On The Dolev-Yao Model of Symmetric Cascade Protocol

Varsha Jarali[0000−0003−3198−8448] and Shashi Kant Pandey[0000−0002−0818−6984]

Society for Electronic Transactions and Security (SETS)
(Under O/o PSA to GOI)
Taramani, Chennai -600113
India
varshapjarali@gmail.com
shashikantshvet@gmail.com

Abstract. Security protocols enable authentication, key distribution, and secure information exchange, making them essential for network security, yet flaws in their design can lead to attacks. To prevent this, formal verification methods are vital for analyzing protocol correctness. The Dolev–Yao (DY) [1] model introduced formalization for name-stamp and cascade protocols, where users apply public-key operations on messages. Brook and Otto [2] later distinguished between symmetric and non-symmetric cascade protocols, noting that all DY cases were symmetric and that attacker choices were not fully addressed. They highlighted the incomplete characterization of an attacker's power as an open problem. In this work, we extend the DY model by systematically analyzing all remaining symmetric two-party cascade protocol cases, aiming to provide a more complete foundation for building formal verification tools.

Keywords: DY Model · Formal verification · Cyber security.
Mathematics Subject Classification (MSC2020) 133574 · 03D05 . 03B38

1 Introduction

The security of communication over the internet is based on the infeasibility of reverse engineering on the public directory created for the communication protocol. Reliability and security are two important requirements in any communication protocol for cyber applications. TLS (Transport Layer Security), HTTPS (Hypertext Transfer Protocol Secure), and IPsec (Internet Protocol Security) are examples of protocols that attain all of these components during communication. A robust encryption-decryption scheme provides confidentiality, an authentication mechanism provides identity verification, and integrity protects the data from alteration. Formal methods are crucial for finding flaws in security protocol design. There's a growing interest in automating security validation through formal verification of communication protocols. This formal analysis is highly significant for building a secure cybersecurity infrastructure.

Decision procedures in cryptographic protocols rely on two key components, traceability and equivalence properties, which are central to formal analysis. Protocol design is primarily guided by two fundamental cryptographic principles: encryption ensures confidentiality, while authentication ensures identity. Dolev and Yao formalized these principles through two symbolic models for public-key protocols: the Cascade Protocol and the Name-Stamp Protocol. Their DY model remains one of the foundational symbolic approaches to formal analysis [1], providing a mathematical proof of security for two-party public-key protocols in the presence of an active adversary. The extension of the Dolev-Yao model is studied in [2], where the main idea is for 'verifiability' of the protocols at both sender and receiver levels. It characterised the model of protocols in two parts: symmetric and asymmetric. The characterisation of the protocol has been done based on the $E_X D_X = D_X E_X = 1$ and $E_X D_X = D_X E_X \neq 1$ respectively, where E_X is the encryption function and D_X is the decryption function for any user X. Based on this condition, it can be concluded that the DY model [1] covered only symmetric cascade and name stamp protocols. Asymmetric cascade and namestamp protocol proposed in [2] concerning the inequality of composition of encryption (E_X) and decryption (D_X) at the end of every user X. Many key security properties can be expressed as DY hyperproperties, which require quantification over both execution paths and attacker computations. In [4], the authors proposed a procedure to decide DY hyperproperties under the standard setting where the number of protocol sessions is bounded and the equational theory modeling cryptography is subterm-convergent. They also established the complexity of the decision problem for several important fragments of the corresponding hyperlogic. The encryption protocols have been symbolically analysed in [5] with the assumption that, obviously, desirable properties, such as that no key will coincidentally decrypt text encrypted using a different key. Some encryption protocols, such as Otway-Rees encryption, Needham-Schroeder, and a recursive protocol, have been successfully analysed using Isabelle/HOL-based proof generation tool [6]. A recent survey of symbolic methods can be refereed from [7]. ProVerif is a tool to perform symbolic analysis of a communication protocol [10]. This tool can prove three properties: reachability, correspondence assertion, and observational equivalence. A computational model provides another perspective for analyzing cryptographic operations, as it allows reasoning about their security based on underlying algorithmic assumptions and computational hardness. This model refers to the security properties in terms of the probability and computational complexity of successful attacks [9]. The connection between symbolic and computational models for doing formal analysis has been well explained in [8]. One of the recent examples of analysing privacy for TLS 1.3 handshake can be seen in [11] using symbolic protocol analyzer ProVerif.

In this paper, we present a description of the symbolic proof mechanism for security verification based on the DY model. Section 2 has complete details of the DY model along with a formal proof establishing the safety of a protocol. Section 2.1 characterizes public key protocols under the cascade model, while Section 2.2 introduces new security conditions for the doubly verified cascade protocol and

provides its complete characterization based on the attacker's capabilities. The following are the main contributions in this paper,

- A detailed review and elaborated proofs to re-look at the DY model in terms of the exhaustive set of capabilities of users X, Y, and an attacker Z are presented.
- We have characterised all remaining choices for the capabilities of users X and Y for the DY model.
- We have proposed four other sets of protocols T_1, T_2, T_3 and T_4 based on the characterisation of the capability of users X,Y and the attacker Z.
- The symbolic proof for the security and verification of protocols T_1, T_2, T_3 and T_4 has been presented.

2 Dolev Yao Formal Model

The Dolev–Yao (DY) model is a symbolic framework used to analyze security vulnerabilities in public-key protocols. It makes cryptography simpler so that the logical processes of a protocol can be emphasized instead of the complex algorithms. The model assumes a powerful attacker with full control of the network, capable of intercepting, modifying, and replaying messages. In this paper, we adopt similar symbolic representations from the DY model [1] to examine security properties of public-key protocols.

Let $\mathcal{P}$ be a public key system, where every user X has an encryption function E_X, and a decryption function D_X. A secure public directory contains all the (X, E_X) pairs, while the decryption function D_X is known only to user X.

In public-key protocols, the availability of public information and open communication channels provides significant advantages to an active adversary. By altering messages or impersonating users, the adversary can compromise protocol security. The DY symbolic model captures this threat by defining the following basic conditions for public-key protocol security:

- Knowing $E_X(M)$ and the public directory does not reveal anything about the value M.
- $E_X D_X = D_X E_X = 1$.

The representation of a finite set of symbols and the set of all finite sequences composed of symbols are represented as $\sum$ and $\sum^*$, respectively. For a symmetric cascade protocol any string of operators like $(\sigma E_X D_X \sigma')P = (\sigma D_X E_X \sigma')P = (\sigma\sigma')P$ is true for all plain text $P \in \{0,1\}^*$. $\bar{\gamma}$ is the reduced form of a string of operators γ. The symbol γ^c is denoted as a complement of γ, and $lt(\gamma)$ is the set of symbols in the string of operators available in γ.

2.1 The Cascade Protocol

The cascade model refers to the protocols where the user is applying any public key encryption and decryption on a secret message during communications. If

the users are allowed to append a name stamp or any identity feature to the encryption, then it falls under the name-stamp model. For ease of understanding, we have chosen notations similar to the notation available in the original DY model [1]. A two-party cascade protocol T is specified by a series of finite strings $\tilde{\alpha_i}$ and $\tilde{\beta_i}$ such that,

$$\alpha_i(X,Y) = \tilde{\alpha_i} \in \{z_1, z_2, z_3\}^*, \quad 1 \leq i \leq t \tag{1}$$

and

$$\beta_i(X,Y) = \tilde{\beta_i} \in \{z_1, z_2, z_4\}^*, \quad 1 \leq i \leq t', \tag{2}$$

where $t' = t$ or $t-1$. The symbol z_1, z_2, z_3, z_4 represents E_X, E_Y, D_X, D_Y. Let X, Y be two distinct users communicating under the cascade protocol T, and the sequence of operators before transmitting messages,

$$N_1(X,Y) = \alpha_1(X,Y),$$
$$N_{2j} = \beta_j(X,Y)N_{2j-1}(X,Y), \quad 1 \leq j \leq t'$$
$$N_{2i+1} = \alpha_{i+1}(X,Y)N_{2i}(X,Y), \quad 1 \leq i \leq t-1.$$

The exchange of a message M from X to Y can be seen as the set of symbols as $N_i(X,Y)M$ for $i = 1, 2, ..., t+t'$. The message exchange represented by $N_i(X,Y)M$ is illustrated in the following figure 1.

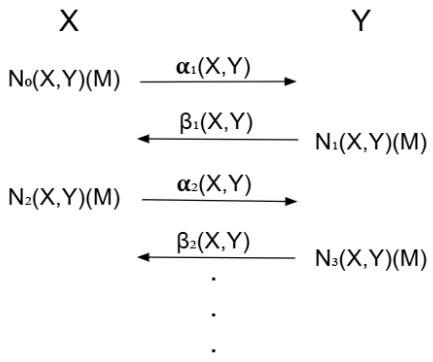


Fig. 1. A symmetric cascade protocol T

With the following example, we can get a clear picture of the cascade protocol, where messages from both users are generated through encryption-decryption operators.

Example 1. In the basic handshake protocol, the client initiates communication. The server creates a new session key, signs it with its private key, and encrypts the signed key using the client's public key. The client then decrypts the message with its private key, verifies the server's signature using the server's public key, and retrieves the session key. To complete the handshake, the client selects a

secret, encrypts it with the session key, and sends it to the server. Finally, the server decrypts this message using the same session key, confirming its integrity and establishing a secure connection. The sequence of operations between client and server is illustrated in Figure 2.

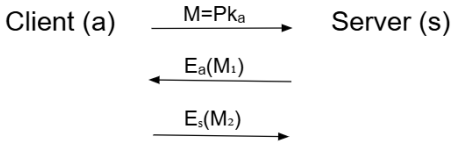


Fig. 2. Handshake Protocol

The handshake presented in figure 2 is vulnerable to Man-in-the-Middle attacks. An attacker can intercept the client's request, impersonate the client to obtain a fresh session key from the server, and then impersonate the server to the client by forwarding that key. The client, unaware of the interception, encrypts its secret with this key, allowing the attacker to decrypt and steal the secret before passing (or altering) it to the legitimate server. An illustration of this attack, showing the attacker intercepting the handshake to steal the client's secret key, is presented in figure 3.



Fig. 3. Handshake Protocol with an Active Attacker

The sequence of symbolic combinations of encryption and decryption in a public database is the main tool for finding security and leakages in functional correctness at the protocol level. To define security symbolically, we now present the definition of security for a two-party cascade model, which was given in Definitions 1 and 2 in [1]. Let X, Y and Z be three distinct user names, E be the set of all E_A, and D be the set of all D_A. Three new symbols are introduced for a two-party cascade protocol T, $\Sigma_1(Z), \Sigma_2$, and Σ_3 defined as follows,

$$\Sigma_1(Z) = E \cup \{D_Z\} = (E_A, D_Z)^*,$$
$$\Sigma_2 = \{\alpha_i(A,B) |\ \textit{for all}\ \ A \neq B\ \ \textit{and}\ \ i \geq 2\} = (E_A, E_B, D_A)^*,$$
$$\Sigma_3 = \{\beta_i(A,B) |\ \textit{for all}\ \ A \neq B\ \ \textit{and}\ \ i \geq 1\} = (E_A, E_B, D_B)^*.$$

A protocol T is insecure if some $\gamma \in (\Sigma_1(Z) \cup \Sigma_2 \cup \Sigma_3)$ exists such that

$$\overline{\gamma N_i(X,Y)} = \lambda \tag{3}$$

for some $N_i(X, Y)$. This model assumes that X, Y and Z are three users names where Z is playing the role of attacker. Let $f_1, ..., f_n$ be a sequence of operators on messages to establish a secure communication between two parties, A and B. The start of the protocol is as follows, and it can be seen in the figure 4,

1. A knows message M, f_1 is applied to M,
2. for even i the functions f_i are applied by B
3. and for odd i the functions f_i are applied by A,
4. an adversary Z has all of the f_i, E_X for all users X, D_Z and $f_1(M)$.
5. the protocol is said to be insecure if Z can construct a function g where $g \cdot f_1 = id$.

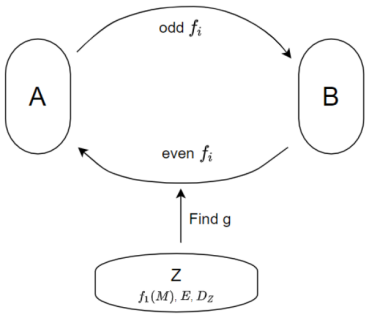


Fig. 4. An attacker model for Cascade protocol

The following definition gives the balancing property of a protocol with respect to any user A.

Definition 1. *[1] Let $\pi \in \{E, D\}^*$ be a string and A be a user name. We say that π has the balancing property with respect to A if*

$$D_A \in lt(\pi) \implies E_A \in lt(\pi).$$

Note: Z may get M if $\overline{\gamma N_i(X, Y)} = \lambda$, for some $\gamma \in (\Sigma_1(Z) \cup \Sigma_2 \cup \Sigma_3)^*$. Symbolic proof of the security of the two-party cascade protocol relies on the balanced property of the sequence and condition (3). For clarity and to support further derivations, we restate and rephrase Theorem 1 from [1] as follows.

Theorem 1. *A two-party cascade protocol $T = \left\{\tilde{\alpha_i}, \tilde{\beta_j}\right\}$ is secure if and only if*

1. *$lt(\alpha_1(X, Y)) \cap \{E_X, E_Y\} \neq \phi$, and*
2. *T is balanced.*

Proof. Let us consider Z to be a user name distinct from X and Y.

A) Necessity: Assume that either property 1) or 2) is not true, and have to show that T is insecure i.e., there exists $\gamma \in (\Sigma_1(Z) \cup \Sigma_2 \cup \Sigma_3)^*$ such that $\overline{\gamma N_i(X, Y)} = \lambda$ for some i.

If 1) is not true, i.e., $(E_X, E_Y, D_X) \cap (E_X, E_Y) = \phi$, this implies E_X, E_Y are not there in the intersection. A simpler, reduced form of $\{E_X, E_Y\}$ is constant. This implies that $\overline{\gamma N_1(X,Y)} = \lambda$ (since $\alpha_1(X,Y) = N_1(X,Y) = \{E_X, E_Y, D_X\}^*$, $\alpha^c = \{D_X, D_Y, E_X\}$), such that γ is a sequence of (E_X, E_Y). Therefore, $\gamma = \alpha_1^c = E_X \in \Sigma_1(Z)^*$ (since $\Sigma_1(Z)^* = E \cup \{D_Z\}$). Hence, T is not secure.

Now, assume 2) is false, i.e., T is not balanced. By definition, either some $\beta_k(X,Y)$ contains D_Y but not E_Y or some $\alpha_i(X,Y)(i \geq 2)$ contains D_X but not E_X. We restrict ourselves to the former case ($\beta_k(X,Y)$ contains D_Y but not E_Y); the latter case will be treated later. Under this restriction, we will establish the following stronger result. Since $\beta_k = (E_X, E_Y, D_Y)^*$, now based on the not balanced definition $\beta_k = (E_X, D_Y)^*$. For any $\delta \in (\{E_X, E_Y\} \cup D)^*$, there exists

$$\begin{aligned}\gamma &\in \{\Sigma_1(Z) \cup (\beta_k(Z,X), \beta_k(Z,Y))\}^* \\ &= \{(E \cup D_Z) \cup \{(E_Z, E_X, D_X), (E_Z, E_Y, D_Y)\}\}^*\end{aligned}$$

such that $\overline{\gamma\delta} = \lambda$.

The proof will be carried out by induction on r, the number of E_X and E_Y in the string δ. If $r = 0$ then $\gamma = \delta^c \in \Sigma_1(Z)^*$, where $\delta^c = E$, hence proved.

If $r > 0$, assume the result is true for all smaller values of r. δ is string containing exactly r number of E_X and E_Y. Without loss of generality, we can assume that the leftmost E is an E_Y. Write $\delta = \sigma_1 E_Y \sigma_2$, where $lt(\sigma_1) \cap (E_X, E_Y) = \phi$. Clearly, $\sigma_1^c \in (\Sigma_1(Z))^*$ (i.e., $\{E_X, E_Y\} \notin \sigma_1$ implies $\sigma_1 \in \{D_X, D_Y, D_Z, E_Z\}$ hence $\sigma_1^c \in (E_X, E_Y, D_Z, E_Z) \in (\Sigma_1(Z))^*$). By assuming $\beta_k(Z,Y)$ contains D_Y but not E_Y implies $\beta_k(Z,Y) = \tau_1 D_Y \tau_2$, where $\tau_1 \in (E_Z D_Y)$ and $\tau_2 \in (E_Z)$, therefore $\tau_1^c \in (D_Z E_Y)$ and $\tau_2^c \in (D_Z)$. Clearly, $\tau_i^c \in (D_Z E_Y)^* \in (\Sigma_1(Z))^*$ for $i = 1, 2$.

σ_2 contains $r - 1$ E's by induction, there exists

$$\gamma' \in (\Sigma_1(Z) \cup \{\beta_k(Z,X), \beta_k(Z,Y)\})^*$$

such that $\overline{\gamma'\sigma_2} = \lambda$. Define $\gamma = \gamma' \tau_1^c \beta_k(Z,Y) \tau_2^c \sigma_1^c$, then

$$\gamma \in (\Sigma_1(Z) \cup \{\beta_k(Z,X), \beta_k(Z,Y)\})^*.$$

From the above discussion, we get

$$\begin{aligned}\overline{\gamma\delta} &= \overline{\gamma' \tau_1^c \beta_k(Z,Y) \tau_2^c \sigma_1^c \sigma_1 E_Y \sigma_2} \\ &= \overline{\gamma' \tau_1^c E_Z E_Y D_Y \tau_2^c E_Y \sigma_2} \\ &= \overline{\gamma' \tau_1^c E_Z \tau_2^c E_Y \sigma_2} \\ &= \overline{\gamma' \sigma_2} = \lambda. \end{aligned} \tag{4}$$

Sufficiency: Assuming that both properties 1) and 2) are satisfied, we have to prove that T is secure. Suppose, to the contrary, there exists a $\gamma \in (\Sigma_1(Z) \cup \Sigma_2 \cup \Sigma_3)^*$ such that $\overline{\gamma N_i(X,Y)} = \lambda$ for some i. Write $\gamma N_i(X,Y) = P\alpha_1(X,Y)$ such that $P \in (\Sigma_1(Z) \cup \Sigma_2 \cup \Sigma_3)^*$. By definition of γ, we have

$$\overline{\overline{P}\alpha_1(X,Y)} = \lambda. \tag{5}$$

By the definition of a protocol, $lt(\alpha_1(X,Y)) \subseteq \{E_X, E_Y, D_X\}$. We distinguish two cases.

Case 1: $E_Y \in lt(\alpha_1(X,Y))$: As the string α_1 does not contain D_Y, the only possibility for (5) to hold is that $\overline{P}$ contains some D_Y but not E_Y. This implies that P does not have the balancing property with respect to Y. We know that any string $\eta \in (\Sigma_1(Z) \cup \Sigma_2 \cup \Sigma_3)^*$, $\bar{\eta}$ has the balanced property with respect to every $Y \neq Z$, hence this contradicts.

Case 2: $E_Y \notin lt(\alpha_1(X,Y))$: In this case $D_X \notin lt(\alpha_1(X,Y))$, because $\alpha_1(X,Y) = (E_X, E_Y, D_X)$, and protocol satisfy the property that

$$lt(\alpha_1(X,Y)) \cap \{E_X, E_Y\} \neq \phi$$

$((E_X, D_X) \cap \{E_X, E_Y\} \neq \phi \implies \{1\} \cap \{E_X, E_Y\} \neq \phi$, which is contradiction for assumption 1). This implies $\alpha_1(X,Y) = \{E_X\}$. Similarly, equation (5) holds if $\overline{P}$ contains D_X but does not contain E_X, hence, this leads to a contradiction. Therefore, T is secure.

Doubly verification is one of the key properties that ensures the balanced nature of symbols in the symbolic security proof of a protocol. A definition and theorem from [1] concerning the doubly verified cascade protocol are presented below.

Definition 2. *[1] A cascade protocol T is doubly verified if for some i,*

$$lt(\overline{N_i(X,Y)}) \subseteq \{E_Y, D_Y, D_X\}$$

and for some $j \geq 2$, $lt(\overline{N_j(X,Y)}) \subseteq \{E_X, D_X, D_Y\}$.

Theorem 2. *[1] Every doubly verified protocol is insecure.*

The proposed DY model in [1] proved the insecurity of the public key authentication using Diffie and Hellman by theorem 2. It emphasizes that after the first transmission, if X is not able to reconstruct M then the protocol T is doubly verified and insecure.

The choice of two distinct strings from the set $\{z_1, z_2, z_3, z_4\}^*$ plays a key role in the model proposed in [1]. The next section presents four new combinations formed by these strings along with their security proofs.

2.2 The characterization of Cascade protocol

The prior conditions (1) and (2) in theorem 1 are defined for the model of two-party symmetric cascade protocols of the DY model [1]. Here we consider remaining possibilities along with these conditions on the set from which the choices of strings $\tilde{\alpha_i}$ and $\tilde{\beta_i}$ are made. Following are all four possible cases of sets to choose a finite string for a two party cascade protocol:

1. **Case-1:** $\tilde{\alpha_i} \in \{z_1, z_2, z_3\}^*, \quad 1 \leq i \leq t,$

2. **Case-2:** $\tilde{\beta}_i \in \{z_1, z_2, z_4\}^*, \quad 1 \leq i \leq t'$,
3. **Case-3:** $\tilde{\rho}_i \in \{z_2, z_4, z_3\}^*, \quad 1 \leq i \leq t''$ and
4. **Case-4:** $\tilde{\delta}_i \in \{z_1, z_3, z_4\}^*, \quad 1 \leq i \leq t'''$,

For each pair of distinct users X and Y, let $\alpha_i(X,Y), \beta_i(X,Y), \rho_i(X,Y), \delta_i(X,Y)$ denote the string $\tilde{\alpha}_i, \tilde{\beta}_i, \tilde{\rho}_i, \tilde{\delta}_i$ respectively, where the symbols z_1, z_2, z_3, z_4 are replaced by E_X, E_Y, D_X, D_Y.

All of the cases 1 to 4 mentioned above provide a complete characterization of a two-party cascade protocol based on the capabilities of users X and Y. Interestingly, only five combinations can arise from the selection of two sets of strings among four cases of distinct strings $\tilde{\alpha}_i, \tilde{\beta}_i, \tilde{\rho}_i$ and $\tilde{\delta}_i$. In [1], the author defined one of those choices and presented the two-party cascade protocol T, specified it by $\left\{\tilde{\alpha}_i, \tilde{\beta}_i\right\}$ and proved that the defined protocol T is secure. Here we present the remaining four cases of two-party cascade protocols named as T_1, T_2, T_3 and T_4 with their complete characterization in terms of security analysis. A better understanding of each variant for two-party symmetric cascade protocols T_1 is presented here in figures 5. A similar analogy applies to the protocols T_2, T_3, and T_4.

Definition 3. *A two-party cascade protocol T_1 is specified by a series of finite strings $\{\tilde{\alpha}_i, \tilde{\rho}_l | 1 \leq i \leq t, 1 \leq l \leq t'\}$.*

$$\tilde{\alpha}_i \in \{z_1, z_2, z_3\}^*, \quad 1 \leq i \leq t$$

and

$$\tilde{\rho}_l \in \{z_2, z_4, z_3\}^*, \quad 1 \leq l \leq t'$$

where $t' = t$ or $t-1$. The symbol z_1, z_2, z_3, z_4 represents E_X, E_Y, D_X, D_Y. For two distinct users X and Y sequence of operators before transmitting messages is defined as,

$$N_1(X,Y) = \alpha_1(X,Y)$$
$$N_{2l} = \rho_l(X,Y) N_{2l-1}(X,Y), \quad 1 \leq l \leq t'$$
$$N_{2i+1} = \alpha_{i+1}(X,Y) N_{2i}(X,Y), \quad 1 \leq i \leq t-1.$$

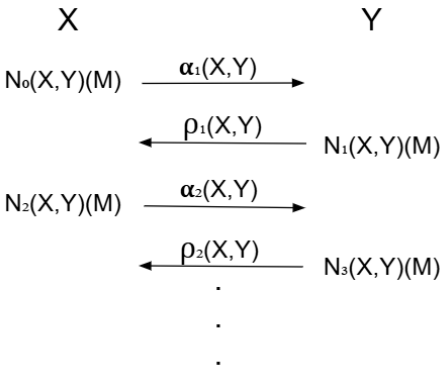


Fig. 5. A symmetric cascade protocol T_1

To define the security of the protocol, we first define the following sets.

Definition 4. *For two distinct users A and B a two-party cascade protocol T_1 is specified by a series of finite strings $\{\tilde{\alpha_i}, \tilde{\rho_l}\}$.*

$$\Sigma_1^1(Z) = E \cup \{D_Z\},$$

$$\Sigma_2^1 = \{\alpha_i(A,B)| \ \ for\ all \ \ A \neq B \ \ \ and \ \ \ i \geq 2\}$$

$$\Sigma_3^1 = \{\rho_i(A,B)| \ \ for\ all \ \ A \neq B \ \ \ and \ \ \ i \geq 1\}.$$

The corollary below provides the security for the two-party cascade protocol T_1.

Corollary 1. *A protocol T_1 is insecure if some $\gamma \in (\Sigma_1^1(Z) \cup \Sigma_2^1 \cup \Sigma_3^1)$ exists such that*

$$\overline{\gamma N_i(X,Y)} = \lambda.$$

Proof. Proof follows from the definition of T_1 and equation (3).

The balancing property for two party cascade protocol $T_1 = \{\tilde{\alpha_i}, \tilde{\rho_l}\}$ is defined below.

Definition 5. *Let X, Y be two distinct user names. A two party cascade protocol $T_1 = \{\tilde{\alpha_i}, \tilde{\rho_l}\}$ is a balanced cascade protocol if*

1. *for every $i \geq 2$, $\alpha_i(X,Y)$ has the balancing property with respect to X, and*
2. *for every $l \geq 1$, $\rho_l(X,Y)$ has the balancing property with respect to Y.*

In the below theorem, we prove that a two-party cascade protocol $T_1 = \{\tilde{\alpha_i}, \tilde{\rho_l}\}$ is secure with the following conditions.

Theorem 3. *A two-party cascade protocol $T_1 = \{\tilde{\alpha_i}, \tilde{\rho_l}\}$ is secure if and only if*

1. *$lt(\alpha_1(X,Y)) \cap \{E_Y, D_X\} \neq \phi$, and*
2. *T_1 is balanced.*

Proof. Let us consider Z be a user name distinct from X and Y.

(A). Necessity: Assume that either property 1) or 2) is not true, and have to show that T is insecure i.e., there exists $\gamma \in (\Sigma_1(Z) \cup \Sigma_2 \cup \Sigma_3)^*$ such that $\overline{\gamma N_i(X,Y)} = \lambda$ for some i.

If 1) is not true, i.e., $(E_X, E_Y, D_X) \cap (E_Y, D_X) = \phi$, this implies E_Y, D_X are not there in the intersection. A simpler, reduced form of $\{E_X, E_Y\}$ is constant. Since $\alpha_1(X,Y) = N_1(X,Y) = \{E_X, E_Y, D_X\}^*$, $\alpha_1^c = \{D_X, D_Y, E_X\}$, such that γ is a sequence of (E_Y, D_X). This implies $\gamma = \alpha_1^c = D_X \in (\Sigma_2^1 \cup \Sigma_3^1)^*$, hence $\overline{\gamma N_1(X,Y)} = \lambda$. Therefore, T_1 is not secure.

Now, assume 2) is false, i.e., T_1 is not balanced. By definition, either some $\rho_k(X,Y)$ contains D_Y but not E_Y or some $\alpha_i(X,Y)(i \geq 2)$ contains D_X but not E_X. We restrict ourselves to the former case ($\rho_k(X,Y)$ contains D_Y but not E_Y); the latter case will be treated later. Under this restriction, we will establish

the following stronger result. Since $\rho_k = (E_Y, D_Y, D_X)^*$, now based on the not balanced definition $\rho_k = (D_Y, D_X)^*$. For any $\delta \in (\{D_Y, D_X\} \cup E)^*$, there exists

$$\gamma \in \{\Sigma_1(Z) \cup (\rho_k(Z,X), \rho_k(Z,Y))\}^*$$

or

$$\gamma \in \{(E \cup D_Z) \cup \{(E_X, D_X, D_Z), (E_Y, D_Y, D_Z)\}\}^*$$

such that $\overline{\gamma\delta} = \lambda$.

The proof will be carried out by induction on r, the number of D_X and D_Y in the string δ. If $r = 0$ then $\gamma = \delta^c = D \in (\rho_k(Z,X), \rho_k(Z,Y))^*$, hence proved.

If $r > 0$, assume the result is true for all smaller values of r. δ is string containing exactly r number of D_X and D_Y. Without loss of generality, we can assume that the leftmost D is an D_Y. Write $\delta = \sigma_1 D_Y \sigma_2$, where $lt(\sigma_1) \cap (E_Y, D_X) = \phi$. Clearly, $\sigma_1^c \in (\rho_k(Z,X))^*$ (i.e., $\{D_X, E_Y\} \notin \sigma_1$ implies $\sigma_1 \in \{E_X, D_Z, E_Z, D_Y\}$ hence $\sigma_1^c \in (D_X, D_Z, E_Z, E_Y) \in (\rho_k(Z,X), \rho_k(Z,Y))^*$. By assuming $\rho_k(Z,Y)$ contains D_Y but not E_Y implies $\rho_k(Z,Y) = \tau_1 D_Y \tau_2$, where $\tau_1 \in (D_Z, D_Y)^*$ and $\tau_2 \in (D_Z)^*$, therefore $\tau_1^c \in (E_Z, E_Y)^*$ and $\tau_2^c \in (E_Z)^*$. Clearly, $\tau_i^c \in (\Sigma_1^1(Z))^*$ for $i = 1, 2$.

σ_2 contains $r-1$ D's by induction, there exists

$$\gamma' \in (\Sigma_1^1(Z) \cup \{\rho_k(Z,X), \rho_k(Z,Y)\})^*$$

such that $\overline{\gamma'\sigma_2} = \lambda$. Define $\gamma = \gamma' \tau_2^c \rho_k(Z,X) \tau_1^c \sigma_1^c$, then

$$\gamma \in (\Sigma_1^1(Z) \cup \{\rho_k(Z,X), \rho_k(Z,Y)\})^*.$$

From the above discussions, we get

$$\begin{aligned}
\overline{\gamma\delta} &= \overline{\gamma' \tau_2^c \rho_k(Z,X) \tau_1^c \sigma_1^c \sigma_1 E_Y \sigma_2} \\
&= \overline{\gamma' E_X E_X D_Z D_X E_X E_Y D_Y \sigma_2} \\
&= \overline{\gamma' E_X E_X \sigma_2} \\
&= \overline{\gamma' \sigma_2} \\
&= \lambda
\end{aligned} \tag{6}$$

(B). Sufficiency: Assuming that both properties 1) and 2) are satisfied, we have to prove that T_1 is secure. Suppose, to the contrary, there exists a $\gamma \in (\Sigma_1^1(Z) \cup \Sigma_2^1 \cup \Sigma_3^1)^*$ such that $\overline{\gamma N_i(X,Y)} = \lambda$ for some i. Write $\gamma N_i(X,Y) = P\alpha_1(X,Y)$ such that $P \in (\Sigma_1^1(Z) \cup \Sigma_2^1 \cup \Sigma_3^1)^*$. By definition of γ, we have

$$\overline{\overline{P}\alpha_1(X,Y)} = \lambda. \tag{7}$$

By the definition of a protocol, $lt(\alpha_1(X,Y)) \subseteq \{E_X, E_Y, D_X\}$. We distinguish two cases.

Case 1: $E_Y \in lt(\alpha_1(X,Y))$: As the string α_1 does not contain D_Y, the only possibility for (7) to hold is that $\overline{P}$ contains some D_Y but not E_Y. This implies that $\overline{P}$ does not have the balancing property with respect to Y. We know that

any string $\eta \in (\Sigma_1^1(Z) \cup \Sigma_2^1 \cup \Sigma_3^1)^*$, $\bar{\eta}$ has the balanced property with respect to every $Y \neq Z$, hence this contradicts.

Case 2: $E_Y \notin lt(\alpha_1(X,Y))$: In this case $D_X \notin lt(\alpha_1(X,Y))$, because $\alpha_1(X,Y) = (E_X, E_Y, D_X)$, and protocol satisfy the property that

$$lt(\alpha_1(X,Y)) \cap \{E_Y, D_X\} \neq \phi$$

$((E_X, D_X) \cap \{E_Y, D_X\} \neq \phi \implies \{1\} \cap \{E_Y, D_X\} \neq \phi$, which is contradiction for assumption 1). This implies $\alpha_1(X,Y) = E_X$. Similarly, equation (7) holds if $\bar{P}$ contains D_X and does not E_X, hence, this leads to a contradiction.

Therefore, T_1 is secure.

The result below defines the doubly verified property of protocol T_1.

Definition 6. *A cascade protocol is T_1 called doubly verified if for some i, $lt(\overline{N_i(X,Y)}) \subseteq \{E_Y, D_X, D_Y\}$ and for some $l \geq 2$, $lt(\overline{N_l(X,Y)}) \subseteq \{E_X, D_X, D_Y\}$.*

The following result proves that the doubly verified protocol T_1 is insecure.

Theorem 4. *Every doubly verified protocol T_1 is insecure.*

Proof. Let T_1 be a doubly verified protocol such that

$$lt(\overline{N_n}) \subseteq \{E_Y, D_X, D_Y\} \tag{8}$$

$$lt(\overline{N_m}) \subseteq \{E_X, D_X, D_Y\} \tag{9}$$

where $\overline{N_i} = N_i(X,Y)$. Write $\overline{N_1} = \alpha_1(X,Y)$, $\overline{N_n} = \overline{\gamma_n \alpha_1(X,Y)}$ and $\overline{N_m} = \overline{\gamma_m \alpha_1(X,Y)}$, where $\gamma_p \in (\Sigma_2^1 \cup \Sigma_3^1)^*$. Assuming T_1 is secure, we derive a contradiction. If T_1 is secure, then by Theorem 3, T_1 must be balanced.

Case 1. $E_Y \in lt(\alpha_1(X,Y))$: Clearly, to hold equation (9), $\overline{\gamma_m}$ should contain D_Y but not E_Y. This contradicts the fact that any string $\eta \in (\Sigma_1^1(Z) \cup \Sigma_2^1 \cup \Sigma_3^1)^*$, $\bar{\eta}$ has the balanced property with respect to every $Y \neq Z$.

Case 2. $E_X \in lt(\alpha_1(X,Y))$ and $E_Y \notin lt(\alpha_1(X,Y))$: In this case, equation (8) requires that, $\overline{\gamma_n}$ should contain D_X but not E_X. This contradicts the fact that, any string $\eta \in (\Sigma_1^1(Z) \cup \Sigma_2^1 \cup \Sigma_3^1)^*$, $\bar{\eta}$ has the balanced property with respect to every $X \neq Z$.

The definitions of the two-party cascade protocols and their corresponding balanceness properties for $T_2 = \left\{\tilde{\alpha_i}, \tilde{\delta_k}\right\}, T_3 = \left\{\tilde{\rho_l}, \tilde{\beta_j}\right\}$ and $T_4 = \left\{\tilde{\delta_k}, \tilde{\beta_j}\right\}$ are provided below.

Definition 7. *A two-party cascade protocol T_2 is specified by a series of finite strings $\left\{\tilde{\alpha_i}, \tilde{\delta_k}\right\}$.*

$$\tilde{\alpha_i} \in \{z_1, z_2, z_3\}^*, \quad 1 \leq i \leq t$$

and

$$\tilde{\delta_k} \in \{z_1, z_3, z_4\}^*, \quad 1 \leq k \leq t''',$$

where $t''' = t$ or $t-1$. The symbol z_1, z_2, z_3, z_4 represents E_X, E_Y, D_X, D_Y. Here, the sequence of operators before transmitting messages for two distinct users X and Y are defined as,

$$N_1(X,Y) = \alpha_1(X,Y)$$

$$N_{2k} = \delta_k(X,Y)N_{2k-1}(X,Y), \quad 1 \le k \le t'$$

$$N_{2i+1} = \alpha_{i+1}(X,Y)N_{2i}(X,Y), \quad 1 \le i \le t-1.$$

To define the security of the protocol, we need to define the following sets.

Definition 8. *For two distinct users A and B a two-party cascade protocol T_2 is specified by a series of finite strings $\left\{\tilde{\alpha_i}, \tilde{\delta_k}\right\}$.*

$$\Sigma_1^2(Z) {}^{=} E \cup \{D_Z\},$$

$$\Sigma_2^2 = \{\alpha_i(A,B) | \ \textit{for all} \ \ A \neq B \ \ \textit{and} \ \ i \ge 2\}$$

$$\Sigma_3^2 = \{\delta_i(A,B) | \ \textit{for all} \ \ A \neq B \ \ \textit{and} \ \ i \ge 1\}.$$

The corollary below provides the security for the two-party cascade protocol T_2.

Corollary 2. *A protocol T_2 is insecure if some $\gamma \in (\Sigma_1^2(Z) \cup \Sigma_2^2 \cup \Sigma_3^2)$ exists such that*

$$\overline{\gamma N_i(X,Y)} = \lambda.$$

Proof. Proof follows from the definition of T_2 and equation (3).

Definition 9. *A two-party cascade protocol $T_3 = \left\{\tilde{\rho_l}, \tilde{\beta_j}\right\}$ is specified by a series of finite strings $\left\{\tilde{\rho_l}, \tilde{\beta_j}\right\}$.*

$$\tilde{\rho_l} \in \{z_2, z_4, z_3\}^*, \quad 1 \le l \le t''$$

and

$$\tilde{\beta_j} \in \{z_1, z_2, z_4\}^*, \quad 1 \le j \le t',$$

where $t' = t''$ or $t''-1$. The symbol z_1, z_2, z_3, z_4 represents E_X, E_Y, D_X, D_Y. The sequence of operators before transmission of the message for two distinct users X and Y is presented as,

$$N_1(X,Y) = \rho_1(X,Y)$$

$$N_{2j} = \beta_j(X,Y)N_{2j-1}(X,Y), \quad 1 \le j \le t'$$

$$N_{2l+1} = \rho_{l+1}(X,Y)N_{2l}(X,Y), \quad 1 \le l \le t-1.$$

To define the security of the protocol, we have to define the following sets.

Definition 10. *For two distinct users A and B two-party cascade protocol T_3 is specified by a series of finite strings $\left\{\tilde{\rho_l}, \tilde{\beta_j}\right\}$.*

$$\Sigma_1^3(Z) = E \cup \{D_Z\},$$

$$\Sigma_2^3 = \{\rho_i(A,B)| \ \ for\ all \ \ A \neq B \ \ \ and \ \ i \geq 2\}$$
$$\Sigma_3^3 = \{\beta_i(A,B)| \ \ for\ all \ \ A \neq B \ \ \ and \ \ i \geq 1\}.$$

The corollary below provides the security for the two-party cascade protocol T_3.

Corollary 3. *A protocol T_3 is insecure if some $\gamma \in (\Sigma_1^3(Z) \cup \Sigma_2^3 \cup \Sigma_3^3)$ exists such that*

$$\overline{\gamma N_j(X,Y)} = \lambda$$

Proof. Proof follows from the definition of T_3 and equation (3).

Definition 11. *A two-party cascade protocol T_4 is specified by a series of finite strings $\left\{\tilde{\delta_k}, \tilde{\beta_j}\right\}$.*

$$\tilde{\delta_k} \in \{z_1, z_3, z_4\}^*, \quad 1 \leq k \leq t''',$$

and

$$\tilde{\beta_j} \in \{z_1, z_2, z_4\}^*, \quad 1 \leq j \leq t',$$

where $t' = t'''$ or $t'''-1$. The symbol z_1, z_2, z_3, z_4 represents E_X, E_Y, D_X, D_Y, and the following is the sequence of operators before transmission of the message,

$$N_1(X,Y) = \delta_1(X,Y)$$

$$N_{2j} = \beta_j(X,Y)N_{2j-1}(X,Y), \quad 1 \leq j \leq t'$$
$$N_{2k+1} = \delta_{k+1}(X,Y)N_{2k}(X,Y), \quad 1 \leq k \leq t-1.$$

Definition 12. *For two distinct users A and B a two-party cascade protocol T_4 is specified by a series of finite strings $\left\{\tilde{\delta_k}, \tilde{\beta_j}\right\}$.*

$$\Sigma_1^4(Z) = E \cup \{D_Z\},$$

$$\Sigma_2^4 = \{\delta_i(A,B)| \ \ for\ all \ \ A \neq B \ \ \ and \ \ i \geq 2\}$$
$$\Sigma_3^4 = \{\beta_i(A,B)| \ \ for\ all \ \ A \neq B \ \ \ and \ \ i \geq 1\}.$$

The corollary below provides the security for the two-party cascade protocol T_4.

Corollary 4. *A protocol T_4 is insecure if some $\gamma \in (\Sigma_1^4(Z) \cup \Sigma_2^4 \cup \Sigma_3^4)$ exists such that*

$$\overline{\gamma N_k(X,Y)} = \lambda.$$

Proof. Proof follows from the definition of T_4 and equation (3).

The balancing property for the two two-party cascade protocols T_2, T_3, T_4 are defined below.

Definition 13. *Let X, Y be two distinct user names.*

1. *A two party cascade protocol $T_2 = \left\{\tilde{\alpha_i}, \tilde{\delta_k}\right\}$ is a balanced cascade protocol if $\alpha_i(X,Y)$ has the balancing property with respect to X for every $i \geq 2$, and $\delta_k(X,Y)$ has the balancing property with respect to Y for every $k \geq 1$.*
2. *A two party cascade protocol $T_3 = \left\{\tilde{\rho_l}, \tilde{\beta_j}\right\}$ is a balanced cascade protocol if $\rho_l(X,Y)$ has the balancing property with respect to X for every $l \geq 2$, and $\beta_j(X,Y)$ has the balancing property with respect to Y for every $j \geq 1$.*
3. *A two party cascade protocol $T_4 = \left\{\tilde{\delta_k}, \tilde{\beta_j}\right\}$ is a balanced cascade protocol if $\delta_k(X,Y)$ has the balancing property with respect to X for every $k \geq 2$, and $\beta_j(X,Y)$ has the balancing property with respect to Y for every $j \geq 1$.*

In the below theorem, we prove that a two-party cascade protocol $T_2 = \left\{\tilde{\alpha_i}, \tilde{\delta_k}\right\}$ is insecure.

Theorem 5. *A two-party cascade protocol $T_2 = \left\{\tilde{\alpha_i}, \tilde{\delta_k}\right\}$ is insecure.*

Proof. $\delta_k(X,Y)$ is not balanced with respect to Y, for every $k \geq 1$ and theorem 1 implies the conclusion.

In the below theorem, we prove that a two-party cascade protocol $T_3 = \left\{\tilde{\rho_l}, \tilde{\beta_j}\right\}$ is insecure.

Theorem 6. *A two-party cascade protocol $T_3 = \left\{\tilde{\rho_l}, \tilde{\beta_j}\right\}$ is insecure.*

Proof. $\rho_l(X,Y)$ is not balanced with respect to X, for every $l \geq 2$ and theorem 1 implies the conclusion.

In the below theorem, we prove that a two-party cascade protocol $T_4 = \left\{\tilde{\delta_k}, \tilde{\beta_j}\right\}$ is secure with the following conditions.

Theorem 7. *A two-party cascade protocol $T_4 = \left\{\tilde{\delta_k}, \tilde{\beta_j}\right\}$ is secure if and only if*

1. *$lt(\delta_1(X,Y)) \cap \{E_Y, E_X\} \neq \phi$, and*
2. *T_4 is balanced.*

Proof. The proof proceeds by the same argument as that employed in Theorem 3.

The result below defines the doubly verified property of protocol T_4.

Definition 14. *A cascade protocol* $T_4 = \left\{\tilde{\delta_k}, \tilde{\beta_j}\right\}$ *is called doubly verified if for some* k, $lt(\overline{N_k(X,Y)}) \subseteq \{D_Y, E_X, E_Y\}$ *and for some* $j \geq 2$, $lt(\overline{N_l(X,Y)}) \subseteq \{D_X, E_X, E_Y\}$.

The following result proves that the doubly verified protocol T_4 is insecure.

Theorem 8. *Every doubly verified protocol* T_4 *is insecure.*

Proof. Let T_4 be a doubly verified protocol such that

$$lt(\overline{N_m}) \subseteq \{D_Y, E_X, E_Y\} \tag{10}$$

$$lt(\overline{N_m}) \subseteq \{D_X, E_X, E_Y\} \tag{11}$$

where $\overline{N_i} = N_i(X,Y)$. Write $\overline{N_1} = \delta_1(X,Y)$, $\overline{N_n} = \overline{\gamma_n \delta_1(X,Y)}$ and $\overline{N_m} = \overline{\gamma_m \delta_1(X,Y)}$, where $\gamma_p \in (\Sigma_2^4 \cup \Sigma_3^4)^*$. Assuming T_1 is secure, we derive a contradiction. If T_4 is secure, then by Theorem 7, T_4 must be balanced.

Case 1. $D_Y \in lt(\delta_1(X,Y))$: Clearly, to hold equation (11), $\overline{\gamma_m}$ should contain E_Y but not D_Y. This contradicts the fact that any string $\eta \in (\Sigma_1^4(Z) \cup \Sigma_2^4 \cup \Sigma_3^4)^*$, $\bar{\eta}$ has the balanced property with respect to all $Y \neq Z$.

Case 2. $D_X \in lt(\delta_1(X,Y))$ and $D_Y \notin lt(\delta_1(X,Y))$: In this case, equation (10) requires that, $\overline{\gamma_n}$ should contain E_X but not D_X. This contradicts the fact that, any string $\eta \in (\Sigma_1^4(Z) \cup \Sigma_2^4 \cup \Sigma_3^4)^*$, $\bar{\eta}$ has the balanced property with respect to every $X \neq Z$.

3 Conclusion

The assumptions of the DY model are based on the capabilities of users. The security and verification of two user cases have been presented with the presence of an attacker Z. In this paper, we presented all four generalized cases of two-party cascade protocols denoted T_1, T_2, T_3 and T_4, and provided their complete characterization with respect to security analysis. It is evident from the definitions of protocols T_2 and T_3 that a doubly verified protocol cannot be defined in these cases. Further, we outlined the security conditions for two cases of the protocol T_1 and T_4. Consequently, exploring the security of generalized versions of the symmetric cascade and name-stamp protocols for scenarios involving more than two users for future work.

References

1. Dolev D., and Yao A. C., On the security of public key protocols. Proceedings of the 22nd Annual Symposium on Foundations of Computer Science (FOCS), 350–357, (1983).
2. Book R. V., Otto F., The verifiability of two-party protocols. In Advances in Cryptology – EUROCRYPT '85 (pp. 233–242). Springer.
3. Blanchet, B. (2001). Automatic verification of correspondences for security protocols. Proceedings of the IEEE Computer Security Foundations Workshop (CSFW), (1985).

4. Rakotonirina I., Barthe G., and Schneidewind C., Decision and complexity of Dolev-Yao hyperproperties. Proceedings of the ACM on Programming Languages, 8(POPL), Article 64, 1913–1944, (2024).
5. Kemmerer R. A., Analyzing encryption protocols using formal verification techniques. IEEE Journal on Selected Areas in Communications, 7(4), 448–457, (1989).
6. Paulson L. C., The inductive approach to verifying cryptographic protocols. Journal of Computer Security, 6(1–2), 85–128, (1998).
7. Véronique C., Steve K., Bogdan W., ASurvey of Symbolic Methods in Computational Analysis of Cryptographic Systems, Journal of Autom Reasoning, 46, 225–259, (2011). 10.1007/s10817-010-9187-9
8. Abadi M., Rogaway P., Reconciling Two Views of Cryptography (The Computational Soundness of Formal Encryption). J. Cryptology 15, 103–127, (2002). https://doi.org/10.1007/s00145-001-0014-7
9. Blanchet B., A computationally sound automatic prover for cryptographic protocols. Workshop on the Link Between Formal and Computational Models, Paris, France, (2005).
10. Blanchet B., Modeling and verifying security protocols with the applied Pi calculus and ProVerif. Foundations and Trends in Privacy and Security, 1(1-2), 1–135, (2016).
11. Bhargavan K., Cheval V., Wood C., A symbolic analysis of privacy for TLS 1.3 with encrypted client hello. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS '22) (pp. 365–379). Association for Computing Machinery, (2022). https://doi.org/10.1145/3548606.3559360