes with control state, except (T'_1, T'_2).

From the hypothesis $\varepsilon_1 : U'_1 \approx_{\mathcal{C}} \varepsilon_2 : U'_2$, we show that $T'_1 \approx_{\mathcal{C}} \leftarrow^* T'_2$ by induction on the series of transitions that yield U_{a1} and U_{a2}. For each transition, we apply clause 3 in the definition of private bisimilarity and remark that the labelled transition commutes with any silent step. Similarly, we have $T'_1 \rightarrow^* \approx_{\mathcal{C}} T'_2$, and thus $T'_1 \approx_{\mathcal{C}} T'_2$.

Using $T'_1 \approx_{\mathcal{C}} T'_2$, we easily show that $\mathcal{R} \cup \approx_{\mathcal{C}}$ is a private bisimulation, and conclude from the initial state $T_1 \approx_{\mathcal{C}} T_2$ when $U_{a1} = U_{a2} = \mathbf{0}$. □

Proof of Theorem 7: Let U be a process of the form $\prod_{i=1}^{n} \overline{\mathit{init}}_{A_i}\langle X_i\rangle$. Since there is no internal step and rule LIFT does not apply to control messages, any accept_B or $\mathit{connect}_A$ message in parallel with U is inert. We let $\_$ range over parallel compositions of such messages.

The transitions of $\varepsilon : U$ are interleavings of the following transitions:

(1) If $X_i = B \in \mathcal{C}$ then, independently of A and B, we have transitions

$$\varepsilon : U' \mid \overline{\mathit{init}}_{A_i}\langle B\rangle \xrightarrow{\overline{\mathit{init}}\,\nu i} \xrightarrow{\overline{\mathit{accept}}\ i} \xrightarrow{\overline{\mathit{connect}}\ i} \varepsilon : U' \mid \_$$

(2) Otherwise, we have transitions

$$\varepsilon : U' \mid \overline{\mathit{init}}_{A_i}\langle X_i\rangle \xrightarrow{\nu i.\overline{\mathit{init}}_{A_i}\langle X_i\rangle} \xrightarrow{\overline{\mathit{connect}}\langle i, V\rangle} \varepsilon : U' \mid \_$$

(3) Independently, we have transitions $\varepsilon : U \xrightarrow{\overline{accept}_B\langle E,V\rangle} \varepsilon : U \mid \_$ (if and only if $E \in S_B$) and $\varepsilon : U \xrightarrow{\overline{accept}_A\langle B\rangle} \varepsilon : U \mid \_$ (whether or not $A \in S_B$).

Let $\mathcal{R}$ be the relation such that (1) $(\varepsilon_1 : U_1 \mid \_, \varepsilon_2 : U_2 \mid \_) \in \mathcal{R}$ for all $\varepsilon_1 : U_1$ and $\varepsilon_2 : U_2$ that meet the conditions of Theorem 7 and (2) $\mathcal{R}$ is closed by application to both processes of any transitions appearing above in cases 1 and 2. The relation $\mathcal{R}$ is a private bisimulation, so $\mathcal{R} \subseteq \approx_{\mathcal{C}}$. We conclude by Lemma 5. □

Proof of Theorem 8: In this proof, for all definitions, we use the set of compliant principals $\mathcal{C} \uplus \{X\}$ with $S_X = \emptyset$ (rather than $\mathcal{C}$). In addition, we let $\mathcal{P}^-(\rho)$ be the translation state $\mathcal{P}(\rho)$ with $\mathbf{0}$ instead of $\mathcal{P}_X$. We use the candidate relation $\mathcal{R}$ defined by

$$\{(\nu\, \mathcal{V}_X.\mathcal{P}(\rho), \nu\, \mathcal{V}_X.\mathcal{P}^-(\rho)) \mid \rho \text{ extends } \varepsilon \text{ and has no } t \text{ initiated by } X\}$$

We rely on Lemma 9 and the case analysis of Lemma 10. We conclude with $\rho = \varepsilon$.

The processes on the left of $\mathcal{R}$ have extra transitions that use the replicated input on c_1 in $\mathcal{P}_X$ (transition 3 with $A \notin S_B$ for $B = X$ in Lemma 10). These inputs can be simulated on the right using transitions 5 with $C \in \mathcal{C}$—since the received message X_1 meets the condition for transition 4 for at most one $B \in \mathcal{C}$ and $|\mathcal{C}| > 1$, we can always choose some $C \in \mathcal{C} \setminus \{B\}$.

All other transitions are in direct correspondence, and lead to related processes for an updated ρ in the same evaluation context. The condition on ρ is preserved by all transitions up to context because $init_X$ is restricted and appears only in a replicated input in $\mathcal{P}(\rho)$. □