GNY logic is not good enough yet. We analyze some general problems of authentication logics. We introduce completeness assumptions, add an inference rule, enhance the protocol parser and introduce maximum belief sets.

Chapter 6

Extending GNY Logic

GNY logic is not powerful enough to analyze the protocols we want to analyze in Chapter 9 of this thesis. Therefore, we will extend GNY logic. Before extending GNY logic, we need to address a few caveats of authentication logics in general.

6.1 Why Authentication Logics Are So Tricky

Arguably the most difficult part of constructing an authentication logic is crafting the list of inference rules. The inference rules should precisely express 'all relevant' properties of cryptographic primitives such as cryptographic hash functions, symmetric encryption and asymmetric encryption. The list of inference rules often has omissions, and individual inference rules can have unstated assumptions, and sometimes even downright fatal flaws. The previous chapter (Chapter 5) elaborated on such a flaw, found in BAN logic. Unstated assumptions and omitted inference rules are sometimes two sides of the same coin, but not necessarily so. An unstated assumption may still be a legitimate assumption. An erroneously omitted inference rule is a Bad Thing: it means that the logic does not detect a flaw which could have been detected if the rule were not omitted.

6.1.1 Unstated Assumptions: Length-Concealment and Non-Incrementality

We will give two examples of unstated assumptions with regard to inference rules. We sketch the problem and offer directions of how to solve the problems that result from the unstated assumptions.

The first example has to do with the constructs used to formalize encrypted messages. Consider the following situation in a fictive university department:

> In the department, all communication is done by placing sealed envelopes on the table in the coffee corner. On these envelopes the names of the sender and the intended recipient are written, and everybody obeys the rule to only open an envelope if he are the intended recipient. Now everybody in the department knows that Alice is writing her long-awaited PhD thesis, which presumably contains shocking results. Everybody is dying to know whether Alice has submitted her manuscript to her supervisor, Bob. As long as the only envelopes from Alice to Bob on the table in the coffee corner are *a few* thin, flimsy ones, everybody will be sure that Alice has not yet submitted her manuscript. This will however change as soon as a one-inch thick envelope from Alice to Bob appears on the table.

Now, reread the description, but read 'encrypted message' where it says 'sealed envelope'. Obviously, if one does not have the right decryption key to an encrypted message, one cannot infer the contents of the message (i.e., look inside the envelope). However, in general it *is* possible to infer the *length* of an encrypted message from the encrypted message without knowing the decryption key (i.e., one can look at the size and measure the weight of a sealed envelope). In fact, length-concealing encryption schemes are a rarity indeed. Within almost every authentication logic however, a length-concealing encryption scheme is assumed. We have not found any paper presenting an authentication logic in which this assumption is explicitly stated. There is only one way to infer that this assumption is actually made: there is no inference rule of roughly the following form[1]:

P-LEN $\dfrac{P \lhd \{X\}_K,\ \mathrm{length}(X, l)}{P \lhd \mathrm{length}(X, l)}$ If a principal is told an encryption $\{X\}_K$ of formula X, and the formula X has length l, then he is considered to have also been told the length l of the formula X.

Length-concealing encryption schemes are, from an information theory perspective, strictly stronger than length-revealing encryption schemes. Therefore, if one analyzes a protocol in an authentication logic, and models a length-revealing encryption scheme as length-concealing, one makes an unjustified assumption. Care should be taken that such an unjustified assumption does not invalidate the correctness proof that is obtained from the authentication logic. An example of a protocol that can be wrongly proven secure using such an assumption is derivable from the university situation described above.

[1] This inference rule follows the notation of GNY logic, see Appendix B. Likewise, the recognizability ($\phi(\cdot)$) concept in GNY logic could be extended to facilitate recognition of the length of formulae.

For our second example on unstated assumptions, we return again to cryptographic hash functions (cf. Chapter 3). Inference rule **P4** of GNY logic[2] states that whoever possesses X, can obtain in a computationally feasible way $H(X)$. There is no inference rule of the following form:

H-INC $\dfrac{P \ni H(X,Y),\ P \ni (Y,Z)}{P \ni H(X,Z)}$

If a principal possesses the hash value of the concatenation of X and Y, and possesses Y and Z, then he is capable of possessing the hash value of the concatenation of X and Z.

Note that in **H-INC**, P may be ignorant of the actual contents of X.

If the cryptographic hash function denoted by $H(\cdot)$ is *incremental* as described in Section 3.6, an inference rule like **H-INC** should be added to the logic. Otherwise, the logic would systematically underestimate the inference capabilities of the principals, which is undesirable. In fact, if this inference rule would be added, the protocols described in Chapters 9 and 10 of this thesis would be rendered worthless. In Chapters 9 and 10, we assume that the used cryptographic hash function is indeed non-incremental.

Thus, one has to be careful and be explicit about whether a hash function, modeled in an authentication logic, is considered to be incremental. It should be noted that the authors of most authentication logics are not to blame for this omitted assumption, as the concept of incremental cryptographic hash functions has emerged years after most authentication logics were conceived.

6.1.2 Omitted Inference Rules: The Key to Incompleteness

It goes almost without saying that it is extremely difficult to create an authentication logic from scratch that captures all possible cryptographic primitives that might be used in a security protocol. To take an example: oblivious transfer is not facilitated by any known authentication logic to date. While it is wise to start small and keep authentication logics as simple as possible, incomplete coverage of cryptographic primitives can have implications for the results obtained by application of an authentication logic. The coverage of cryptographic primitives by an authentication logic is determined by two aspects:

1. *The formal language*
   It must be possible to denote in the formal language of the logic that a certain cryptographic primitive has been applied to some message.

2. *The inference rules*
   The 'essence' of a cryptographic primitive lies in what knowledge or possessions are required to perform specific operations. To correctly reflect a primitive, the inference rules should precisely reflect what can and what can not be done with the use of a certain primitive.

[2] See Appendix B, page 185.

Thus, incomplete coverage of primitives may be due to two types of omissions. Either the the formal language is too restricted, or the list of inference rules does not correctly reflect the workings of the primitives facilitated in the language. The former type omission is not a large problem: if a primitive is omitted in the language, a protocol using the primitive cannot be modeled using the authentication logic, and therefore cannot be falsely 'proven correct'. The latter type of omission poses a serious problem. In fact, it is this type of omission that is partially to blame for the incompleteness of authentication logics.

Omitted inference rules are no rarities in the field of authentication logics. In fact, it is common to add inference rules when needed to prove a protocol correct, and to ignore the inference rules not needed in the particular protocol proof.

To illustrate how an omitted inference rule makes an authentication logic incomplete, consider the following type of protocol. Let us assume we have some protocol which uses asymmetric encryption, but requires the public keys to be kept secret within a certain group of principals. (Actually, we do not know whether such a protocol exists in practice, but there is no reason to render such a protocol unviable.) Within this protocol, at a certain moment, a message $\{X\}_{-K}$ is sent. This is the message X, cryptographically signed with private key $-K$. The message X should remain secret within the group of principals knowing the public key $+K$.

Does this protocol have a major problem? In fact, *it does*! If the signature scheme is like almost any signature scheme used in common practice, it is possible to derive X from $\{X\}_{-K}$ without knowing or possessing the public key $+K$.[3] Thus, for the essentials of such a signature scheme to be reflected properly, an inference rule of the following form is required:[4]

T6′ $\dfrac{P \lhd \{X\}_{-K}}{P \lhd X}$ If a principal is told a formula encrypted with a private key (i.e., a signed formula) then he is considered to have also been told the contents of that formula.

Without an inference rule like **T6′**, an authentication logic fails to find the huge gap in the above-mentioned protocol. If the protocol description were to be adjusted such that it explicitly states that a signature scheme is used that does not leak the message, this should be reflected in an assumption about the inference rules. The assumption should be something like this:

> There is no set of inference rules which allows a principal to derive X from $\{X\}_{-K}$.

This assumption is essentially a *completeness assumption*: it states that the list of inference rules is complete with respect to some essential property of a specific cryptographic primitive. In Chapter 9 we will use completeness assumptions to prove certain principals cannot infer specific information.

[3] This means that is is possible to read a message even if one does not recognize the signature.
[4] This rule **T6′** is of course a strengthened version of rule **T6**.

6.2 Proofs of Knowledge and Ignorance

Proving that security protocols meet their specification generally involves proving three properties of a protocol:

1. the participating principals learn what they should learn,
2. what the participating principals learn is indeed true, and
3. no principal learns facts he ought not to know.

Observe that properties 1 and 2 address mainly *liveness*, and that property 3 addresses *safety*.

Thus, a correctness proof requires both a proof of things that are learned, and things that are *not* learned in course of a protocol run. Authentication logics generally focus on the learning part, and less if at all on the not-learning part. If an analysis of a protocol using an authentication logic does not expose a flaw, this means that properties 1 and 2 are not violated, of course assuming that the logic itself is 'correct'.

If one wants to prove property 3, that principals can *not* infer specific facts, one has to model the limitations of the inference capabilities of the agents, and show that the limitations effectively obstruct principals from inferring certain relevant facts (Cf. [ABV01]). To model the inference limitations of principals, we need to model what inference rules are available to an agent. This is where a nasty property of the authentication logics comes in: none of the authors of these logics claim that the list of inference rules provided in the logic is indeed complete in the sense that no more inference rules can be added. We have to make completeness assumptions (as in Section 6.1.2) to be able to prove property 3 of a protocol. We do not believe nor claim that completeness assumptions are sufficient for proving property 3 of a protocol. This issue will be discussed in Section 6.2.2.[5]

Typically, the (in)ability to draw specific conclusions plays a crucial role in a security protocol: for example showing a message which can only be constructed with knowledge of the specific conclusion constitutes a proof of identity. This is best illustrated with cryptographic signatures. If some principal V sees $\{X\}_{-K}$, and knows $-K$ is the private key of P, then V may believe P once conveyed X. Why is this the case? Essentially, because no principal but P possesses $-K$. This begs the question: is there anything in the authentication logic which prevents a malicious principal to create $\{X\}_{-K}$ out of thin air? The answer is simple: no, there is nothing which prevents a principal to do this *within the authentication logic*. Of course, in an actual 'real world' protocol run, it is impossible to do this. So, the obstruction that one cannot construct messages which are computationally hard to construct, should be incorporated into the logic.[6]

[5] Thus, completeness assumptions are *necessary*, but not necessarily *sufficient*.

[6] Though this type of reasoning is not new in the domain of authentication logics, it has never been incorporated into an authentication logic.

While lacking a means to prove property 3, the meaning of a correctness proof in an authentication logic is only limited. If it exposes a flaw in a protocol, the protocol will indeed be flawed. However, not finding any errors does not guarantee that the protocol is correct. Therefore, proving a protocol correct using an authentication logic, only proves that the protocol has passed a first test of some not-so-obvious flaws. However, we deem such a proof an important step in defending correctness of protocols.

In the light of these considerations, we need to extend the authentication logic we use (GNY) in such a way that

- what *should* be learned can in fact be learned[7], and
- what *should not* be learned, can be proven not to be learned.

What can and cannot be learned should have causal effects in protocol runs. We extend GNY logic in this way in the next two sections.

6.2.1 New Inference Rules for Proving Possession

In Chapters 8–10, we will present new methods for proving possession of information based on cryptographic hash functions. In order to prove these methods correct, we have to model some properties of cryptographic hash functions which have not yet been modeled in any authentication logic. This modeling is done by adding the appropriate inference rule **H2** to GNY logic.

The reasoning behind the added inference rule takes as its starting point another inference rule (**I4**) which reflects a way in which possession can be proven. Slowly we will manipulate this rule until we arrive at **H2**. Note that we do not depart from an inference rule like **H-BAN**, because that rule is faulty (see Chapter 5).

Since we are discussing inference rules which can be applied in protocols in which one principal (the *prover*) proves something to another principal (the *verifier*), we will use in the presented inference rules the names P and V to denote the prover and the verifier (as opposed to using the names P and Q for two arbitrary principals). Moreover, any malicious principal in our discussion will be denoted with the name C (for Charlie). Messages are denoted X.

A well known method for proving possession of a certain message is to sign the message one wants to prove possession of, and then to show this signed message.[8] Obviously this method cannot be used in a setting where the message itself should be kept secret, because the message will be disclosed when showing the signed message. Nevertheless it is interesting to look at the inference rule that captures the interpretation of signed messages, **I4**, repeated below (from [GNY90]; note that the names of the principals have been changed, therefore we name the rule **I4′**).

[7] Of course, without making unjustifiable assumptions.

[8] The signature is required because the communication channel does not reliably 'say' who has sent a particular message. This is a result of using the malicious adversary model.

I4′
$$\frac{V \lhd \{X\}_{-K}, \quad V \ni +K, \quad V \models \overset{+K}{\mapsto} P, \quad V \models \phi(X)}{V \models P \mid\sim X, \quad V \models P \mid\sim \{X\}_{-K}}$$

What the rule says is this: If V sees a signed message $\{X\}_{-K}$, knows the public key $+K$, knows the corresponding private key $-K$ belongs to P, and recognizes X to be a message, then V is entitled to believe that P once conveyed the signed message $\{X\}_{-K}$, and thus also once conveyed the message X itself.

Important to note are two silent assumptions of this inference rule:

1. For any X, no principal C will convey $\{X\}_{-K}$ where $C \neq P$. Thus, P's private key $-K$ is only known to P and P will never convey $-K$.[9]
2. P will, in a sense, be conservative in what he signs: P will only sign intentionally and with consent.[10] P will never sign unseen messages.

The reason for the second assumption is that a digital signature is non-repudiatable.[11]. Making similar assumptions, we could introduce a new identity-related inference rule:

H1
$$\frac{V \lhd *H(X,P), \quad V \ni (X,P)}{V \models P \mid\sim (X,P), \quad V \models P \mid\sim H(X,P)}$$

If V sees a message $H(X,P)$, which V did not send himself previously, and also possesses (X,P), then V is entitled to believe that P once conveyed (X,P) and $H(X,P)$.

The assumptions under which this rule is justified are these:

1. For any X, no principal C will convey $H(X,P)$ where $C \neq P$.
2. P will, in a sense, be conservative in the set of X's for which he (P) conveys $H(X,P)$. More specifically, P will only convey $H(X,P)$ for X's of which he (P) wants to show other principals he possesses X.

These two assumptions tie together just like the two assumptions of rule **I4′**: the first assumption states that only one principal is capable of sending certain messages, and the second states that this principal will only do so with informed consent.[12]

However, the first assumption of inference rule **H1** is not justifiable. Relying on rule **H1**, a malicious principal C, knowing P and any secret X, can

[9] More precisely, we mean that no C will send $\{X\}_{-K}$ before receiving $\{X\}_{-K}$: Thus, C could perform replays of messages, but cannot generate messages signed with the key $-K$.

[10] For example, P will not sign his own death penalty.

[11] It is however important to distinguish the different *intentions* a signature may convey. A signature may convey, for example, a confirmation of a contract, or a receipt, or something completely different. The signer should always assure that he consents the intention which he conveys with his signature. In particular, a principal may sign an unseen message in a challenge-response protocol as long as the context guarantees that the signature only conveys a receipt. This can be assured by using a particular keyset for such signatures, or by including the intention in the signed message itself.

[12] Since **I4′** is just a syntactic variation of **I4**, it of course also applies to **I4**.

'commit' P to conveying the secret X by broadcasting $H(X, P)$. Assumption 1 of inference rule **I4′** does not suffer from such a problem: to construct $\{X\}_{-K}$, one has to possess $-K$.

To prevent malicious principals from creating havoc by sending $H(X, P)$, we should require the message sent to be authenticated, i.e., that it is known that P sent the message $H(X, P)$. Using sender identification, a verifier can distinguish proofs of possession by malicious principals from proofs by intended principals. When we incorporate sender identification, we can introduce a more moderate rule like this one:

H2 $$\frac{V \models P \mid\sim *H(X, P), \quad V \ni (X, P)}{V \models P \mid\sim (X, P)}$$

If V believes P once conveyed the the message $H(X, P)$, which V did not send himself previously, and if V also possesses (X, P), then V is entitled to believe that P once conveyed (X, P). This effectively eliminates the first assumption of rule **H1**.

Rule **H2** is justified under the following assumptions:

1. For any X, no principal C will convey $H(X, P)$ where $C \neq P$.
2. P will, in a sense, be conservative in the set of X's for which he conveys $H(X, P)$ in an authenticated manner (that is, such that P can be identified as the sender). More specifically, P will only convey $H(X, P)$ for X's of which he wants to show other principals that he possesses X.

Though these assumptions are not very different from the assumptions of rule **H1**, the working of rule **H2** is quite different. Firstly, a malicious principal C, knowing both X and P, cannot 'commit' P to conveying the secret X by broadcasting $H(X, P)$. Moreover, if a malicious principal would broadcast $H(X, P)$, and if P would receive it, sign it, and broadcast the signed message, this would still not result in anyone being convinced that P actually possesses X. Some may be convinced that P *conveyed* X, but conveying a message does not imply *possessing* a message!

For any principal V to believe that P possesses X, based on rule **H2**, X should be *fresh*. More precisely, X should contain a term that V believes to be fresh, and then V could apply rule **I6**. Typically, V should construct a fresh term F, and this term should be combined with X, yielding $X' = (X, F)$ and then $H(X', P)$ should be computed. If P possesses X and receives F, then P can construct a convincing proof of possession.

However, if P has an assistant C who possesses X, P might forward F to C, and C might compute $H(X', P)$ and send this term to P. In turn, P could sign this term and send it on to V, who will be convinced. Is this a problem? Well, both yes and no. Yes, because strictly spoken it does not guarantee that P actually possesses X. No, because it does guarantee that either P possesses X or P has a rather cooperative assistant who does possess X and is willing to perform computations on X on behalf of P. Assumption 1 above essentially rules out that such an assistant exists.

There is a slight technical issue with rule **H2**, which also applies to rule **H1**: How can a principal P make sure that he is not sending some message M in an authenticated manner (that is, such that P can be identified as the sender) without actually knowing that he is sending $H(X, P)$? For example, P might be required to sign a challenge M. P cannot verify whether this challenge M in fact is equal to an $H(X, P)$ if he does not possess X. This problem can be solved by adding something like a publicly known and recognizable *speech act token* to the hash value that has to be signed: P would have to sign ("I know", $H(X, P)$) instead of just $H(X, P)$. The speech act token can always be recognized by P, and therefore P can prevent erroneously signing hash values. The inference rule needs to be adjusted to reflect this, giving rule **H3** shown below. In such a way, P can make sure that he never accidentally signs a value that may be interpreted using inference rule **H3**.

$$\textbf{H3} \qquad \frac{V \models P \mid\sim (\text{"I know"}, *H(X, P)), \quad V \ni (X, P)}{V \models P \mid\sim (X, P)}$$

This rule has the same assumption as **H2**, except that for **H3**, P can really make sure the assumption is true, because P always knows it when he sends a signed message may be used using inference rule **H3**. This rule requires that the language of the GNY logic be extended with *tokens*. We decide not to do this (yet), and use rule **H2**, knowing that we can trivially modify protocols and proofs to reflect rule **H3** instead of rule **H2**.

6.2.2 Proving That Principals Do Not Learn Too Much

Authentication logics focus on establishing whether the principals interacting in a security protocol draw correct conclusions. However, for security protocols, it is also crucial to prove that certain principals *cannot* draw some specific conclusions. In this section, we will enhance GNY logic to extend its reasoning capabilities about *not* learning. First, we make sure that not learning has *causal effects* on protocol analysis, and secondly we enhance the logic to allow us to precisely state *in what circumstances* it can be guaranteed that certain facts are not learned.

Our proposal for incorporation is simple and effective. The protocol parser which translates an idealized protocol into a number of step transitions (see Section 4.3), should require that the sending party actually possesses ($\ni$) the message it is supposed to send, before sending the message.[13] Thus, any step transition is of the following form:[14]

$$[Y, \quad P \ni X]\,(P \to Q\colon X)\,[Y, \quad P \ni X, \quad Q \lhd X]$$

whereas in the original GNY logic, the step transition has only this form:

$$[Y]\,(P \to Q\colon X)\,[Y, \quad Q \lhd X]$$

[13] This way of reasoning has also been used in our earlier work [TvdRO03].

[14] For simplicity, we omit the $*$ (not-originated-here) sign which the GNY protocol parser in some cases adds to the postcondition [GNY90, Section 5]. The not-originated-here sign is implicated.

The effectiveness of this modified protocol parser lies in the fact that the protocol parser introduces a precondition that should be derivable from earlier statements. If it is impossible to derive the precondition $P \ni X$, then it is impossible to perform the protocol step $P \rightarrow Q: X$, and it is impossible to create a *legal annotation* of a protocol.

To incorporate this precondition into our notion of a *heavy annotation* (see Section 4.3), we require in a heavy annotation that every assertion of the form '*is told* ($\lhd$)' which is added after a protocol step is not only annotated with the step number, but also with the assertion number which shows that the sender actually possessed the message before sending it.[15]

This modified protocol parser and annotation requirements make sure that, just as in the 'real world', there is a causal connection from *not knowing* X to *not being able to send* X.[16]

Now that we have made sure that the inability to draw certain conclusions has causal effects on the protocol analysis, let us focus on the inability to draw certain conclusions. This should be proven for both active and passive attacks. An *active attack* is an attack in which a malicious principal manipulates the messages exchanged in a protocol in such a way that the honest participating principals learn other things than intended by the protocol. A *passive attack* is an attack in which the attacker learns something he should not learn, while the only capabilities available to the attacker are eavesdropping and inference, and notably *not* message interception and modification, as in the malicious adversary model (Dolev-Yao threat model). The literature about authentication logics generally addresses the case of active attacks (like in the Needham-Schroeder Public-Key protocol (NSPK) [Low96]), but not the case of passive attacks.

We demonstrate an approach to proving that principals cannot learn specific facts in the course of a protocol run. We believe that this is a significant contribution to establishing for concrete protocols a proof of property 3 as mentioned in Section 6.2, page 71.

Normally, when proving a protocol using an authentication logic, assumptions about the participating principals are stated. We introduce an extra principal and show that this principal cannot infer what should be kept secret. This new principal E is Eve the *e*vil *e*avesdropper. We make no assumptions on what role Eve takes in the protocol: Eve may either be one of the participants or an external observer. Just as with any other principal, we list assumptions about what Eve possesses and believes at the beginning of the protocol. The meaning of the assumptions is somewhat different, however. When we state

[15] If we return to the heavy annotation of the example of the signing parrot protocol, shown in Figure 4.4 on page 53, line 6 should carry as justification '[1](4)' instead of just '[1]'. Line 7 cannot be justified right away, but from line 6, using inference rules **P1** and **P8**, $B \ni \{N\}_{-K}$ can be inferred. The line on which $A \lhd * \{N\}_{-K}$ is inserted, should carry as justification '[2](x)', where x is the line number on which $B \ni \{N\}_{-K}$ is inferred.

[16] Note that this modified protocol parser does not rule out attacks in which forwarding of messages plays a role: to forward a message, the intruder still has to observe the message.

an assumption for a 'normal' principal participating in the protocol, this is in some sense a weakness of the protocol: it has to be met in order for the protocol to be correct. When we state an assumption about Eve, this is a strength of the protocol: even if Eve knows or possesses this a priori, the protocol is still correct in the sense that Eve cannot infer the secret. Thus, we establish the maximum amount of a priori beliefs and possessions Eve may have under which it is still impossible for Eve to infer the secret facts, a *maximum belief set*.[17] Just as with 'normal' authentication logic proofs, the list of assumptions allows to reason about subtleties concerning the quality and applicability of a protocol.

To sum up, we model two properties of a passive attacker, namely

1. its beliefs and possessions (by means of a *maximum belief set*), and
2. its inference capabilities (by means of *completeness assumptions*, see Section 6.1.2).

Using these beliefs, possessions and inference capabilities, we can compute what a passive attacker can learn from observing a protocol run. The things that should be kept secret should not be learnable for the passive attacker.

6.3 Conclusion

Authentication logics are powerful instruments that should be created and handled with care. Two types of mistakes that are easily made are (1) making implicit (unstated) assumptions, and (2) omitting inference rules. When all inference rules modeling a particular cryptographic primitive are added to an authentication logic, one can guarantee a limited kind of completeness of an authentication logic.

The important elements of our extension to GNY logic are the following:

Heavy annotations which make sure verification of a protocol analysis is structured and simple. The general structure of heavy protocol annotations that has been explained in Section 4.3 (page 53) is extended in Section 6.2.2 (page 76).

Completeness assumptions which allows one to state that an authentication logic models all essential properties of a cryptographic primitive. Completeness assumptions are introduced in Section 6.1.2 (page 70).

Inference rule H2 which captures an important property of cryptographic hash functions that had not yet been incorporated into any authentication logic. This inference rule is explained and introduced in Section 6.2.1.

A modified protocol parser which requires principals to possess a message before they can send it. This is needed for proving that not learning specific facts has causal effects on protocol evolution. This modified protocol parser is introduced in Section 6.2.2.

[17] This maximum belief set is not necessarily unique.

Maximum belief sets which allow one to reason about passive attackers and what they can and cannot learn in the course of a protocol run, depending on their a priori knowledge and possessions. Maximum belief sets are explained in Section 6.2.2.

Later on in this thesis, in Chapter 9, we will use our extended version of GNY logic to analyze our protocols.