Unlinkability of an Improved Key Agreement Protocol for EMV 2nd Gen Payments

Ross Horne
Department of Computer Science
University of Luxembourg
Esch-sur-Alzette, Luxembourg
ross.horne@uni.lu

Sjouke Mauw
Department of Computer Science
University of Luxembourg
Esch-sur-Alzette, Luxembourg
sjouke.mauw@uni.lu

Semen Yurkov[†]
Department of Computer Science
University of Luxembourg
Esch-sur-Alzette, Luxembourg
semen.yurkov@uni.lu

***Abstract*—To address known privacy problems with the EMV standard, EMVCo have proposed a *Blinded Diffie-Hellman* key establishment protocol, which is intended to be part of a future 2nd Gen EMV protocol. We point out that active attackers were not previously accounted for in the privacy requirements of this proposal protocol, and demonstrate that an active attacker can compromise unlinkability within a distance of 100cm. Here, we adopt a strong definition of unlinkability that does account for active attackers and propose an enhancement of the protocol proposed by EMVCo. We prove that our protocol does satisfy strong unlinkability, while preserving authentication.**

***Index Terms*—unlinkability, authentication, key agreement, protocols, bisimilarity**

I. Introduction

The majority of payment cards and terminals use the EMV standard [1], the set of protocols developed by the union of payment processing companies Europay, Mastercard and Visa to execute financial operations. The initial purpose of EMV, introduced in 1996, was to support the replacement of magstripe cards with integrated circuit cards that are harder to copy. The EMV standard now supports contactless cards, that require no cardholder action to be involved in the EMV session with any capable device. The nature of contactless cards allows an active attacker to easily interact with the card without the cardholder realising, making privacy properties harder to enforce.

In this paper, we address privacy vulnerabilities in payment cards with a particular focus on the *unlinkability* of payments. The EMV standard trivially does not satisfy privacy properties such as anonymity and unlinkability. This is due to the current EMV standard transferring the card number in cleartext during a transaction. Hence transaction data allows us to link transactions made with the same card and effortlessly track cardholders. The fact that no actual payments need to be made eases the task of the adversary when tracking a contactless card as it is ready to present its identity to any device.

In 2011 EMVCo launched the development of the new version of the standard, the EMV 2nd Gen, where the card should be protected against eavesdropping. To facilitate this, EMVCo proposed the use of *secret channels*. A secret channel is a symmetric key that the card and the terminal establish at the start of each session and use to encrypt further communications. A channel establishment procedure is based on Diffie-Hellman key agreement with a twist: the card uses a freshly blinded static certified public key instead of an ephemeral public key. Hence, the name of the proposed protocol, *Blinded Diffie-Hellman* (BDH) [2].

The BDH protocol is meant to satisfy the official requirements for channel establishment from the architecture overview of the EMV 2nd Gen [3]:

- Use elliptic-curve based cryptography (ECC).
- Computational resources of the card are respected.
- An attacker who passively eavesdrops on communications cannot identify a particular card.

Several authors published a security proof for the Blinded Diffie-Hellman protocol [4], [5] and established that a passive eavesdropper, that only listens to transmitted messages, cannot reidentify a card, therefore BDH satisfies the above requirements. Brzuska, Smart, Warinschi, and Watson [4] named this property of BDH “external unlinkability”.

We discuss a potential strengthening of the requirements for BDH listed above, i.e. we lift the limitation of attackers being passive. In the context of contactless payments such requirement is a realistic one, since it is easy for an attacker to initiate sessions with contactless cards using devices, such as smartphones, that need not be official terminals. In fact, different capabilities must be considered in a wireless environment depending on their distance from the card. It is difficult to perform an eavesdropping attack outside of the approximately 20m radius [6]. However, successful attacks executed by *a passive attacker* are reported within the range between 20m and 100cm [7], [8] and, with the right equipment, within 100cm *an active attacker* can power up the card and start communication [9]. A close active attacker is a real threat to the privacy of anyone having a card in their pocket, since the distance of 100cm is easily achievable, e.g. at doorways or checkouts.

Unsurprisingly, the proposed BDH protocol is no longer secure in such strictly stronger threat model. To accommodate our strengthened threat model we consider an enhancement of the proposed protocol that is unlinkable and untraceable. The definition of unlinkability we propose is formalised as

[†] Semen Yurkov is supported by the Luxembourg National Research Fund through grant PRIDE15/10621687/SPsquared.

a process equivalence problem in the applied π-calculus and accommodates active attackers. Our enhancement of BDH uses a generic *anonymous credentials* scheme to hide the card's identity. At least one anonymous credential scheme, Verheul certificates, is known to respect the limited computing power of smart cards making our upgrade minor.

To define unlinkability in the context of EMV payments, we are building on the state-of-the-art bisimilarity-based approach developed by Horne and Mauw [10]. The use of quasi-open bisimilarity [11] in our new definition has the following two reasons behind it.

- It helps to reduce drastically the amount of work needed for verification: being a congruence relation, quasi-open bisimilarity allows us to take into account cards only, since not having a common secret between cards and terminals is a part of the philosophy of EMV.
- It ensures that our results hold also in weaker models like trace equivalence: being a bisimilarity, quasi-open bisimilarity is a strictly finer equivalence.

The contributions of the work are as follows.

- A new definition for unlinkability suitable for EMV payments that accounts for active attackers.
- An attack invalidating our strong unlinkability goal on Blinded Diffie-Hellman, in the form initially proposed by EMVCo described by a modal logic formula.
- An improved proposal for the Blinded Diffie-Hellman protocol by integrating blind certificates.
- The proof of the unlinkability of our improved BDH.
- A discussion on the unlinkable, in our stronger sense, EMV transactions.

The paper is organised as follows. Section II is devoted to previous work on EMV security and privacy issues. In Section III, we present the original Blinded Diffie-Hellman and illustrate why there are attacks on unlinkability in the presence of an active attacker. In Section IV, we provide background on the applied π-calculus. Sections V and VI contain the main contributions of the paper: a new definition of unlinkability for EMV payments, an enhanced version of BDH that includes blinded certificates, and a proof of the unlinkability of this enhanced version. Section VII confirms that authentication properties are preserved by our enhanced protocol. In Section VIII we discuss unlinkable transactions with respect to our strong threat model and explain that it would be impossible without touching the fundamentals of the current EMV infrastructure, since the account number would eventually be received by the dishonest terminal. Section IX concludes the paper and presents directions for future research.

II. Related work

Much related work on EMV is concerned with authentication and secrecy problems essential for avoiding fraudulent payments. The recent works of Basin, Sasse, and Toro-Pozo [12], [13] contain an overview of attacks on EMV that can lead to fraudulent transactions, e.g. criminals can make high-value purchases using a contactless Visa card without knowing the PIN. Contactless specific *relay* attacks may be mitigated by using distance-bounding techniques [14], for instance Chothia, de Ruiter and Smyth verified Mastercard's RRP protocol [15], Boureanu et al. analyse relay-resistance EMV-based protocols in the presence of rogue readers [16], and Radu et al. combine man-in-the-middle replay and relay attacks to bypass the Apple Pay (working with Visa card) lock screen [17]. In a *skimming attack*, an attacker secretly activates a contactless card and communicates with it. A skimming attack may be a part of a relay attack and serves as the basis of the attack on the unlinkability of BDH we present in this paper. Habraken et al. constructed an antenna in the form of a gate of up to 100cm width that can power the card and communicate with it [9]. For a passive counterpart of skimming, an *eavesdropping attack*, Engelhardt et al. achieved the distance of 18m [8].

To enhance privacy in EMV it is natural to consider anonymous credentials systems, since they allow credentials to be verified without disclosing the identity of a cardholder; although such mechanisms have not been explored in the context of EMV payments. Idemix [18] and U-Prove [19] are general-purpose examples of such systems. However, they barely fit the context of this paper since Idemix requires a large key size, therefore implementation on smart-cards is rather slow [20] and it is straightforward to link transactions in U-Prove when the same credentials (in our case, the card's identity) are used twice. A more suitable anonymous credential system is the self-blindable attribute certificates due to Verheul [21]. Verheul certificates use elliptic curve cryptography, aligning with stated requirements of EMV 2nd Gen, and have been demonstrated to be efficiently implementable on smart cards [22].

Arapinis et al. [23] proposed to express unlinkability as an equivalence problem; specifically, they defined *strong unlinkability* using bisimulation. Horne and Mauw [10] propose a new scheme by adding session channels to the model and thoroughly study the advantages of the bisimilarity approach. We partially employ the formulation of the π-calculus presented in their work. Hirschi, Baelde, and Delaune [24] weakened the definition from [23] by redefining unlinkability as a trace equivalence problem for which they develop tool support for obtaining proofs of unlinkability. Using trace equivalence, however, may lead to missing attacks as pointed out in [25] where Filimonov et al. study ePassport protocols and revisit bisimilarity-based strong unlinkability definitions. There is an ongoing debate [24], [25] on the benefits of each equivalence, but in this work either is appropriate since we prove properties in the strongest of these models and find attacks in the weakest.

Finally, we mention works on symbolic methods for analysing Diffie-Hellman (DH) groups. The general case requires both exponentiation and the group operation to be modelled and a straightforward approach may lead to the unification problem in a field [26] which is undecidable. Tools like Tamarin [27] or ProVerif [28] use prime order group abstractions to facilitate verification. Cremers and Jackson investigate in detail the subtleties of modelling DH groups in automated tools and propose improved models in [29].

III. Blinded Diffie-Hellman and external unlinkability

In this motivating section, we introduce the Blinded Diffie-Hellman protocol from the original EMVCo request for comments [2] and highlight its unlinkability issues.

A. The Blinded Diffie-Hellman protocol

To present the BDH protocol we define the syntax of messages in Fig. 1.

$M, N ::=$	g	DH group generator (constant)
$\mid$	x	variable
$\mid$	$M \cdot N$	multiplication
$\mid$	$\phi(M, N)$	scalar multiplication
$\mid$	$\mathtt{pk}(M)$	public key
$\mid$	$\mathtt{sig}(M, N)$	signature
$\mid$	$\mathtt{h}(M)$	hash (for key derivation)
$\mid$	$\langle M,\, N \rangle$	pair
$\mid$	$\{M\}_N$	symmetric encryption
$\mid$	$\mathtt{check}(M, N)$	check signature
$\mid$	$\mathtt{fst}(M)$	get first
$\mid$	$\mathtt{snd}(M)$	get second
$\mid$	$\mathtt{dec}(M, N)$	symmetric decryption
$\mid$	$\mathtt{auth}$	authenticate

Fig. 1: Blinded Diffie-Hellman syntax.

The syntax for messages includes abstractions for the arithmetic operations on elliptic curves that protocols in this paper employ, enabling us to represent protocols symbolically. We leave the cryptographic details for multiplication, scalar multiplication and public key operations together with ECC domain parameters as a footnote[1]. The exact signing mechanism modelled by $\mathtt{sig}(M, N)$ is not specified by EMVCo in the proposal [2]. Hash, pair and encryption are standard and $\mathtt{auth}$ is a message that upon being output indicates that the terminal believes it has authenticated the card. The authentication property is explained in Section VII.

The equational theory E_0 axiomatising the properties of the cryptographic functions is given in Fig. 2 The first three equations capture the interaction between field arithmetic and scalar multiplication followed by standard destructors: projections, decryption, and signature check. Notice that we model signature verification in a manner that is standard when symbolically verifying protocols: the signature is verified iff the message is successfully extracted by applying $\mathtt{check}$ from $\mathtt{sig}(K, M)$ using the corresponding public key $\mathtt{pk}(K)$.

$$
\begin{aligned}
&M \cdot N =_{E_0} N \cdot M \\
&(M \cdot N) \cdot K =_{E_0} M \cdot (N \cdot K) \\
&\phi(M \cdot N, K) =_{E_0} \phi(M, \phi(N, K)) \\
&\mathtt{fst}(\langle M,\, N \rangle) =_{E_0} M \\
&\mathtt{snd}(\langle M,\, N \rangle) =_{E_0} N \\
&\mathtt{dec}(\{M\}_K\,, K) =_{E_0} M \\
&\mathtt{check}(\mathtt{sig}(M, K)\,, \mathtt{pk}(K)) =_{E_0} M
\end{aligned}
$$

Fig. 2: Equational theory E_0 for the Blinded Diffie-Hellman protocol.

The Blinded Diffie-Hellman protocol is presented in Fig. 3. There are two honest agents in the system that participate in the execution of the protocol: the card C and the terminal T. The payment system holds a secret key s and acts as a certification authority. The private key c, the public key $\phi(c, \mathrm{g})$ and the certificate $\langle \phi(c, \mathrm{g}),\ \mathtt{sig}(\phi(c, \mathrm{g})\,, s)\rangle$ are permanently embedded in the card when it is manufactured. The card can only be issued by the bank in cooperation with payment systems like Amex, Visa, etc. The terminal, in contrast to the card, can be manufactured by anyone. To verify the legitimacy of the card, the terminal uses a public key of the payment system $\mathtt{pk}(s)$ that is available on the system's website.

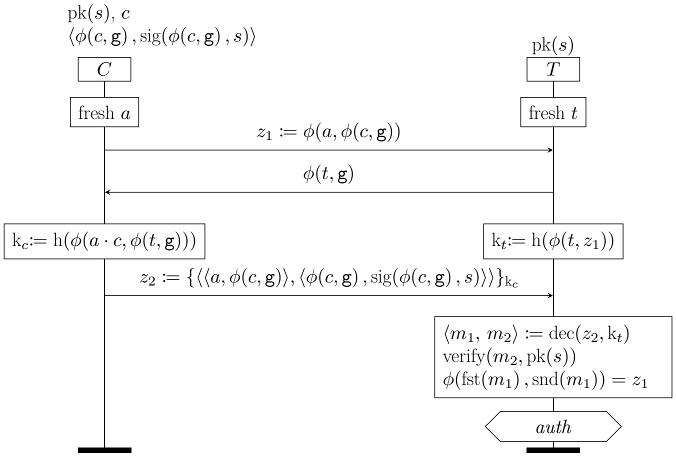
pk(s), c
⟨ϕ(c, g) , sig(ϕ(c, g) , s)⟩
pk(s)
C
T
fresh a
fresh t
z₁ := ϕ(a, ϕ(c, g))
ϕ(t, g)
k_c:= h(ϕ(a · c, ϕ(t, g)))
k_t:= h(ϕ(t, z₁))
z₂ := {⟨⟨a, ϕ(c, g)⟩, ⟨ϕ(c, g) , sig(ϕ(c, g) , s)⟩⟩}_{k_c}
⟨m₁, m₂⟩ := dec(z₂, k_t)
verify(m₂, pk(s))
ϕ(fst(m₁) , snd(m₁)) = z₁
auth

Fig. 3: EMV 2nd Gen key establishment.

The card starts the communication by sending its public key $\phi(c, \mathrm{g})$ blinded with a fresh scalar a to the terminal. In response, the terminal sends ephemeral public key $\phi(t, \mathrm{g})$ to the card. This is enough to establish a common secret key k_c= k_t. The card uses this key to encrypt the authentication data: blinding scalar a, static public key $\phi(c, \mathrm{g})$, and the certificate $\langle \phi(c, \mathrm{g})\,, \mathtt{sig}(\phi(c, \mathrm{g})\,, s)\rangle\rangle$. Finally, the terminal verifies the received certificate by checking the signature against the public key of the payment system $\mathtt{pk}(s)$, checks that $\phi(c, \mathrm{g})$ blinded with a coincides with the first message z_1 received from the card. Upon success, the terminal authenticates the card and is ready to continue with the transaction on the encrypted channel.

[1] The public parameters are as follows: a finite field $\mathbb{F}_p$; a Diffie-Hellman group G, defined over an elliptic curve $E(\mathbb{F}_p)$; the (prime) order q of G; the generator $\mathrm{g} \in G$; the key-derivation function h; the public key of the payment system $\mathtt{pk}(s)$ for the certificate verification. We employ (left) group action notation $\phi\colon \mathbb{F}_q^\times \times G \to G$ for group operation: we write $\phi(r, Q)$ for the element Q added with itself r times and call ϕ *scalar multiplication*. The symbol $\cdot$ denotes multiplication between two scalars (field elements). All freshly generated values are picked uniformly at random from $\mathbb{F}_q$. The secret key k is an element of $\mathbb{F}_q$ and the corresponding public key is of the form $\phi(k, \mathrm{g})$. Blinding of the element Q uses a fresh scalar a and internally works as a scalar multiplication: $\phi(a, Q)$.

B. Blinded Diffie-Hellman and active attackers

In order to verify that blinding the card's public key protects against eavesdroppers external to the execution, the property of *external unlinkability* was introduced [4]. In an externally unlinkable payment system, an attacker observing a message exchange between a card and a terminal cannot link that card's current session with a previous session from the same card.

In the real world, anyone could build a device imitating the terminal, for instance, an app on a smartphone supporting NFC or a skimming gate [9]. Such a device need not be certified or connected to any bank. Taking this into account, there is a straightforward attack on the BDH protocol (Fig. 3) in the presence of malicious terminals:

1) A malicious terminal establishes a key with an honest card, then successfully decrypts the message z_2 and obtains the card's public key $\phi(c, \mathrm{g})$.
2) Another terminal operated by the attacker runs a new session with the same card to obtain again the card's public key $\phi(c, \mathrm{g})$; and hence recognises the card.

This attack however would not be considered to be an attack on external unlinkability, due to the fact that, at the second step, the attacker actively starts comunicating with the card. Since it is easy to activate a contactless card, e.g. while the card is in the wallet, external unlinkability is too weak. This compels us to adopt a stronger notion of unlinkability which can be used to discover the above attack formally.

The above attack suggests that any network of malicious powerful terminal-like devices unrelated to any payment system may track selected contactless cards in real-time without the cardholder being aware simply by starting sessions with the card in the cardholder's pocket. Thus we propose to view unlinkability as a *property of the card* in a hostile environment that should hold with or without the presence of honest terminals. The attack also highlights why the BDH protocol is not unlinkable in the presence of active attackers – the ability of the terminal to obtain the card's public key which serves as the card's identity.

To address the unlinkability vulnerability highlighted above, we propose to modify the Blinded Diffie-Hellman protocol in such a way that the signature may also be blinded and hence the public key need never be revealed to a terminal in order to check the signature. We will present and verify our enhanced version of BDH in Section VI. However, first, we dedicate the next two sections to the machinery required to formally specify and verify that our proposal is unlinkable.

IV. Applied π-calculus and quasi-open bisimilarity

This section contains background on a state-of-the-art formulation of the applied π-calculus [30], a language for modelling concurrent processes and their interactions. The calculus is presented in a reduced form that is just enough for the purpose of the paper. We start with the syntax and move towards the definition of an equivalence relation on processes that we use to express the unlinkability definition in Section V.

A. Syntax, notation, conventions

The syntax of processes is presented in Fig. 4.

$$\begin{array}{rcll}
P, Q & ::= & 0 & \text{deadlock} \\
 & | & \overline{M}\langle N\rangle.P & \text{send} \\
 & | & M(y).P & \text{receive} \\
 & | & \nu x.P & \text{new} \\
 & | & P \mid Q & \text{parallel} \\
 & | & !P & \text{replication} \\
 & | & \mathtt{if}\, M = N\, \mathtt{then}\, P & \text{match}
\end{array}$$

Fig. 4: A syntax for processes in applied π-calculus processes.

Processes are used to capture the behaviour of a system, and, in particular, a behaviour of honest parties during the execution of a protocol. Processes can output and consume messages. To do that they use *channels*, e.g. $\overline{M}\langle N\rangle$ means that the message N is sent out on the channel M. Messages can be defined with respect to any message language (e.g. in Fig. 1) subject to any equational theory (e.g. in Fig. 2). We write $M =_E N$ for equality modulo an equational theory E.

Variables in processes may be *bound* by new name binders or inputs: specifically $\nu x.P$ and $M(x).P$ bind x in the scope P. In other words, the variable x becomes local to the process P. If a variable is not bound, it is a *free* variable. We denote by $\mathrm{fv}(T)$ the set of free variables in a process or a message term T.

The processes P and Q in $P \mid Q$ run concurrently. The replication $!P$ is an infinite parallel composition of P with itself. Finally, the process $\mathtt{if}\, M = N\, \mathtt{then}\, P$ can behave as P whenever $M =_E N$.

A *substitution* is a function from a finite set of variables to message terms. We use vector notation to indicate the list of variables $\vec{x}$ or messages $\vec{M}$. Whenever $\vec{x}$ is involved in set-theoretic operations we treat $\vec{x}$ as the set of variables in $\vec{x}$. We use σ, ρ and θ to refer to substitutions and write $x\sigma$ for σ applied to the variable x. The result of applying the substitution σ to the process P is the replacement of any free occurrence of x in P with $x\sigma$. We write $P\sigma$ for the resulting process. When σ is given explicitly, we write $\sigma = \left\{{}^{\vec{M}}/_{\vec{x}}\right\}$. Substitutions must avoid capture of bound variables: if a bound variable x in the process P occurs in the range of σ, it must be renamed to avoid a name clash. The renaming of bound variables is a standard operation in the π-calculus known as α-conversion [31] and we always consider processes up to α-conversion. For instance, to compute $a(x).\overline{a}\langle\{\langle x, y\rangle\}_k\rangle\left\{{}^{\mathrm{h}(x)}/_{y}\right\}$ we apply α-conversion first and get $a(z).\overline{a}\langle\{\langle z, y\rangle\}_k\rangle\left\{{}^{\mathrm{h}(x)}/_{y}\right\}$, where z is chosen fresh for $\mathrm{h}(x)$ and $\overline{a}\langle\{\langle x, y\rangle\}_k\rangle$, i.e., $z \notin \{a, x, y, k\}$, and then apply the substitution to obtain the result $a(z).\overline{a}\langle\{\langle z, \mathrm{h}(x)\rangle\}_k\rangle$.

We generalise the concept of a variable not belonging to some set of variables in the following definition.

Definition 1. (fresh, #) *The set of variables $\vec{x}$ is fresh for the set of variables $\vec{y}$ if $\vec{x} \cap \vec{y} = \varnothing$; $\vec{x}$ is fresh for a term P if $\vec{x}$*

is fresh for $\mathrm{fv}(P)$*;* $\vec{x}$ *is fresh for a substitution* σ *whenever* $\vec{x}$ *is fresh for* $\mathrm{dom}(\sigma)$ *and fresh for* $\mathrm{fv}(y\sigma)$ *for any* y *fresh for* $\vec{x}$*. Notation:* $\vec{x} \# \vec{y}$*,* $\vec{x} \# P$*,* $\vec{x} \# \sigma$*.*

That is, fresh variables never appear in the set of free variables or the domain and the range of the substitution.

Throughout the paper we use several conventions. We do not distinguish between $\nu x_1.\nu x_2.P$ and $\nu x_1.(\nu x_2.P)$ and typically write $\nu x_1, x_2.P$. The symbol $\triangleq$ is used to define a process. For readability purposes we introduce the following abbreviations.

$$\begin{array}{l}
\mathtt{let}\, x \coloneqq M \;\mathtt{in}\; P \;\triangleq\; P\left\{{}^{M}\!/_{x}\right\} \\
\mathtt{let}\, \langle x_1, x_2\rangle = M \;\mathtt{in}\; P \triangleq P\left\{{}^{\mathtt{fst}(M),\mathtt{snd}(M)}\!/_{x_1,x_2}\right\}
\end{array}$$

As an example of the introduced syntax, below we give the formal specification (that uses the equational theory E_0 from Fig. 2) for the roles in the BDH protocol presented in Fig. 3.

$$\begin{array}{l}
C_{\mathrm{rfc}}(s, c, ch) \triangleq \nu a.\overline{ch}\langle \phi(a, \phi(c, \mathrm{g}))\rangle. \\
\quad ch(y). \\
\quad \mathtt{let}\; k_c \coloneqq \mathrm{h}(\phi(a \cdot c, y))\;\mathtt{in} \\
\qquad \mathtt{let}\; \mathit{cert} \coloneqq \langle \phi(c, \mathrm{g}),\, \mathtt{sig}(\phi(c, \mathrm{g}), s)\rangle\;\mathtt{in} \\
\quad \overline{ch}\left\langle \{\langle\langle a,\, \phi(c, \mathrm{g})\rangle,\, \mathit{cert}\rangle\}_{k_c}\right\rangle
\end{array}$$

$$\begin{array}{l}
T_{\mathrm{rfc}}(pk_s, ch) \triangleq \nu t.ch(z_1). \\
\quad \overline{ch}\langle \phi(t, \mathrm{g})\rangle. \\
\quad ch(z_2). \\
\quad \mathtt{let}\; k_t \coloneqq \mathrm{h}(\phi(t, z_1))\;\mathtt{in} \\
\quad \mathtt{let}\; \langle m_1, m_2\rangle \coloneqq \\
\quad \langle \mathtt{fst}(\mathtt{dec}(z_2, k_t)), \mathtt{snd}(\mathtt{dec}(z_2, k_t))\rangle\;\mathtt{in} \\
\qquad \mathtt{if}\,\mathtt{snd}(m_1) = \mathtt{check}(\mathtt{snd}(m_2), pk_s)\,\mathtt{then} \\
\quad \mathtt{if}\,\phi(\mathtt{fst}(m_1), \mathtt{snd}(m_1)) = z_1\,\mathtt{then}\;\overline{ch}\langle \mathtt{auth}\rangle
\end{array}$$

The card role process is parametrised by the secret key s of the payment system, the secret key c of the card and the session channel ch. The terminal role is parametrised only by the system's public key pk_s and ch. The action $\overline{ch}\langle\mathtt{auth}\rangle$ is an event used to indicate at what point the terminal believes it has authenticated the card.

B. Semantics

We present the state of a process as an *extended process* $\nu\vec{x}.(\sigma \mid P)$. The syntax for extended processes is given in Fig. 5. An extended process comprises private values $\vec{x}$, playing the role of keys, nonces, fresh channels, etc., messages already sent on the network σ and the *future actions* P. For example, the extended process $\nu s.\left(\left\{{}^{\mathtt{pk}(s),M}\!/_{u_1,u_2}\right\} \mid a(z)\right)$ is composed of the fresh private secret key s, the sent messages $\mathtt{pk}(s)$ and M, and the input action $a(z)$, that is not executed yet. Notice that to list the messages sent we use the substitution $\sigma = \left\{{}^{M_1,\cdots,M_n}\!/_{u_1,\cdots,u_n}\right\}$, meaning that the message M_i is available through the "alias" variable u_i. When a substitution serves as a ledger of sent messages, we refer to it as a *frame*.

Extended processes:

$$\begin{array}{rcl}
A & ::= & \sigma \mid P \\
 & | & \nu x.A
\end{array}$$

Transition labels:

$$\begin{array}{rcl}
\pi & ::= & \tau \\
 & | & \overline{M}(z) \\
 & | & M\,N
\end{array}$$

Fig. 5: A syntax for extended processes and transition labels.

We require extended processes $\nu\vec{x}.(\sigma \mid P)$ to be in normal form, i.e. to satisfy the restriction that the variables in $\mathrm{dom}(\sigma)$ are fresh for $\vec{x}$, $\mathrm{fv}(P)$ and $\mathrm{fv}(y\sigma)$, for all variables y. That is, σ is idempotent, and substitutions are fully applied to P. We follow the convention that operational rules are defined directly on extended processes in normal form. This avoids numerous complications caused by the structural congruence in the original definition of bisimilarity for the applied π-calculus.

An extended process $\nu\vec{x}.(\sigma \mid P)$ may make a transition to a new state by executing an action available in P. We present transitions as labelled arrows. The syntax for labels is presented in Fig. 5. To describe the transition rules we define the *bound names* of the transition label such that $\mathrm{bn}(\pi) = \{x\}$ only if $\pi = \overline{M}(x)$ and $\mathrm{bn}(\pi) = \varnothing$ otherwise and the *names* such that $\mathrm{n}(M\,N) = \mathrm{fv}(M) \cup \mathrm{fv}(N)$ and $\mathrm{n}(M(x)) = \mathrm{fv}(M) \cup \{x\}$. Finally, we present the transition rules in Fig. 6. The label of a transition represents an action that the process takes to arrive at a new state. In our reduced version of the applied π-calculus those actions are either input or output: specifically $M\,N$ denotes the input of message N on channel M and $\overline{M}(z)$ denotes an output on channel M of a message bound to the variable z. Notice the importance of capture avoidance in the rule Inp. For instance $\nu n, k.\left(\left\{{}^{\{n\}_k}\!/_{u}\right\} \mid c(x).P\right)$ can execute an input action $c\,n$: since n is a bound name, it is renamed using α-conversion to e.g. z in the initial process to avoid a name clash occurring when the substitution is applied to P (bottom-right of the Inp rule). The resulting process is $\nu z, k\left(\left\{{}^{\{z\}_k}\!/_{u}\right\} \mid P\left\{{}^{z,n}\!/_{n,x}\right\}\right)$.

Fig. 6 is missing any rule for τ transitions, invisible for external observers. We purposefully left these rules out since the protocols are modeled in a way that no τ actions can occur. The absence of τ transitions, in turn, allows us to employ *strong* semantics in the definition of quasi-open bisimilarity which simplifies the bisimilarity check since it ensures certain finiteness, i.e., each process has finitely many π-labelled transitions for any label π. Strong bisimilarity, in contrast to *weak* bisimilarity, cares about the number of silent τ transitions which is always zero in this paper.

C. Equivalence notion

In Section V we define the unlinkability of the payment system as an equivalence notion: if the system behaves like the ideal unlinkable system, then it is unlinkable. In this subsection we formally define the exact equivalence relation for extended processes that we use in the paper.

$$\frac{M\sigma =_E K}{\sigma \mid K(y).P \xrightarrow{M\,N} \sigma \mid P\left\{{}^{N\sigma}/_{y}\right\}}\,\textsf{Inp} \qquad \frac{u \mathrel{\#} M, N, P, \sigma \qquad M\sigma =_E K}{\sigma \mid \overline{K}\langle N\rangle.P \xrightarrow{\overline{M}(u)} \left\{{}^{N}/_{u}\right\} \circ \sigma \mid P}\,\textsf{Out} \qquad \frac{\sigma \mid P \xrightarrow{\pi} A \qquad M =_E N}{\sigma \mid \mathtt{if}\, M = N \,\mathtt{then}\, P \xrightarrow{\pi} A}\,\textsf{Mat}$$

$$\frac{A \xrightarrow{\pi} B \qquad x \mathrel{\#} \mathrm{n}(\pi)}{\nu x.A \xrightarrow{\pi} \nu x.B}\,\textsf{Res} \qquad \frac{\sigma \mid P \xrightarrow{\pi} B \qquad x \mathrel{\#} \mathrm{n}(\pi)\,, \sigma}{\sigma \mid \nu x.P \xrightarrow{\pi} \nu x.B}\,\textsf{Extrusion} \qquad \frac{\sigma \mid P \xrightarrow{\pi} \nu\vec{x}.(\rho \mid Q) \qquad \vec{x} \cup \mathrm{bn}(\pi) \mathrel{\#} P}{\sigma \mid !P \xrightarrow{\pi} \nu\vec{x}.(\rho \mid Q \mid !P)}\,\textsf{Rep-act}$$

$$\frac{\sigma \mid P \xrightarrow{\pi} \nu\vec{x}.(\rho \mid R) \qquad \vec{x} \cup \mathrm{bn}(\pi) \mathrel{\#} Q}{\sigma \mid P \mid Q \xrightarrow{\pi} \nu\vec{x}.(\rho \mid R \mid Q)}\,\textsf{Par-l} \qquad \frac{\sigma \mid P \xrightarrow{\pi} \nu\vec{x}.(\rho \mid R) \qquad \vec{x} \cup \mathrm{bn}(\pi) \mathrel{\#} Q}{\sigma \mid Q \mid P \xrightarrow{\pi} \nu\vec{x}.(\rho \mid Q \mid R)}\,\textsf{Par-r}$$

Fig. 6: A labelled transition system defined on extended processes in normal form.

An equivalence captures both static and dynamic parts of processes' behaviour: no distinction is made for processes if they output the same sequence of messages so far and if they can match each other's actions. That is, we require such relation to be *bisimilarity*. The exact type of bisimilarity is important though, and there are many notions of bisimilarity [31]. We consider a bisimilarity that is also a *congruence*. Recall that we study the unlinkability of the card in a hostile environment, hence we wish the unlinkability property to hold in *any context*, e.g. in the absence or presence of any terminals.

We start with a standard definition of static equivalence, that captures the distinction between two snapshots of the protocol execution and then will make our way to a bisimilarity congruence.

Definition 2. (static equivalence) *Two extended processes $\nu\vec{x}.(\sigma \mid P)$ and $\nu\vec{y}.(\theta \mid Q)$ are statically equivalent whenever for all messages M and N such that $\vec{x}, \vec{y} \mathrel{\#} M, N$, we have $M\sigma =_E N\sigma$ if and only if $M\theta =_E N\theta$.*

In the context of the definition above, we say that the message M is a *recipe* for some message K under σ if $M \mathrel{\#} \vec{x}$ and $M\sigma =_E K$.

To ensure that our notion of bisimilarity is a congruence relation we require our bisimulation to be an open relation. A relation is *open* if it is preserved under substitutions fresh for the domain of the frame of the extended process, as stated formally below. By introducing this condition we give our attacker the capacity to influence messages bound to free variables (therefore accounting for all possible ways to "stage" the attack) without access to the outputs recorded in the frame – they may only be used as a part of the input.

Definition 3. (open relation) *A relation $\mathcal{R}$ over extended processes is open whenever, if $A = \nu\vec{x}.(\sigma \mid P)$ and $B = \nu\vec{y}.(\theta \mid Q)$ and $A \mathrel{\mathcal{R}} B$, then for all ρ such that $\mathrm{dom}(\sigma)$ is*[2] *fresh for ρ, we have $A\rho \mathrel{\mathcal{R}} B\rho$.*

The precise technical name for the notion of bisimilarity restricted to open relations is *quasi-open bisimilarity* — the coarsest of bisimilarities for the applied π-calculus that is a congruence [11]. We stress the importance of coarseness here: verifying against too fine equivalence may lead to spurious attacks.

Definition 4. (quasi-open bisimilarity) *An open symmetric relation between extended processes $\mathcal{R}$ is a quasi-open bisimulation whenever, if $A \mathrel{\mathcal{R}} B$ then the following hold:*

- *A and B are statically equivalent.*
- *If $A \xrightarrow{\pi} A'$ there exists B' such that $B \xrightarrow{\pi} B'$ and $A' \mathrel{\mathcal{R}} B'$.*

Processes P and Q are quasi-open bisimilar, written $P \sim Q$, whenever $P \mathrel{\mathcal{R}} Q$ for some quasi-open bisimulation $\mathcal{R}$.

In the next section we define our unlinkability property by using quasi-open bisimilarity. The bisimilarity-based approach takes into account the fact that an attacker can make decisions during the execution of a protocol. Moreover, in comparison to familiar trace equivalence, for checking which tools like DeepSec [32] may help, bisimilarity is a *safer* option since trace equivalence is coarser. Spelled out, this means that if a privacy property is defined using bisimilarity and it holds, then it holds when the bisimilarity is replaced by trace equivalence in the definition of the property. The opposite is not true as illustrated in related work using the BAC protocol for ePassport [10], [25], which is unlinkable if trace equivalence is instead employed, while, with respect to bisimilarity, there is a distinguishing strategy that can be exploited to link two sessions involving the same ePassport. Moreover, the openness/congruence feature of our chosen notion of bisimilarity allows us to prove properties in a modular way: an equivalence-based property that a smaller subsystem satisfies extends to a larger system for free.

D. Describing attacks as modal logic formulas

To conclude the background section we describe a succinct way of expressing attacks on bisimilarity. We will use a minimal fragment of a modal logic [33], [34], sufficient for the purpose of the paper. The syntax for formulae is very concise.

$$\begin{array}{rlll} \psi ::= & M = N & & \text{equality} \\ \mid & \langle\pi\rangle\psi & & \text{diamond} \end{array}$$

The semantics of our minimal modal logic is as follows.

$$\begin{array}{lcl} \nu\vec{x}.(\sigma \mid P) \vDash M = N & \Leftrightarrow & M\sigma =_E N\sigma \text{ and } \vec{x} \mathrel{\#} M, N \\ A \vDash \langle\pi\rangle\psi & \Leftrightarrow & \exists B, \text{ s.t. } A \xrightarrow{\pi} B \text{ and } B \vDash \psi \end{array}$$

[2] $\mathrm{dom}(\sigma) = \mathrm{dom}(\theta)$ is an invariant property of a bisimulation, meaning that any pairs of processes not satisfying this property can be safely removed from a bisimulation.

If there is a formula ψ that is satisfied by one process, but is not satisfied by the other, e.g. $A \vDash \psi$, but $B \nvDash \psi$, then we know that $A \nsim B$ holds. The converse does not hold unless we take a larger modal logic, but this fragment suffices for the current protocol. The formula ψ captures the strategy of an attacker for distinguishing two processes. Such a distinguishing strategy is a trace of transitions that the process A can make, but the process B may fail to match, followed by a test $M = N$ demonstrating the violation of static equivalence. We will use this modal logic approach to formally present our previously mentioned attack on the BDH protocol in the proof of Theorem 1 in the next section.

V. Unlinkability

In this section we introduce a formal definition of unlinkability as a process equivalence, and show that the BDH protocol in Fig. 3 does not satisfy this definition.

A. *Verification of unlinkability is challenging*

There is a certain feature in the original definition of (strong) unlinkability proposed by Arapinis et al. [23], namely the use of *weak* transitions in the underlying bisimilarity notion. In the weak semantics for a given process A and the transition label π there could be infinitely many states B s.t. $A \xrightarrow{\pi} B$ which can make verification a difficult task.

To overcome this obstacle we follow a method developed by Horne and Mauw [10] allowing the reduction of weak to strong bisimilarity that supports image finiteness. The insight of their work is that a certain way of expressing a protocol in the applied π-calculus makes verification easier without compromising unlinkability in the original sense. We adopt this method not only out of safety consideration: bisimilarity is stronger than trace equivalence and we are not losing anything when verifying in a stronger setting, but also to open up a discussion on possible automation of checking the bisimilarity of two processes. Studying the detailed proof of bisimilarity we provide in Section VI-B may outline the steps and challenges to overcome when considering tool support.

Another, parallel, approach to unlinkability verification is rolling back to a familiar trace equivalence, an equivalence with established tool support, by proving that a particular class of protocols indeed allows doing that. For instance, this approach was taken recently by Baelde, Delaune, and Moreau to analyse stateful protocols [35]. This work also demonstrates limitations of our method: analysing BDH, we *can* drop one party, the terminal, from consideration because cards and terminals share no secret, while otherwise, an observed reaction of an honest participant may break unlinkability.

B. *Definition of unlinkability*

Perhaps the most straightforward way to design unlinkable payments is to introduce cards that immediately expire after one use. Such cards can never participate in a purchase more than once and payments are undoubtedly unlinkable. We say that if the “real world” system where cards are used multiple times, is indistinguishable by an attacker from an idealised unlinkable world in which cards are disposed of after each use, then unlinkability of payments is achieved.

Let $C(s, c, ch)$ be the card process scheme parametrised by the payment system’s secret key s, communication channel ch and the card’s secret key c. Then we have the following.

Definition 5. (unlinkability) *A card process scheme C is unlinkable whenever*

$$\nu s.\overline{out}\langle pk(s)\rangle.!\nu c.\nu ch.\overline{card}\langle ch\rangle.C(s, c, ch)$$
$$\sim$$
$$\nu s.\overline{out}\langle pk(s)\rangle.!\nu c.!\nu ch.\overline{card}\langle ch\rangle.C(s, c, ch)$$

The process on the left of the above relation models the idealised world where a card participates in no more than one transaction. This process starts by creating the secret key of the payment system s. Then the public key $\mathrm{pk}(s)$ of the payment system is made available via the output on the public channel out. Each newly manufactured card c is allowed to participate in the execution of the payment protocol just once. The process on the right of the above relation models the more realistic situation where each card c may participate in several runs of the protocol. If the idealised situation is equivalent to the real world one, where the equivalence we employ is quasi-open bisimilarity (Def. 4), we say that the payment system satisfies unlinkability. Since our chosen notion of equivalence is a congruence, we can check unlinkability for a subsystem comprising cards only and be sure that the presence of any terminals would not make the whole system linkable. This can be illustrated by the context

$$\nu out.\Big(\{\cdot\}\,|\,out(pk_s).\overline{out'}\langle pk_s\rangle!\nu ch_t.\overline{term}\langle ch_t\rangle.T(pk_s,ch_t)\Big)$$

where out' is a public channel used in a full system to announce the system’s public key. The detailed proof that this context indeed leads to a correct representation of a full system with cards and terminals is given in the companion paper [11] dedicated entirely to the notion of quasi-open bisimilarity and containing proofs that Def. 4 is correct in the sense that it is a sound and complete congruence with respect to an established testing semantics. The contributions of the current paper and the above mentioned companion paper are disjoint, since the paper you are now reading focusses on using quasi-open bisimilarity to verify a protocol.

In summary, the process scheme on the left of the equation in Def. 5 is the specification and on the right is the implementation. If we can prove that the equivalence problem holds for a particular protocol then that protocol complies with the specification. This is similar to the pattern for specifying unlinkability introduced in related work [23], with the key difference being that only cards need be accounted for, since the only information shared with the terminal is the public key of the payment system.

In both cases in the definition above the card uses a newly created session channel ch that is output on the public channel $card$, hence an attacker has the power to observe and control the creation of radio frequency communication channels.

Moreover, a dedicated channel per session is a requirement in the transport protocol ISO/IEC 14443 [36] that is used in contactless EMV cards. Conveniently, session channels are the reason behind the lack of silent τ-transitions in the protocols we study in the paper, which allows us to employ the strong notion of bisimilarity and simplifies proofs.

C. BDH is not unlinkable

Given the formal definition of unlinkability in Def. 5 we can now establish that the Blinded Diffie-Hellman protocol from Fig. 3 is not unlinkable.

Theorem 1. *$C_{\text{rfc}}(s, c, ch)$ violates unlinkability.*

Proof. To describe the attack on unlinkability of the BDH protocol we follow the modal logic formula notation described in Section IV-D. Consider the following processes, where C_{rfc} is as defined in Section IV-A.

$$\mathit{RFC}_{\text{spec}} \triangleq \nu s.\overline{out}\langle \mathtt{pk}(s)\rangle.!\nu c.\nu ch.\overline{card}\langle ch\rangle.C_{\text{rfc}}$$

$$\mathit{RFC}_{\text{impl}} \triangleq \nu s.\overline{out}\langle \mathtt{pk}(s)\rangle.!\nu c.!\nu ch.\overline{card}\langle ch\rangle.C_{\text{rfc}}$$

To show that $\mathit{RFC}_{\text{spec}} \not\sim \mathit{RFC}_{\text{impl}}$ we present a formula that is satisfied by $\mathit{RFC}_{\text{impl}}$, but not by $\mathit{RFC}_{\text{spec}}$. Let the formula ψ be as follows.

$$\begin{aligned}
&\langle \overline{out}(pk_s)\rangle\\
&\langle \overline{card}(u_1)\rangle\langle \overline{u_1}(v_1)\rangle\langle u_1\, \phi(y_1, \mathtt{g})\,\rangle\langle \overline{u_1}(w_1)\rangle\\
&\langle \overline{card}(u_2)\rangle\langle \overline{u_2}(v_2)\rangle\langle u_2\, \phi(y_2, \mathtt{g})\,\rangle\langle \overline{u_2}(w_2)\rangle\\
&(\,\mathtt{snd}(\mathtt{dec}(w_1, \mathtt{h}(\phi(y_1, v_1)))) =\\
&\qquad\qquad \mathtt{snd}(\mathtt{dec}(w_2, \mathtt{h}(\phi(y_2, v_2))))\,)
\end{aligned}$$

The above formula describes two sessions of the BDH protocol, which, for $\mathit{RFC}_{\text{impl}}$, can be with the same card, say c_1. The equality test at the end of ψ compares the certificates obtained from each session to each other, which the terminal can decrypt in both sessions. This certificate can be the same for both sessions of $\mathit{RFC}_{\text{impl}}$ involving the same card, since it is bound to the card's identity c_1. Therefore $\mathit{RFC}_{\text{impl}} \vDash \psi$. In contrast, $\mathit{RFC}_{\text{spec}} \nvDash \psi$ since every session is with a new card and hence the equality test never holds, since the certificates will always differ. □

In the next section, we present our enhanced BDH protocol that satisfies unlinkability.

VI. Making Blinded Diffie-Hellman truly unlinkable

In this section, we propose our improvement to the BDH protocol proposed by EMVCo called *Unlinkable BDH* (UBDH). This improvement makes use of a certification scheme with certificates invariant under blinding. We point to an existing instance of such a certification scheme, the Verheul certification scheme, and, finally, we prove that our improvement indeed makes the Blinded Diffie-Hellman protocol unlinkable [2].

A. Blinded Diffie-Hellman with blinded certificates

Recall from Section III-B and Theorem 1 that the reason behind the failure of unlinkability of the BDH protocol proposed by EMVCo is that the card gives away its static certificate and its blinding factor. While this allows an honest terminal to authenticate the card, the public key of the card ultimately obtained by the terminal may be used to track the card in the future. We demonstrate in this section that authentication can still be performed without disclosing the public key or the signature. In order to achieve this, we specify more precisely the signature scheme (initially unspecified by EMVCo) used for certificate verification. In particular, we require that blinding and signing operations must commute. In this case, the signature can be blinded with the same nonce as the card's public key at the beginning of the session and later checked against the public key of the payment system directly in its blinded form. As a result, only the blinded version of the card's public key is ever revealed.

The equational theory E for the improved protocol is the equational theory E_0 in Fig. 2 extended with the property expressed in Fig. 7, which permits scalar multiplication and signing to commute.

$$\phi(M, \mathtt{sig}(N, K)) =_E \mathtt{sig}(\phi(M, N), K)$$

Fig. 7: Equation for blinding extending the equational theory in Fig. 2.

It now follows from the blinding condition above and the last equation in Fig. 2 that the check of the signature, blinded with some blinding factor, returns the message, blinded with the same factor. This property of signatures in the equational theory E is used by the terminal when authenticating the card in our proposed update of the BDH protocol. The updated BDH protocol is presented informally in Fig. 8 and the corresponding formal π-calculus specification of the two roles involved is presented below.

$$\begin{aligned}
&C_{\text{upd}}(s, c, ch) \triangleq \nu a.\overline{ch}\langle \phi(a, \phi(c, \mathtt{g}))\rangle.\\
&\quad ch(y).\\
&\quad \mathtt{let}\ k_c \coloneqq \mathtt{h}(\phi(a \cdot c, y))\ \mathtt{in}\\
&\quad \mathtt{let}\ m \coloneqq \langle \phi(a, \phi(c, \mathtt{g})), \phi(a, \mathtt{sig}(\phi(c, \mathtt{g}), s))\rangle\ \mathtt{in}\\
&\quad \overline{ch}\langle \{m\}_{k_c}\rangle
\end{aligned}$$

$$\begin{aligned}
&T_{\text{upd}}(pk_s, ch) \triangleq \nu t.ch(z_1).\\
&\quad \overline{ch}\langle \phi(t, \mathtt{g})\rangle.\\
&\quad ch(z_2).\\
&\quad \mathtt{let}\ k_t \coloneqq \mathtt{h}(\phi(t, z_1))\ \mathtt{in}\\
&\quad \mathtt{let}\ \langle m_1, m_2\rangle \coloneqq\\
&\quad \langle \mathtt{fst}(\mathtt{dec}(z_2, k_t)), \mathtt{snd}(\mathtt{dec}(z_2, k_t))\rangle\ \mathtt{in}\\
&\quad \mathtt{if}\ m_1 = \mathtt{check}(m_2, pk_s)\ \mathtt{then}\\
&\quad \mathtt{if}\ m_1 = z_1\ \mathtt{then}\ \overline{ch}\langle \mathtt{auth}\rangle
\end{aligned}$$

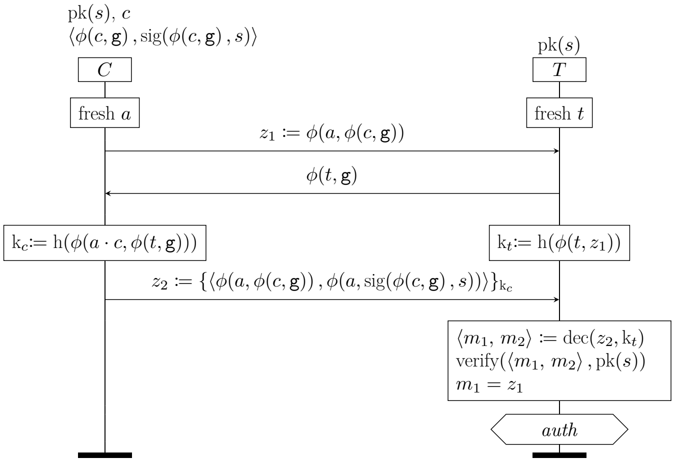
pk(s), c
$\langle \phi(c, \mathbf{g}), \mathrm{sig}(\phi(c, \mathbf{g}), s) \rangle$
pk(s)
C
T
fresh a
fresh t
$z_1 := \phi(a, \phi(c, \mathbf{g}))$
$\phi(t, \mathbf{g})$
$\mathrm{k}_c := \mathrm{h}(\phi(a \cdot c, \phi(t, \mathbf{g})))$
$\mathrm{k}_t := \mathrm{h}(\phi(t, z_1))$
$z_2 := \{\langle \phi(a, \phi(c, \mathbf{g})), \phi(a, \mathrm{sig}(\phi(c, \mathbf{g}), s)) \rangle\}_{\mathrm{k}_c}$
$\langle m_1, m_2 \rangle := \mathrm{dec}(z_2, \mathrm{k}_t)$
verify($\langle m_1, m_2 \rangle$, pk(s))
$m_1 = z_1$
auth

Fig. 8: The Unlinkable BDH protocol.

Our version differs from the original proposal in message z_2 sent by the card to the terminal, i.e. now only the (encrypted) blinded certificate is transferred. At no point in the protocol, can the terminal unblind the card's public key since the blinding factor a is never revealed to any terminal.

We conclude this subsection by mentioning that there is a signature scheme satisfying both the blinding condition in Fig. 7, and the technical requirements of the BDH protocol [2], namely the Verheul certification scheme [21]. The scheme has been implemented on smart cards [22] by Batina et al., using BN[3] curves [38] with time presenting one blinded certificate of 0.45 seconds, which is within the limit of 500ms of the card present in the reader field [39]. In the proposal [2] EMVCo intends to use p256 curve, however switching over to a pairing-friendly BN curve would not introduce any slowdowns, compared to p256 curve, in on-card computation as was shown by Dzurenda et al. in the performance analysis [40] of different elliptic curves on smart cards.

B. Self-blindable certificates bring unlinkability in BDH

In this section we present a detailed proof of unlinkability of the Unlinkable BDH protocol that will illustrate the importance of the chosen equivalence relation (quasi-open bisimilarity, Def. 4). We define *UPD*$_{\text{spec}}$ and *UPD*$_{\text{impl}}$ as

$$\mathit{UPD}_{\text{spec}} \triangleq \nu s.\overline{out}\langle \mathtt{pk}(s) \rangle.!\nu c.\nu ch.\overline{card}\langle ch \rangle.C_{\text{upd}}(s, c, ch)$$

$$\mathit{UPD}_{\text{impl}} \triangleq \nu s.\overline{out}\langle \mathtt{pk}(s) \rangle.!\nu c.!\nu ch.\overline{card}\langle ch \rangle.C_{\text{upd}}(s, c, ch)$$

The UBDH protocol is unlinkable as established by the following theorem.

Theorem 2. *$C_{\text{upd}}(s, c, ch)$ satisfies unlinkability.*

Proof. By Def. 5 of unlinkability, we must show that *UPD*$_{\text{spec}}$ ~ *UPD*$_{\text{impl}}$. Therefore we shall provide a quasi-open bisimulation relation $\mathfrak{R}$ such that *UPD*$_{\text{spec}}$ $\mathfrak{R}$ *UPD*$_{\text{impl}}$.

To define such $\mathfrak{R}$ we have to introduce some notation. Let L, $D \in \mathbb{N}$ be the number of sessions and the number of cards in the system, respectively. We use indices $l \in \{1, \cdots, L\}$ and $d \in \{1, \cdots, D\}$ to track sessions and cards.

Define $m^d(a, y)$ as the encrypted blinded certificate:

$$m^d(a,y) := \{\langle \phi(a, \phi(c_d, \mathbf{g})), \phi(a, \mathtt{sig}(\phi(c_d, \mathbf{g}), s)) \rangle\}_{\mathtt{h}(\phi(a \cdot c_d, y))}$$

Define a partition $\Psi := \{\alpha,\ \beta,\ \gamma\ \delta\}$ of the set of all sessions $\{1, \cdots, L\}$, where α is the set of sessions in which the channel is created, but no message has been sent; β is the set of sessions in which the blinded public key has been sent but the response has not been received; γ is the set of all sessions in which the response has been received but the encrypted blinded certificate has not been sent; δ is the set of all sessions in which the encrypted blinded certificate has been sent.

Define a partition $\Omega := \{\zeta^1, \cdots, \zeta^D\}$ of the set of all sessions $\{1, \cdots, L\}$, where ζ^d is the set of all sessions with the card d.

Let $\vec{Y} := (Y_1, \cdots, Y_L)$ be the list of inputs, where Y_l is the input in session l. Recall that Y_l can refer to messages already output on the network (the last line in Fig. 9). Let $K := |\beta \cup \gamma \cup \delta|$ be the number of *started* sessions. Since we consider processes up to α-conversion and permutation of names (aka. equivariance), we assume that a_l is the blinding factor in session l.

Finally, we define the following process subterms, which correspond to the elements of the partition Ψ.

$$\mathcal{E}^d(ch) \triangleq \nu a.\overline{ch}\langle \phi(a, \phi(c_d, \mathbf{g})) \rangle.\mathcal{F}^d(ch, a)$$
$$\mathcal{F}^d(ch, a) \triangleq ch(y).\mathcal{G}^d(ch, a, y)$$
$$\mathcal{G}^d(ch, a, y) \triangleq \overline{ch}\left\langle m^d(a, y) \right\rangle$$
$$\mathcal{H}^d \triangleq 0$$

The bisimulation relation $\mathfrak{R}$ is defined as *the least symmetric open relation* satisfying the constraints[4] in Fig. 9. Spelled out, we pair the reachable states of *UPD*$_{\text{spec}}$ and *UPD*$_{\text{impl}}$ based on the number of sessions and the respective stages of the card in a session. Notice that *UPD*$_{\text{spec}}$ $\mathfrak{R}$ *UPD*$_{\text{impl}}$ by the definition of $\mathfrak{R}$.

To prove that $\mathfrak{R}$ is indeed a quasi-open bisimulation, according to Def. 4, we must demonstrate

1) *(bisimulation)* Whenever $A\ \mathfrak{R}\ B$, and $A \xrightarrow{\pi} A'$, there exists B' such that $B \xrightarrow{\pi} B'$ and $A'\ \mathfrak{R}\ B'$.
2) *(openness)* $\mathfrak{R}$ is closed under the application of a substitution fresh for the domain of the frame of any of the related states.
3) *(static equivalence)* Whenever $A\ \mathfrak{R}\ B$, A is statically equivalent to B.

Since $\mathfrak{R}$ is by definition a symmetric relation, we provide proof only for the cases when the left-side process starts first. Below we present the exhaustive list of cases for the defining conditions of the relation $\mathfrak{R}$ in Fig. 9. Proof trees justifying each transition can be found in Appendix C. Openness and static equivalence are discussed separately.

[3] The original paper [21], in contrast, describes the system using symmetric pairings on a supersingular curve. This approach historically precedes the asymmetric one, making certain Decisional Diffie-Hellman problem simple and requires greater field size (which would slow down on-card computation) to achieve the same level of security as a non-supersingular curve based system [37].

[4] The relation $\mathfrak{R}$ may not be the *smallest* quasi-open bisimilarity satisfying *UPD*$_{\text{spec}}$ $\mathfrak{R}$ *UPD*$_{\text{impl}}$.

$$\mathit{UPD}_{\text{spec}}\ \mathfrak{R}\ \mathit{UPD}_{\text{impl}}$$

$$\begin{array}{l}
\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y}) \triangleq \nu s, c_1, \cdots, c_L, ch_1, \cdots, ch_L,\\
a_{l_1}, \cdots, a_{l_K}.(\sigma\\
\quad \mid C_1 \mid \cdots \mid C_L\\
\quad \mid !\nu c.\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c, ch))
\end{array}$$

$$\mathfrak{R}$$

$$\begin{array}{l}
\mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \triangleq \nu s, c_1, \cdots, c_D, ch_1, \cdots, ch_L,\\
a_{l_1}, \cdots, a_{l_K}.(\theta\\
\quad \mid \cdots \mid C^d_l \mid \cdots \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_d, ch)\\
\quad \mid !\nu c.!\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}((s, ch, c)))
\end{array}$$

$$C_l = \begin{cases} \mathcal{E}^l(ch_l) & \text{if } l \in \alpha\\ \mathcal{F}^l(ch_l, a_l) & \text{if } l \in \beta\\ \mathcal{G}^l(ch_l, a_l, Y_l\sigma) & \text{if } l \in \gamma\\ \mathcal{H}^l & \text{if } l \in \delta \end{cases}$$

$$C^d_l = \begin{cases} \mathcal{E}^d(ch_l) & \text{if } l \in \zeta^d \cap \alpha\\ \mathcal{F}^d(ch_l, a_l) & \text{if } l \in \zeta^d \cap \beta\\ \mathcal{G}^d(ch_l, a_l, Y_l\theta) & \text{if } l \in \zeta^d \cap \gamma\\ \mathcal{H}^d & \text{if } l \in \zeta^d \cap \delta \end{cases}$$

$$\begin{array}{ll}
pk_s\sigma = \mathtt{pk}(s) & \\
u_l\sigma = ch_l & \text{if } l \in \{1, \cdots, L\}\\
v_l\sigma = \phi(a_l, \phi(c_l, \mathrm{g})) & \text{if } l \in \beta \cup \gamma \cup \delta\\
w_l\sigma = m^l(a_l, Y_l\sigma) & \text{if } l \in \delta
\end{array}$$

$$\begin{array}{ll}
pk_s\theta = \mathtt{pk}(s) & \\
u_l\theta = ch_l & \text{if } l \in \{1, \cdots, L\}\\
v_l\theta = \phi(a_l, \phi(c_d, \mathrm{g})) & \text{if } l \in \zeta^d \cap (\beta \cup \gamma \cup \delta)\\
w_l\theta = m^d(a_l, Y_l\theta) & \text{if } l \in \zeta^d \cap \delta
\end{array}$$

$$\Psi \coloneqq \{\alpha, \beta, \gamma, \delta\},\ \ \Omega \coloneqq \{\zeta^1, \cdots, \zeta^D\} \text{ are partitions of } \{1, \cdots, L\}$$

$$K \coloneqq |\beta \cup \gamma \cup \delta| \qquad l_1, \cdots, l_K \in \beta \cup \gamma \cup \delta$$

$$pk_s, u_l, v_l, w_l \ \#\ \{card, s\} \cup \{c_l, ch_l, a_l | l \in \{1, \cdots, L\}\}$$

$$Y_l \ \#\ \{s\} \cup \{c_l, ch_l, a_l | l \in \{1, \cdots, L\}\}$$

$$\mathrm{fv}(Y_l) \cap (\{v_i | i \in \alpha\} \cup \{w_i | i \in \alpha \cup \beta \cup \gamma \cup \{l\}\}) = \varnothing$$

Fig. 9: Defining conditions for the bisimulation relation $\mathfrak{R}$.

Case 1. $\mathit{UPD}_{\text{spec}}\ \mathfrak{R}\ \mathit{UPD}_{\text{impl}}$, $\overline{out}(pk_s)$. The process $\mathit{UPD}_{\text{spec}}$ can do the transition $\overline{out}(pk_s)$ to the state $\mathit{UPD}^{\varnothing}_{\text{spec}}(\varnothing)$. There is a state $\mathit{UPD}^{\varnothing,\varnothing}_{\text{impl}}(\varnothing)$ to which the process $\mathit{UPD}_{\text{impl}}$ can do the transition $\overline{out}(pk_s)$. By the definition of $\mathfrak{R}$ we have $\mathit{UPD}^{\varnothing}_{\text{spec}}(\varnothing)\ \mathfrak{R}\ \mathit{UPD}^{\varnothing,\varnothing}_{\text{impl}}(\varnothing)$.

Case 2. $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})\ \mathfrak{R}\ \mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$, $\overline{card}(u_{L+1})$. The process $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})$ can do the transition $\overline{card}(u_{L+1})$ to the state $\mathit{CH}_{\text{spec}} \triangleq \mathit{UPD}^{\{\alpha\cup\{L+1\},\beta,\gamma,\delta\}}_{\text{spec}}((Y_1, \cdots, Y_L, \varnothing))$. In the process $\mathit{UPD}^{\Psi,\Omega}_{\text{impl}}$ either some card d starts a new session and the resulting state is $\mathit{CH}_{\text{impl}} \triangleq \mathit{UPD}^{\{\alpha\cup\{L+1\},\beta,\gamma,\delta\},\{\cdots,\zeta^d\cup\{L+1\},\cdots\}}_{\text{impl}}((Y_1, \cdots, Y_L, \varnothing))$ or the new card is created and the resulting state is $\mathit{CHC}_{\text{impl}} \triangleq \mathit{UPD}^{\{\alpha\cup\{L+1\},\beta,\gamma,\delta\},\Omega\cup\{\{L+1\}\}}_{\text{impl}}((Y_1, \cdots, Y_L, \varnothing))$. In both cases by the definition of $\mathfrak{R}$ we have $\mathit{CH}_{\text{spec}}\ \mathfrak{R}\ \mathit{CH}_{\text{impl}}$ and $\mathit{CH}_{\text{spec}}\ \mathfrak{R}\ \mathit{CHC}_{\text{impl}}$.

Case 3. $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})\ \mathfrak{R}\ \mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$, $\overline{u_l}(v_l)$, and $l \in \alpha$. The process $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})$ can do the transition $\overline{u_l}(v_l)$ to the state $\mathit{APK}_{\text{spec}} \triangleq \mathit{UPD}^{\{\alpha\setminus\{l\},\beta\cup\{l\},\gamma,\delta\}}_{\text{spec}}(\vec{Y})$. There is a state $\mathit{APK}_{\text{impl}} \triangleq \mathit{UPD}^{\alpha\setminus\{l\},\beta\cup\{l\},\gamma,\delta\},\Omega}_{\text{impl}}(\vec{Y})$ to which the process $\mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$ can do the transition $\overline{u_l}(v_l)$. By the definition of $\mathfrak{R}$ we have $\mathit{APK}_{\text{spec}}\ \mathfrak{R}\ \mathit{APK}_{\text{impl}}$.

Case 4. $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})\ \mathfrak{R}\ \mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$, $u_l\, Y_l$, and $l \in \beta$. Let $\chi_l(\vec{Y}, M)$ be the list of message terms obtained from $\vec{Y}$ by the replacement of lth entry in $\vec{Y}$ with M. The process $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})$ can do the transition $u_l\, Y_l$ to the state $\mathit{IN}_{\text{spec}} \triangleq \mathit{UPD}^{\{\alpha,\beta\setminus\{l\},\gamma\cup\{l\},\delta\}}_{\text{spec}}(\chi_l(\vec{Y}, Y_l))$. There is a state $\mathit{IN}_{\text{impl}} \triangleq \mathit{UPD}^{\{\alpha,\beta\setminus\{l\},\gamma\cup\{l\},\delta\},\Omega}_{\text{impl}}(\chi_l(\vec{Y}, Y_l))$ to which the process $\mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$ can do the transition $u_l\, Y_l$. By the definition of $\mathfrak{R}$ we have $\mathit{IN}_{\text{spec}}\ \mathfrak{R}\ \mathit{IN}_{\text{impl}}$.

Case 5. $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})\ \mathfrak{R}\ \mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$, $\overline{u_l}(w_l)$, and $l \in \beta$. The process $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})$ can do a transition $\overline{u_l}(w_l)$ to the state $\mathit{CRT}_{\text{spec}} \triangleq \mathit{UPD}^{\{\alpha,\beta,\gamma\setminus\{l\},\delta\cup\{l\}\}}_{\text{spec}}(\vec{Y})$. There is a state $\mathit{CRT}_{\text{impl}} \triangleq \mathit{UPD}^{\{\alpha,\beta,\gamma\setminus\{l\},\delta\cup\{l\}\},\Omega}_{\text{impl}}(\vec{Y})$ to which the process $\mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$ can do a transition $\overline{ch_l}(w_l)$. By the definition of $\mathfrak{R}$ we have $\mathit{CRT}_{\text{spec}}\ \mathfrak{R}\ \mathit{CRT}_{\text{impl}}$.

Openness. $\mathfrak{R}$, by definition, is open: whenever $A\ \mathfrak{R}\ B$, then $A\rho\ \mathfrak{R}\ B\rho$ for any ρ fresh for the domain of the frame of A. No such substitution ρ introduce transitions not considered above. Indeed, since $\mathrm{fv}(\mathit{UPD}_{\text{spec}}) = \mathrm{fv}(\mathit{UPD}_{\text{impl}}) = \{out, card\}$, the substitution ρ may only affect out, $card$ and free variables in the input Y_l. Therefore it is straightforward to modify proof trees [see the repository [41] for details]: the transition label $\overline{out}(pk_s)$ is replaced by $\overline{out\rho}(pk_s)$, the transition label $\overline{card}(u_{L+1})$ with $\overline{card\rho}(u_{L+1})$ and $\vec{Y}$ with $\vec{Y}\rho \coloneqq (Y_1\rho, \cdots, Y_L\rho)$, where $\varnothing\rho \coloneqq \varnothing$. By α-conversion we may assume that the range of ρ does not contain variables “reserved” for future outputs or private nonces: $\mathrm{fv}(y\rho) \cap (\{s, pk_s\} \cup \{c_i, ch_i, a_i, u_i, v_i, w_i | l \in \mathbb{N}\}) = \varnothing$ for any y. Then, freshness conditions remain untouched up to the renaming of variables directly affected by ρ.

Static equivalence. To conclude, we prove that A is statically equivalent to B whenever $A\ \mathfrak{R}\ B$. There is nothing to prove in the case of $\mathit{UPD}_{\text{spec}}\ \mathfrak{R}\ \mathit{UPD}_{\text{impl}}$ since frames are empty. The proof for the case $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})\ \mathfrak{R}\ \mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$ is presented separately in Lemma 3. □

To prove that $\mathit{UPD}^{\Psi}_{\text{spec}}(\vec{Y})$ is statically equivalent to $\mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$ we use a weak notion of the *normal form* $M\!\downharpoonleft$ of a message term M, that captures the least complex, up to multiplication, expression of M; and the notion of the *normalisation* of a frame σ with respect to the equational theory E, which is a saturation of the range of σ with weak normal forms of messages that have a recipe under σ. Recipes can be conveniently recorded in the domain of normalisation. Definitions are standard and given in Appendix A.

Definition 6. *(m-atomic, ϕ-atomic) A message term M is m-atomic if there are no such M_1, M_2, s.t. $M =_E M_1 \cdot M_2$; it is ϕ-atomic if there are no such M_1, M_2, s.t. $M =_E \phi(M_1, M_2)$*

A subterm N of M is an *immediate m-factor* if it is m-atomic and there is a message term K, s.t. $N \cdot K = M$.

Definition 7. *(non-trivial recipe) The recipe M is* non-trivial *under σ if* $\mathrm{fv}(M{\downarrow}) \cap \mathrm{dom}(\sigma) \neq \varnothing$.

We conclude the proof of Theorem 2 with the following.

Lemma 3. *$UPD^{\Psi}_{spec}(\vec{Y})$ is statically equivalent to $UPD^{\Psi,\Omega}_{impl}(\vec{Y})$.*

Proof. Considering the definition of $\mathfrak{R}$ in Fig. 9, let $\nu\vec{x}.(\sigma \mid P) \triangleq UPD^{\Psi}_{\text{spec}}(\vec{Y})$ and $\nu\vec{y}.(\theta \mid Q) \triangleq UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y})$. We aim to show that $\nu\vec{x}.(\sigma \mid P)$ is statically equivalent to $\nu\vec{y}.(\theta \mid Q)$. Since $\vec{x}$ is always a superset of $\vec{y}$, we prove that for all messages M and N, s.t. $\vec{x} \# M, N$, we have $M\sigma =_E N\sigma$ if and only if $M\theta =_E N\theta$.

Recall the definition of $m^d(a, y)$:

$$m^d(a,y) := \{\langle\phi(a,\phi(c_d,\mathrm{g})),\phi(a,\mathtt{sig}(\phi(c_d,\mathrm{g}),s))\rangle\}_{\mathrm{h}(\phi(a\cdot c_d,y))}$$

Since it is sufficient to consider normalisations when proving static equivalence, we present the normalisations for a fixed partitions $\{\alpha, \beta, \gamma, \delta\}$, $\{\zeta^1, \cdots, \zeta^D\}$ of the set of all sessions $\{1, \cdots, L\}$ of σ and θ with respect to E below.

$$\begin{array}{l}
pk_s\sigma = \mathtt{pk}(s)\\
u_l\sigma = ch_l \text{ if } l \in \{1, \cdots, L\}\\
v_l\sigma = \phi(a_l \cdot c_l, \mathrm{g}) \text{ if } l \in \beta \cup \gamma \cup \delta\\
\text{if } l \in \delta \text{ and } Y_l = \phi(T_l, \mathrm{g})\\
\mathtt{fst}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l))))\,\sigma = \phi(a_l \cdot c_l, \mathrm{g})\\
\mathtt{snd}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l))))\,\sigma = \phi(a_l \cdot c_l, \mathtt{sig}(\mathrm{g}, s))\\
\mathtt{check}(\mathtt{snd}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l)))), pk_s)\,\sigma = \phi(a_l \cdot c_l, \mathrm{g})\\
\text{if } l \in \delta \text{ and } Y_l \neq \phi(T_l, \mathrm{g})\\
w_l\sigma = m^l(a_l, Y_l\sigma)
\end{array}$$

$$\begin{array}{l}
pk_s\theta = \mathtt{pk}(s)\\
u_l\theta = ch_l \text{ if } l \in \{1, \cdots, L\}\\
v_l\theta = \phi(a_l \cdot c_d, \mathrm{g}) \text{ if } l \in \zeta^d \cap (\beta \cup \gamma \cap \delta)\\
\text{if } l \in \zeta^d \cap \delta \text{ and } Y_l = \phi(T_l, \mathrm{g})\\
\mathtt{fst}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l))))\,\theta = \phi(a_l \cdot c_d, \mathrm{g})\\
\mathtt{snd}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l))))\,\theta = \phi(a_l \cdot c_d, \mathtt{sig}(\mathrm{g}, s))\\
\mathtt{check}(\mathtt{snd}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l)))), pk_s)\,\theta = \phi(a_l \cdot c_d, \mathrm{g})\\
\text{if } l \in \zeta^d \cap \delta \text{ and } Y_l \neq \phi(T_l, \mathrm{g})\\
w_l\theta = m^d(a_l, Y_l\theta)
\end{array}$$

We prove static equivalence by induction on the structure of the weak normal form of $N\sigma$ exploring all cases allowed by the grammar in Fig. 1. We present proofs starting from the equation under the frame σ. The argument for the converse case is the same. From now on M, M_k, N, N_k are always fresh for $\vec{x}$.

Case 1. $N\sigma =_E \mathrm{g}$.

Case 1.1. $N = \mathrm{g}$. If M is a recipe for g, then $M = \mathrm{g}$, since there is no non-trivial recipe for g under the normalisation of σ. Then we have $\mathrm{g}\sigma =_E \mathrm{g}\sigma$ if and only if $\mathrm{g}\theta =_E \mathrm{g}\theta$ as required.

Case 1.2. $N \neq \mathrm{g}$. There is nothing to prove in this case, since there is no non-trivial recipe for g under the normalisation of σ.

Case 2. $N\sigma =_E z$, z is a variable.

Case 2.1. $N = z$. If M is a recipe for z, then $M = z$, since there is no non-trivial recipe for z under the normalisation of σ. Then we have $z\sigma =_E z\sigma$ if and only if $z\theta =_E z\theta$ as required.

Case 2.2. $N\sigma =_E ch_l$. Since N is fresh for $\vec{x}$, $N = u_l$. There is unique recipe $M = u_l$ for ch_l and we have $u_l\sigma =_E u_l\sigma$ if and only if $u_l\theta =_E u_l\theta$ as required.

Case 3. $N\sigma = K_1 \cdot K_2$.

Any message term in the normalisation of σ is m-atomic, hence no message is an immediate m-factor of another message in the normalisation of σ. Therefore there is only one case to consider.

Case 3.1. $N = N_1^{j_1} \cdot \cdots \cdot N_k^{j_k}$, that is $N\sigma$ is generated by m-factors which have a recipe under the normalisation of σ: $N\sigma = N_1^{j_1}\sigma \cdot \cdots \cdot N_k^{j_k}\sigma$. By the induction hypothesis suppose that for all recipes M_i for an m-factor $N_i\sigma$ of $N\sigma$, we have $M_i\sigma =_E N_i\sigma$ if and only if $M_i\theta =_E N_i\theta$, $i \in \{1, \cdots, k\}$. By applying multiplication, we have $M_1^{j_1}\theta \cdot \cdots \cdot M_k^{j_k}\theta = (M_1^{j_1} \cdot \cdots \cdot M_k^{j_k})\theta =_E (N_1^{j_1} \cdot \cdots \cdot N_k^{j_k})\theta = N_1^{j_1}\theta \cdot \cdots \cdot N_k^{j_k}\theta$ as required, and $N_i\theta$ is an m-factor of $N\theta$.

Case 4. $N\sigma = \phi(K_1, K_2)$.

Let us define

$$\begin{array}{l}
V_1 := v_l,\ V_2 := \mathtt{fst}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l)))),\\
V_3 := \mathtt{snd}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l))))\\
V_4 := \mathtt{check}(\mathtt{snd}(\mathtt{dec}(w_l, \mathtt{h}(\phi(T_l, v_l)))), pk_s)
\end{array}$$

Case 4.1. $N\sigma = \phi(a_l \cdot c_l, \mathrm{g})$ and $Y_l = \phi(T_l, \mathrm{g})$. Since N is fresh for $\vec{x}$, $N \in \{V_1, V_2, V_4\}$. Let M be a recipe for $\phi(a_l \cdot c_l, \mathrm{g})$, then $M \in \{V_1, V_2, V_4\}$ and we have $M\sigma =_E N\sigma$ if and only if $M\theta =_E N\theta$ for any N and M as required. In case $Y_l \neq \phi(T_l, \mathrm{g})$, $N = V_1$, there is unique recipe $M_1 = V_1$ and the argument is the same.

Case 4.2. $N\sigma = \phi(a_l \cdot c_l, \mathtt{sig}(\mathrm{g}, s))$ and $Y_l = \phi(T_l, \mathrm{g})$. Since N is fresh for $\vec{x}$, $N = V_3$ and there is unique recipe $M = V_3$ for $\phi(a_l \cdot c_l, \mathrm{g})$, and we have $V_3\sigma =_E V_3\sigma$ if and only if $V_3\theta =_E V_3\theta$ as required. If $Y_l \neq \phi(T_l, \mathrm{g})$, there is no recipe for $\phi(a_l \cdot c_l, \mathtt{sig}(\mathrm{g}, s))$ and there is nothing to prove.

Case 4.3. $N = \phi(N_1, N_2)$, $N_2 \in \{V_1, V_2, V_3, V_4\}$ and $Y_l = \phi(T_l, \mathrm{g})$. By the induction hypothesis suppose that for all recipes M_1 for $N_1\sigma$, we have $M_1\sigma =_E N_1\sigma$ if and only if $M_1\theta =_E N_1\theta$, then multiply N_2 by a scalar M_1 and obtain $\phi(M_1\theta, N_2\theta) = \phi(M_1, N_2)\,\theta =_E \phi(N_1, N_2)\,\theta = \phi(N_1\theta, N_2\theta)$ for any N_2 as required. In case $Y_l \neq \phi(T_l, \mathrm{g})$, $N_2 = V_1$ and the argument is the same.

Case 4.4. $N = \mathtt{sig}(\cdots\mathtt{sig}(N_1,N_2)\cdots,N_k)$, $N_1 \in \{V_1,V_2,V_3,V_4\}$ and $Y_l = \phi(T_l,\mathrm{g})$. By the induction hypothesis suppose that for all recipes M_i for $N_i\sigma$ we have $M_i\sigma =_E N_i\sigma$ if and only if $M_i\theta =_E N_i\theta$, $i \in \{2,\cdots,k\}$. By applying the signature operation to N_1, we have

$$\begin{aligned}&\mathtt{sig}(\cdots\mathtt{sig}(N_1,M_2)\cdots,M_k)\theta =\\ &\mathtt{sig}(\cdots\mathtt{sig}(N_1\theta,M_2\theta)\cdots,M_k\theta) =_E\\ &\mathtt{sig}(\cdots\mathtt{sig}(N_1\theta,N_2\theta)\cdots,N_k\theta) =\\ &\mathtt{sig}(\cdots\mathtt{sig}(N_1,N_2)\cdots,N_k)\theta\end{aligned}$$

as required. In case $Y_l \neq \phi(T_l,\mathrm{g})$, $N_1 = V_1$ and the argument is the same.

Case 4.5. $N = \phi(N_1,N_2)$. Similar to case 3.1 with $j_i = 1, i \in \{1,2\}$.

Case 5. $N\sigma = \langle K_1, K_2\rangle$.

Since no pair is contained in the normalisation of σ, there is only one case to consider.

Case 5.1. $N = \langle N_1, N_2\rangle$. Similar to case 4.5.

Case 6. $N\sigma = \mathtt{h}(K_1)$. Similar to case 5 with $j_i = 1, i = 1$.

Case 7. $N\sigma = \mathtt{pk}(K_1)$.

Case 7.1. $N\sigma = \mathtt{pk}(s)$. Then $N = pk_s$, since N is fresh for $\vec{x}$. There is a unique recipe $M = pk_s$ for $\mathtt{pk}(s)$ and we have $pk_s\sigma =_E pk_s\sigma$ if and only if $pk_s\theta =_E pk_s\theta$ as required.

Case 7.2. $N = \mathtt{pk}(N_1)$. Similar to case 6.

Case 8. $N\sigma = \mathtt{sig}(K_1,K_2)$. Similar to case 5.

Case 9. $N\sigma = \{K_1\}_{K_2}$.

If $Y_l = \phi(T_l,\mathrm{g})$ no encrypted message term is contained in the normalised frame and there is only one case to consider.

Case 9.1. $N = \{N_1\}_{N_2}$. Similar to case 5.

If $Y_l \neq \phi(T_l,\mathrm{g})$, there is also the following.

Case 9.2. $N\sigma =_E m^l(a_l, Y_l\sigma)$. Since N is fresh for $\vec{x}$, $N = w_l$. There is unique recipe $M = w_l$ for $m^l(a_l,Y_l\sigma)$ and we have $w_l\sigma = w_l\sigma$ if and only if $w_l\theta = w_l\theta$ as required.

□

VII. Unlinkable Authentication for BDH

The twofold aim of the BDH protocol is to guarantee unlinkability of the card, while allowing the terminal to authenticate the card. In this paper we emphasise unlinkability, since this is the more novel of the two requirements. Indeed, ProVerif, and other tools, can be used to automatically confirm our target authentication property – injective agreement [42] – holds for both BDH protocols in this paper.

The process scheme below specifies the behaviour of honest terminals and honest cards. The attacker is the implicit environment that interacts with these honest participants.

$$\begin{aligned}\mathit{SYS} \triangleq\ \ \nu s.\Big(&!\nu c.!\nu ch_c.\overline{\mathit{card}}\langle ch_c\rangle.C(s,c,ch_c)\ \ \mid\\ &\overline{\mathit{out}}\langle\mathtt{pk}(s)\rangle.!\nu ch_t.\overline{\mathit{term}}\langle ch_t\rangle.T(\mathtt{pk}(s)\,,ch_t)\ \Big)\end{aligned}$$

In the above, the processes C and T can be instantiated with C_{rfc} and T_{rfc} or with C_{upd} and T_{upd} to obtain $\mathit{SYS}_{\mathrm{rfc}}$ and $\mathit{SYS}_{\mathrm{upd}}$, respectively. Notice a fresh channel for each run is advertised on channels *card* or *term*. These allow the messages associated with a run to be uniquely identified in the formulation of injective agreement below.

The following injective agreement property is standard [42], [43]. *Agreement* here means that when a terminal thinks it has authenticated a card, an honest card really executed the protocol while exchanging the same messages as the terminal. *Injectivity* strengthens agreement by ensuring that every successfully authenticating run of a terminal corresponds to a separate run of a card.

Definition 8. *(injective agreement) Process SYS satisfies injective agreement iff for every trace π_0, π_1, …, π_n such that $\mathit{SYS} \vDash \langle\pi_0\rangle\ldots\langle\pi_n\rangle\mathtt{true}$*[5] *there exists an injective function $f{:}\,\mathbb{N} \rightarrow \mathbb{N}$ such that, for every a such that $0 < a \leq n$, $\pi_a = \overline{T_a}(w)$ and $\mathit{SYS} \vDash \langle\pi_0\rangle\ldots\langle\pi_n\rangle(w = \mathtt{auth})$, we have the following:*

- *for some $0 \leq i < j < k < a$, we have the following $\pi_i = T_i\,M_i$, $\pi_j = \overline{T_j}(u_j)$, and $\pi_k = T_k\,M_k$;*
- *for $0 \leq f(a) < i' < j' < k' < a$, s.t. $\pi_{f(a)} = \overline{C_{f(a)}}(ch_c)$, we have $\pi_{i'} = \overline{C_i}(u_i)$, $\pi_{j'} = C_j\,M_j$, and $\pi_{k'} = \overline{C_k}(u_k)$,*
- *and $\mathit{SYS} \vDash \langle\pi_0\rangle\ldots\langle\pi_n\rangle(C_{f(a)} = \mathit{card} \wedge \phi_i \wedge \phi_j \wedge \phi_k)$, where $\phi_\ell \triangleq u_\ell = M_\ell \wedge ch_c = C_\ell \wedge T_a = T_\ell$.*

We can now verify that our target functional property holds. The proof is conducted in ProVerif [see Appendix B for the code], with respect to an extension of the standard Diffie-Hellman theory for ProVerif, which approximates the equations for multiplication with the equation $\phi(a,\phi(b,\mathrm{g})) = \phi(b,\phi(a,\mathrm{g}))$. We had to extend that standard theory further with an equation $\phi(a,\phi(b,\phi(c,\mathrm{g}))) = \phi(b,\phi(a,\phi(c,\mathrm{g})))$, so that blinding factors are treated correctly.

Theorem 4. *$\mathit{SYS}_{\mathrm{rfc}}$ and $\mathit{SYS}_{\mathrm{upd}}$ satisfy injective agreement.*

Despite ProVerif requiring an approximation of the Diffie-Hellman theory, we find this proof to be sufficient, since authentication already held for the BDH protocol of EMVCo, and we simply aim to show that our proposed fix does not inadvertently break authentication. Notice in particular that our attacker is incapable of relating two public keys, i.e. if she knows $pk_1 \coloneqq \phi(c_1,\mathrm{g})$ and $pk_2 \coloneqq \phi(c_2,\mathrm{g})$, it is infeasible for her to find a scalar h s.t. $h \cdot pk_1 = pk_2$. This contrasts, to our thorough proof of unlinkability (Theorem 2), which takes equations in Fig. 2 and Fig. 7 fully into account.

VIII. Unlinkability challenges in EMV 1st Gen

In this section we explain that even in the presence of the UBDH key agreement it is challenging to make an entire EMV 1st Gen transaction unlinkable without substantial changes to the current protocol [1] and the back-end, i.e., bank-terminal communications. Notice that EMVCo never has made precise how BDH and the rest of the current EMV protocol coexist. In this section we assume the scenario in which the key agreement protocol is run before any data about the transaction is transmitted and we informally describe a generic EMV

[5] We reuse our modal logic to describe traces satisfied by a process, extended such that $A \vDash \mathtt{true}$ holds, and also $A \vDash \phi \wedge \psi$ iff $A \vDash \phi$ and $A \vDash \psi$.

transaction highlighting steps where identifying information about the card is revealed to the terminal. In this discussion we account for the following sources of identities the card has: unique identifiers such as the card number; and coarser identifiers such as the data formats that the card supports. We warn the reader that this section is not a comprehensive EMV protocol description. The current EMV standard admits optional steps that are up to a particular payment system to implement. For improved clarity, these optional steps are mostly omitted.

A. *EMV transaction flow*

An EMV transaction is a sequence of the terminal's commands with an optional data payload and the card's responses. This message exchange is broken down into several stages presented in Fig. 10. A successful transaction ends with the generation of a cryptogram that the terminal eventually sends to the bank in exchange for money. In what follows, we briefly discuss each stage and summarise which of the unlinkability issues are relevant for the eavesdropping-resistant and the active attacker-resistant models.

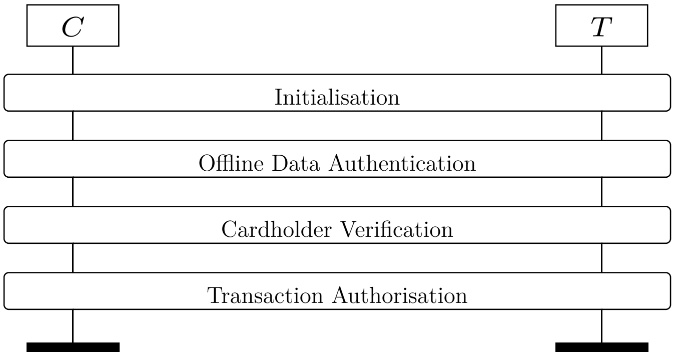

Fig. 10: The EMV 1st Gen protocol stages.

1) Initialisation: Firstly, the terminal asks the card which applications it supports. The card responds with a list of payment application identifiers, e.g. Visa Debit, Maestro, etc. Then the terminal selects a particular application from the presented list. If the intention of EMVCo is that BHD is run immediately after this step to protect further communication, a passive eavesdropper can distinguish two cards selecting different applications thereby violating a coarse form of external unlinkability. This could be averted by all cards having the same identifier, and, in addition, if the key agreement were further enhanced by group signatures. A group signature allows all cards to be signed with different keys, and verified with a single key, without different card issuers revealing their keys to each other. This is an example of something that could be reviewed by the developers of the EMV 2nd Gen standard.

Having selected the application, the card sends the PDOL list specifying which transaction details (e.g. the amount, the date, the currency, etc.) the terminal should send to the card. Next, in one message, the terminal sends the requested PDOL data and requests the AIP list, specifying the functions supported by the card including authentication methods and the AFL list, specifying memory addresses where the card stores its data such as the card number, the certificate, etc. Finally, the terminal uses the addresses from the AFL list to read the actual data from the card. The following data is mandatory for the card to have as specified in the EMV 1st Gen standard.

- Application Expiry Date
- The card's number.
- Card Risk Management Data Object Lists (CDOLs)

CDOLs specify the information the card wants to receive from the terminal to generate cryptogram(s) at the end of the transaction, e.g. the country code, the terminal nonce, etc. Typically, the certificate and the payment system public key index is also available for the terminal to read.

We can see that the application selection step already violates unlinkability. As mentioned above, the list of supported payment systems serves a coarse form of identity, but, fortunately, the PDOL and the transaction data it specifies would be protected from a passive eavesdropper by running BDH. However, these messages would not be protected from an active attacker, and hence would be available to distinguish cards. The same applies to the AIP, AFL, CDOL lists, and the public key index. All this information varies from card to card and contributes to the card's fingerprint that can be used by an attacker to link sessions with the same card with high probability. Notice that it is relatively easy to remove certain messages so they do not contribute to the fingerprint of the card. For instance, all cards may request a standard set of transaction details from the terminal making the PDOL obsolete. The others, such as the list of supported payment applications, would require a substantial reworking of the current protocol. Such reworking would be even more challenging since it should account for the strong forms of the card's identity such as the expiry date, the card's number, and the certificate. The most difficult point to address in order to achieve strong unlinkability, is how to hide the card number from the terminal, since, in EMV 1st Gen, it is revealed in order to route the cryptogram through the network to the bank at the last stage of the transaction.

2) Offline data authentication: Offline Data Authentication is an optional step in the EMV 1st Gen protocol at which point the terminal authenticates the data previously received from the card during the Initialisation step. The ODA can be completed in several ways.[6]

- Dynamic Data Authentication (DDA). At the Initialisation phase, the card could provide the DDOL with the list of data elements that the terminal must send to the card if DDA is selected. This DDOL is a coarse identity that contributes to the card's fingerprint. In case the DDOL is not provided, the terminal should always send a nonce to the card. The card replies with a signature on the DDOL data provided by the terminal and its dynamic data (e.g.

[6] For backward compatibility, EMV retains also a Static Data Authentication (SDA) mode, but, this should be avoided in new cards, since it facilitates card cloning by replaying the signature [44].

a nonce generated by the card). This signature itself does not reveal identifying information about the card, but to verify it the terminal uses the card's public key obtained from the certificate, both of which are card identifiers.
- Combined Data Authentication (CDA). This case does not require additional messages and is a part of the Transaction Authorisation phase. Otherwise, it is similar to DDA, but the card's dynamic data also includes transaction details, e.g. the cryptogram.

A selection of a particular Offline Data Authentication method itself does not contribute to the card's fingerprint since the supported methods have already been disclosed at the Initialisation step by providing the terminal with the AIP, however, we can see that all three Offline Data Authentication methods involve a unique card identifier that, if exposed to a terminal, break unlinkability with respect to an active attacker. For a passive eavesdropper, the data communicated in this phase is unavailable since she is already locked out of the session. In the presence of either BDH or UBDH, the ODA phase is built in to the key agreement, hence could be redundant in a EMV 2nd Gen protocol.

3) Cardholder Verification: Cardholder Verification is also optional in the current EMV protocol. Cards supporting Cardholder Verification provide a list of supported methods in the Initialisation step. This list is a coarse form of the card's identity, contributes to its fingerprint, and is available only to active attackers. Verification methods include a handwritten signature, PIN, and a verification via the consumer's device (e.g. through biometric data entered via a mobile phone) which is out of the scope of the EMV standard. For any threat model we consider in this paper we have to assume that a signature or PIN may only be disclosed to an honest party. For example, a PIN is never entered into a dishonest terminal, because otherwise, if a malicious terminal learns the PIN, the basic security requirement of EMV, the safety of money in the cardholder's account, is compromised; not only the privacy.

4) Transaction Authorisation: Transaction Authorisation is the ultimate and mandatory phase at which the terminal asks the card to generate the Application Cryptogram (AC). A cryptogram is typically a Hash-based Message Authentication Code (HMAC) generated by the card over the data coming both from the card and the terminal (specified by CDOLs the terminal has received in the Initialisation phase) using the key derived from the shared secret mk between the card and the bank and the ATC (Application Transaction Counter). The official EMVCo recommendations on the minimum set of data elements to be included in the cryptogram are listed in Book 2 of the EMV 1st Gen standard [1] and include, for instance, the type of the cryptogram (decline, approve, request online), the card number, ATC, etc. Notice that together with the cryptogram, the data that it was generated over must be provided to verify this cryptogram, which causes linkability issues in the presence of active attackers, i.e. the value of the ATC and the card number are forms of the card identity.

B. The future of Unlinkable EMV transactions

The above discussion demonstrates that even the promotion of the anti-eavesdropping requirement to full unlinkability in the presence of *passive* attackers, relevant in the range from 1m to 20m, demands updates to the EMV 1st Gen, e.g. at the application selection step. This potential minor update would be impossible to roll out incrementally since it would require a great cooperation effort between EMVCo and adopters. For a major update, that strengthens the requirements further to support unlinkability with respect to *active* attackers, relevant within 1m from the card, our analysis clearly suggests that it is unfeasible to hide all identifying information from the terminal without touching critical elements of the standard. Making the messages that contribute to the fingerprint of the card constant and, hence, obsolete would reduce the flexibility of EMV. The direct card's identifiers play a crucial role in steps that ensure the primary EMV goal, a safe money exchange: e.g. the card number is important for network routing, the card's public key is involved in data authentication, etc.

We conclude that the anti-eavesdropping requirement, more specifically, anti-eavesdropping on the phase where the transaction data is communicated, is the only thing BDH fulfills. UBDH, on the other hand, targets a much more ambitious goal, and the respective updates would inevitably require larger compromises and infrastructure updates. Whether such an update to the full transaction protocol exists is future work that should be addressed in coordination with EMVCo.

IX. Conclusion

In this paper, we have investigated the Blinded Diffie-Hellman key agreement protocol in Fig. 3 proposed by EMVCo to introduce encryption into a proposal for 2nd Gen EMV payments. Although BDH indeed introduces a way to establish a symmetric key between the card and the terminal, and meets the initial EMVCo requirements, we have shown that the privacy of the cardholder will not be protected against an active attacker. In particular, in Theorem 1 we have shown that the presence of an active adversary leads to a straightforward failure of BDH to be unlinkable. In our proposal for improving the protocol in Fig. 8, we use a generic signature scheme that respects blinding. To support this, in Section VI-A, we point out that at least one existing signature scheme meets our requirements, namely Verheul signatures. To verify our proposal, we introduced a strong definition of unlinkability in Def. 5 and applied this definition to the applied π-calculus model of UBDH in Theorem 2, thereby proving that UBDH indeed makes the key agreement unlinkable.

The first, core, take away message of our paper is related to the threat model in the EMVCo proposal of secure channel establishment [2]. The anti-eavesdropping requirement in the BDH and 2nd Gen specifications [2], [3] form a reasonable privacy enhancement guided by the current state of the EMV standard and the infrastructure already deployed. We, however, have taken the liberty to look beyond passive eavesdropping and considered the implications of realistic active attackers, as captured in Def. 5. We support this investigation by observing that, the anti-tracking requirement explicitly mentioned in the BDH proposal remains open to interpretation, specifically [2], "The protocol is designed to protect against eavesdropping and card tracking." We conclude that unlinkable EMV key agreement, under such assumptions, is feasible as we prove formally in Theorem 1. Yet, unlinkability in the presence of an active attacker is difficult to extend to the full EMV 1st Gen transaction as we discuss informally in Section VIII.

The second, more general, take away message concerns our method for verifying properties defined in terms of a process equivalence problem. Our evolved unlinkability definition, Def. 5, based on the notion of quasi-open bisimilarity, which is a congruence, enables compositional reasoning about protocols, such as UBDH, without a shared key. This state-of-the-art approach to bisimilarity checking facilitates proving that the property holds for unboundedly many sessions, where the main challenge is to define the relation in Fig. 9, after which we apply the method to show that the relation is a quasi-open bisimulation. Furthermore, the equational theory we employ is not yet covered by equivalence checking tools, so this example proof may help inform the extension of tools to this class of problems.

EMVCo is still in the process of revising the protocol for the EMV 2nd Gen standard [45]. As awareness of these privacy issues is growing and methods, such as ours, for verification of privacy properties emerging, we expect stakeholders to take seriously the possibility of making payments unlinkable.

Acknowledgements: We thank the anonymous reviewers for their constructive suggestions.

References

[1] "EMV Integrated Circuit Card Specifications for Payment Systems. Books 1-4," EMVCo LLC, Tech. Rep., 2011, Accessed: 26-08-2021. [Online]. Available: https://www.emvco.com/document-search/

[2] "EMV ECC key establishment protocols," EMVCo LLC, RFC until 28th January 2013, 2012, Accessed: 01-04-2020. [Online]. Available: http://www.emvco.com/specifications.aspx?id=243

[3] "EMV next generation. next generation kernel system architecture overview," EMVCo LLC, Technical report, 2014.

[4] C. Brzuska, N. P. Smart, B. Warinschi, and G. J. Watson, "An analysis of the EMV channel establishment protocol," in *Proceedings of the 2013 ACM SIGSAC CCS*, ser. CCS '13. New York, NY, USA: Association for Computing Machinery, 2013. doi: 10.1145/2508859.2516748. ISBN 9781450324779 p. 373–386.

[5] Y. Guo, Z. Zhang, J. Zhang, and X. Hu, "Security analysis of EMV channel establishment protocol in an enhanced security model," in *International Conference on Information and Communications Security*, ser. LNCS, vol. 8958. Springer, 2014. doi: 10.1007/978-3-319-21966-0_22 pp. 305–320.

[6] F. Pfeiffer, K. Finkenzeller, and E. Biebl, "Theoretical limits of ISO/IEC 14443 type A RFID eavesdropping attacks," in *Smart SysTech 2012; European Conference on Smart Objects, Systems and Technologies*, 2012, pp. 1–9.

[7] D. R. Novotny, J. R. Guerrieri, M. Francis, and K. Remley, "HF RFID electromagnetic emissions and performance," in *2008 IEEE International Symposium on Electromagnetic Compatibility*, 2008. doi: 10.1109/ISEMC.2008.4652133 pp. 1–7.

[8] M. Engelhardt, F. Pfeiffer, K. Finkenzeller, and E. Biebl, "Extending ISO/IEC 14443 type a eavesdropping range using higher harmonics," in *Proceedings of 2013 European Conference on Smart Objects, Systems and Technologies (SmartSysTech)*, 2013. ISBN 978-3-8007-3521-1 pp. 1–8.

[9] R. Habraken, P. Dolron, E. Poll, and J. de Ruiter, "An RFID skimming gate using higher harmonics," in *Radio Frequency Identification*, S. Mangard and P. Schaumont, Eds. Cham: Springer International Publishing, 2015. doi: 10.1007/978-3-319-24837-0_8. ISBN 978-3-319-24837-0 pp. 122–137.

[10] R. Horne and S. Mauw, "Discovering ePassport Vulnerabilities using Bisimilarity," *Logical Methods in Computer Science*, vol. Volume 17, Issue 2, 2021. doi: 10.23638/LMCS-17(2:24)2021. [Online]. Available: https://lmcs.episciences.org/7537

[11] R. Horne, S. Mauw, and S. Yurkov, "Compositional analysis of protocol equivalence in the applied π-calculus using quasi-open bisimilarity," in *Theoretical Aspects of Computing – ICTAC 2021*, A. Cerone and P. C. Ölveczky, Eds. Springer International Publishing, 2021. ISBN 978-3-030-85315-0 pp. 235–255.

[12] D. Basin, R. Sasse, and J. Toro-Pozo, "The EMV standard: Break, fix, verify," in *2021 IEEE Symposium on Security and Privacy (S&P)*, 2021.

[13] ——, "Card brand mixup attack: Bypassing the PIN in non-Visa cards by using them for Visa transactions," in *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 2021. [Online]. Available: https://www.usenix.org/conference/usenixsecurity21/presentation/basin

[14] S. Mauw, Z. Smith, J. Toro-Pozo, and R. Trujillo-Rasua, "Distance-bounding protocols: Verification without time and location," in *2018 IEEE Symposium on Security and Privacy (S&P)*, 2018. doi: 10.1109/SP.2018.00001 pp. 549–566.

[15] T. Chothia, J. de Ruiter, and B. Smyth, "Modelling and analysis of a hierarchy of distance bounding attacks," in *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD: USENIX Association, Aug. 2018. ISBN 978-1-939133-04-5 pp. 1563–1580. [Online]. Available: https://www.usenix.org/conference/usenixsecurity18/presentation/chothia

[16] I. Boureanu, T. Chothia, A. Debant, and S. Delaune, "Security analysis and implementation of relay-resistant contactless payments," in *Proceedings of the 2020 ACM SIGSAC CCS*, ser. CCS '20. New York, NY, USA: Association for Computing Machinery, 2020. doi: 10.1145/3372297.3417235. ISBN 9781450370899 p. 879–898.

[17] A.-I. R. Radu, T. Chothia, C. J. Newton, I. Boureanu, and L. Chen, "Practical EMV relay protection," in *2022 IEEE Symposium on Security and Privacy (S&P)*, 2022, to appear.

[18] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *EUROCRYPT*, ser. LNCS, vol. 2045. Springer, 2001. doi: 10.1007/3-540-44987-6_7 pp. 93–118.

[19] S. A. Brands, *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. Cambridge, MA, USA: MIT Press, 2000. ISBN 0262024918

[20] H. Tews and B. Jacobs, "Performance issues of selective disclosure and blinded issuing protocols on Java Card," in *IFIP International Workshop on Information Security Theory and Practices*, ser. LNCS, vol. 5746. Springer, 2009. doi: 10.1007/978-3-642-03944-7_8 pp. 95–111.

[21] E. R. Verheul, "Self-blindable credential certificates from the Weil pairing," in *International Conference on the Theory and Application of Cryptology and Information Security*, ser. LNCS, vol. 2248. Springer, 2001. doi: 10.1007/3-540-45682-1_31 pp. 533–551.

[22] L. Batina, J.-H. Hoepman, B. Jacobs, W. Mostowski, and P. Vullers, "Developing efficient blinded attribute certificates on smart cards via pairings," in *International Conference on Smart Card Research and Advanced Applications*, ser. LNCS, vol. 6035. Springer, 2010. doi: 10.1007/978-3-642-12510-2_15 pp. 209–222.

[23] M. Arapinis, T. Chothia, E. Ritter, and M. Ryan, "Analysing unlinkability and anonymity using the applied pi calculus," in *2010 IEEE 23rd Computer Security Foundations Symposium (CSF)*, 2010. doi: 10.1109/CSF.2010.15 pp. 107–121.

[24] L. Hirschi, D. Baelde, and S. Delaune, "A method for verifying privacy-type properties: the unbounded case," in *2016 IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2016. doi: 10.1109/SP.2016.40 pp. 564–581.

[25] I. Filimonov, R. Horne, S. Mauw, and Z. Smith, "Breaking unlinkability of the ICAO 9303 standard for e-passports using bisimilarity," in *ESORICS*, ser. LNCS, vol. 11735. Springer, 2019. doi: 10.1007/978-3-030-29959-0_28 pp. 577–594.

[26] B. Schmidt, "Formal analysis of key exchange protocols and physical protocols," Ph.D. dissertation, ETH Zurich, 2012.

[27] S. Meier, B. Schmidt, C. Cremers, and D. Basin, "The Tamarin prover for the symbolic analysis of security protocols," in *International Conference on Computer Aided Verification*, ser. LNCS, vol. 8044. Springer, 2013. doi: 10.1007/978-3-642-39799-8_48 pp. 696–701.

[28] R. Küsters and T. Truderung, "Using ProVerif to analyze protocols with Diffie-Hellman exponentiation," in *2009 IEEE 22nd Computer Security Foundations Symposium (CSF)*, 2009. doi: 10.1109/CSF.2009.17 pp. 157–171.

[29] C. Cremers and D. Jackson, "Prime, order please! revisiting small subgroup and invalid curve attacks on protocols using diffie-hellman," in *2019 IEEE 32nd Computer Security Foundations Symposium (CSF)*, 2019. doi: 10.1109/CSF.2019.00013 pp. 78–7815.

[30] M. Abadi and C. Fournet, "Mobile values, new names, and secure communication," *SIGPLAN Not.*, vol. 36, no. 3, p. 104–115, Jan. 2001. doi: 10.1145/373243.360213

[31] D. Sangiorgi and D. Walker, *π-Calculus: A Theory of Mobile Processes*. Cambridge University Press, 2001. ISBN 0521781779

[32] V. Cheval, S. Kremer, and I. Rakotonirina, "DEEPSEC: Deciding equivalence properties in security protocols theory and practice," in *2018 IEEE Symposium on Security and Privacy (S&P)*, 2018. doi: 10.1109/SP.2018.00033. ISSN 2375-1207 pp. 529–546.

[33] R. Horne, K. Y. Ahn, S.-w. Lin, and A. Tiu, "Quasi-open bisimilarity with mismatch is intuitionistic," in *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, ser. LICS '18. New York, NY, USA: Association for Computing Machinery, 2018. doi: 10.1145/3209108.3209125. ISBN 9781450355834 p. 26–35. [Online]. Available: https://doi.org/10.1145/3209108.3209125

[34] R. Horne, "A bisimilarity congruence for the applied π-calculus sufficiently coarse to verify privacy properties," *CoRR*, vol. abs/1811.02536, 2018. [Online]. Available: http://arxiv.org/abs/1811.02536

[35] D. Baelde, S. Delaune, and S. MOREAU, "A Method for Proving Unlinkability of Stateful Protocols," in *2020 IEEE 33rd Computer Security Foundations Symposium (CSF)*, Boston, United States, Jun. 2020. doi: 10.3233/JCS-171070. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02459984

[36] "Cards and security devices for personal identification — contactless proximity objects — part 3: Initialization and anticollision," ISO/IEC, Tech. Rep. 14443-3, 2018. [Online]. Available: https://www.iso.org/standard/73598.html

[37] D. Freeman, M. Scott, and E. Teske, "A taxonomy of pairing-friendly elliptic curves," in *Journal of Cryptology*, Berlin, Heidelberg, 2010. doi: 10.1007/s00145-009-9048-z pp. 224–280.

[38] P. S. L. M. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," in *Selected Areas in Cryptography*, B. Preneel and S. Tavares, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. doi: 10.1007/11693383_22. ISBN 978-3-540-33109-4 pp. 319–331.

[39] "EMV Contactless Specifications for Payment Systems. Book A," EMVCo LLC, Tech. Rep., 2021, Accessed: 20-09-2021. [Online]. Available: https://www.emvco.com/document-search/

[40] P. Dzurenda, S. Ricci, J. Hajny, and L. Malina, "Performance analysis and comparison of different elliptic curves on smart cards," in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, 2017. doi: 10.1109/PST.2017.00050 pp. 365–36 509.

[41] "ProVerif codes and prooftrees," Accessed: 17-09-2021. [Online]. Available: https://github.com/unlinkablebdh/UN-2ndGenEMV-BDH

[42] G. Lowe, "A hierarchy of authentication specifications," in *Proceedings 10th Computer Security Foundations Workshop*, June 1997. doi: 10.1109/CSFW.1997.596782. ISSN 1063-6900 pp. 31–43.

[43] C. Cremers and S. Mauw, *Operational Semantics and Verification of Security Protocols*. Springer, 2012. ISBN 978-3-540-78635-1

[44] J. van den Breekel, D. A. Ortiz-Yepes, E. Poll, and J. de Ruiter, "EMV in a nutshell," Tech. Rep., 2016. [Online]. Available: https://www.cs.ru.nl/~erikpoll/papers/EMVtechreport.pdf

[45] "EMVCo Statement – The Advancement of EMV Chip Specifications," EMVCo LLC, Tech. Rep., 2019, accessed: 23-09-2020. [Online]. Available: https://www.emvco.com/wp-content/uploads/documents/2nd-Gen-External-Statement-FINAL.pdf

Appendix

A. Background on normalisation used in the proof of Lemma 3

The *weak normal form* $M{\downharpoonleft}$ of a term M with respect to E captures the least complex (up to multiplication) expression of M. We do not require the weak normal form to be unique.

Definition 9. *(weak normal form) The weak normal $M{\downharpoonleft}$ of a message term M is defined inductively on the structure of M:*

- *$M = \mathrm{g}$ or M is a variable, then $M{\downharpoonleft} = M$.*
- *$M = M_1 \cdot M_2$, then $M{\downharpoonleft} = M_1{\downharpoonleft} \cdot M_2{\downharpoonleft}$.*
- *$M = \phi(M_1, M_2)$, then $M{\downharpoonleft} = \phi(M_1{\downharpoonleft}, M_2{\downharpoonleft})$ if $M_2{\downharpoonleft}$ is ϕ-atomic. Otherwise $M{\downharpoonleft} = \phi\Big(M_1{\downharpoonleft} \cdot M_2^{'}{\downharpoonleft}, M_2^{''}{\downharpoonleft}\Big)$, where $M_2 =_E \phi\Big(M_2^{'}, M_2^{''}\Big)$ and $M_2^{''}{\downharpoonleft}$ is ϕ-atomic.*
- *$M = \langle M_1,\ M_2\rangle$, then $M{\downharpoonleft} = \langle M_1{\downharpoonleft},\ M_2{\downharpoonleft}\rangle$.*
- *$M = \mathtt{h}(M_1)$ or $M = \mathtt{pk}(M_1)$, then $M{\downharpoonleft} = \mathtt{h}(M_1{\downharpoonleft})$ or $M{\downharpoonleft} = \mathtt{pk}(M_1{\downharpoonleft})$ respectively.*
- *$M = \mathtt{sig}(M_1, M_2)$, then $M{\downharpoonleft} = \mathtt{sig}(M_1{\downharpoonleft}, M_2{\downharpoonleft})$ if $M_1{\downharpoonleft}$ is ϕ-atomic. Otherwise $M{\downharpoonleft} = \phi\Big(M_1^{'}{\downharpoonleft}, \mathtt{sig}\Big(M_1^{''}{\downharpoonleft}, M_2{\downharpoonleft}\Big)\Big)$, where $M_1 = \phi\Big(M_1^{'}, M_1^{''}\Big)$ and $M_1^{''}{\downharpoonleft}$ is ϕ-atomic.*
- *$M = \mathtt{fst}(\langle M_1,\ M_2\rangle)$ or $M = \mathtt{snd}(\langle M_1,\ M_2\rangle)$ then $M{\downharpoonleft} = M_1{\downharpoonleft}$ or $M{\downharpoonleft} = M_2{\downharpoonleft}$ respectively.*
- *$M = \mathtt{dec}\Big(\{M_1\}_{M_2}, M_2\Big)$, then $M{\downharpoonleft} = M_1{\downharpoonleft}$.*
- *$M = \mathtt{check}(\mathtt{sig}(M_1, M_2), \mathtt{pk}(M_2))$, then $M{\downharpoonleft} = M_1{\downharpoonleft}$.*
- *Otherwise $M{\downharpoonleft} = M$.*

For a process $\nu\vec{x}.\,(\sigma \mid P)$, the *normalisation* of σ is a frame with recipes allowed in the domain constructed as follows.

1) $u\sigma = M$ for any $u \in \mathrm{dom}(\sigma)$ is replaced by $u\sigma = M{\downharpoonleft}$.
2) If $u\sigma = K_1 \cdot K_2$ and there is a recipe M_1 for an immediate m-factor K_1, then $M_1\sigma$ is added to the normalisation. If there is a recipe M_2 for an immediate m-factor K_2, then $M_2\sigma$ is also added to the normalisation.
3) If $u\sigma = \langle K_1,\ K_2\rangle$, then $u\sigma$ is replaced by $\mathtt{fst}(u)\,\sigma = K_1$ and $\mathtt{snd}(u)\,\sigma = K_2$.
4) If $u\sigma = \{K_1\}_{K_2}$ and there is a recipe M_2 for K_2, then $u\sigma$ is replaced by $\mathtt{dec}(u, M_2)\,\sigma = K_1$.
5) If $u\sigma = \mathtt{sig}(N_1, N_2)$ and there is a recipe M_2 for N_2, then $u\sigma$ is replaced by $\mathtt{check}(u, \mathtt{pk}(M_2))\,\sigma = N_1$.
6) If $u\sigma = \mathtt{sig}(N_1, N_2)$ and there is a recipe M_2 for $\mathtt{pk}(N_2)$, then $\mathtt{check}(u, M_2)\,\sigma = N_1$ is added to the normalisation.

To give an example, consider the extended process

$$\nu\vec{x}.(\sigma|P) \triangleq \nu s,a,b.\Big(\Big\{{}^{\mathtt{pk}(s),\{\mathtt{h}(a)\}_b,b,\mathtt{sig}(\langle a,x\rangle,s)}\!\big/_{pk_s,u_1,u_2,u_3}\Big\}\,|P\Big)$$

Then the normalisation of σ is given below.

$$\begin{aligned}
&pk_s\sigma = \mathtt{pk}(s)\\
&\mathtt{dec}(u_1, u_2)\,\sigma = \mathtt{h}(a)\\
&u_2\sigma = b\\
&\mathtt{fst}(\mathtt{check}(u_3, pk_s))\,\sigma = a\\
&\mathtt{snd}(\mathtt{check}(u_3, pk_s))\,\sigma = x\\
&u_3\sigma = \mathtt{sig}(\langle a,\ x\rangle\,, s)
\end{aligned}$$

The advantage of working with normalisations is the reduction in message complexity in the range of σ without affecting static equivalence: M is a recipe under σ if and only if M is a recipe under the normalisation of σ.

B. *ProvVerif code supporting the proof of Theorem 4*

BDH satisfies injective agreement.

```
free cout, card, term: channel.

type key.
type sskey.
type spkey.
type point.
type scalar.

fun smult(scalar, point): point.
fun h(point): key.
fun pk(sskey): spkey.
fun sign(point , sskey): bitstring.
fun enc(bitstring, key): bitstring.

const G: point [data].

reduc forall m: bitstring, k: key;
 dec(enc(m, k), k) = m.
reduc forall m: point, k: sskey;
 check(sign(m, k), pk(k)) = m.

equation forall a: scalar, b: scalar;
 smult(a, smult(b, G)) =
 smult(b, smult(a, G)).
equation forall a: scalar, b: scalar,
 c: scalar;
 smult(a, smult(b, smult(c, G))) =
 smult(b, smult(a, smult(c, G))).

event snd1(point).
event rec1(point).
event snd2(point).
event rec2(point).
event cardTerm(key, bitstring).
event terminalTerm(key, bitstring).

(*check that terminalTerm is reachable*)
query k: key, z2: bitstring;
 event(terminalTerm(k, z2)).

query k: key, z2: bitstring, z1: point;
 inj-event(terminalTerm(k, z2)) ==>
 (inj-event(rec1(z1)) ==>
 inj-event(snd1(z1))).

query k: key, z2: bitstring, y: point;
 inj-event(terminalTerm(k, z2)) ==>
 (inj-event(rec2(y)) ==>
 inj-event(snd2(y))).

query k: key, z2: bitstring;
 inj-event(terminalTerm(k, z2)) ==>
 inj-event(cardTerm(k, z2)).

let C(s: sskey, c: scalar, ch: channel) =
 new a: scalar;
 event snd1(smult(a, smult(c, G)));
 out(ch, smult(a, smult(c, G)));
 in(ch, y: point);
 event rec2(y);
 let k = h(smult(a, smult(c, y))) in
 let cert = (smult(c, G),
  sign(smult(c, G), s)) in
 event cardTerm(k, enc(((a, smult(c, G)),
  cert), k));
 out(ch, enc(((a, smult(c, G)), cert), k)).

let T(s: sskey, ch: channel) =
 new t: scalar;
 in(ch, z1: point);
 event rec1(z1);
 event snd2(smult(t, G));
 out(ch, smult(t, G));
 in(ch, z2: bitstring);
 let k = h(smult(t, z1)) in
 let ((n1: scalar , n2: point),
  (n3: point, n4: bitstring)) =
   dec(z2, k) in
 if (n2 = check(n4, pk(s))) &&
  (smult(n1, n2) = z1) then
 event terminalTerm(k, z2).

(*populate system with cards*)
let PopCard(s: sskey)=
 new c: scalar;
 !(new chc: channel;
 out(card, chc);
 C(s, c, chc)).

(*populate system with terminals*)
let PopTerminal(s: sskey)=
 new cht: channel;
 out(term, cht);
 T(s, cht).

process
 new s: sskey;
 out(cout, pk(s));
 !PopCard(s) | !PopTerminal(s)
```

UBDH satisfies injective agreement.

```
free cout, card, term: channel.

type key.
type sskey.
type spkey.
type point.
type scalar.

fun smult(scalar, point): point.
fun h(point): key.
fun pk(sskey): spkey.
fun sign(point , sskey): point.
fun enc(bitstring, key): bitstring.

const G: point [data].

reduc forall m: bitstring, k: key;
 dec(enc(m, k), k) = m.
reduc forall a: scalar, m: point,
 k: sskey;
 check(smult(a, sign(m, k)), pk(k)) =
 smult(a, m).

equation forall a: scalar, b: scalar;
 smult(a, smult(b, G)) =
 smult(b, smult(a, G)).
equation forall a: scalar, b: scalar,
 c: scalar;
 smult(a, smult(b, smult(c, G))) =
 smult(b, smult(a, smult(c, G))).


event snd1(point).
event rec1(point).
event snd2(point).
event rec2(point).
event cardTerm(key, bitstring).
event terminalTerm(key, bitstring).

(*check that terminalTerm is reachable*)
query k: key, z2: bitstring;
 event(terminalTerm(k, z2)).

query k: key, z2: bitstring, z1: point;
 inj-event(terminalTerm(k, z2)) ==>
 (inj-event(rec1(z1)) ==>
 inj-event(snd1(z1))).

query k: key, z2: bitstring, y: point;
 inj-event(terminalTerm(k, z2)) ==>
 (inj-event(rec2(y)) ==>
 inj-event(snd2(y))).

query k: key, z2: bitstring;
 inj-event(terminalTerm(k, z2)) ==>
 inj-event(cardTerm(k, z2)).

let C(s: sskey, c: scalar, ch: channel) =
 new a: scalar;
 event snd1(smult(a, smult(c, G)));
 out(ch, smult(a, smult(c, G)));
 in(ch, y: point);
 event rec2(y);
 let k = h(smult(a, smult(c, y))) in
 let m = (smult(a, smult(c, G)),
  smult(a, sign(smult(c, G), s))) in
 event cardTerm(k, enc(m, k));
 out(ch, enc(m, k)).

let T(s: sskey, ch: channel) =
 new t: scalar;
 in(ch, z1: point);
 event rec1(z1);
 event snd2(smult(t, G));
 out(ch, smult(t, G));
 in(ch, z2: bitstring);
 let k = h(smult(t, z1)) in
 let (m1: point, m2: point) =
  dec(z2, k) in
 if (m1 = check(m2, pk(s))) &&
  (m1 = z1) then
 event terminalTerm(k, z2).

(*populate system with cards*)
let PopCard(s: sskey)=
 new c: scalar;
 !(new chc: channel;
 out(card, chc);
 C(s, c, chc)).

(*populate system with terminals*)
let PopTerminal(s: sskey)=
 new cht: channel;
 out(term, cht);
 T(s, cht).

process
 new s: sskey;
 out(cout, pk(s));
 !PopCard(s) | !PopTerminal(s)
```

C. Proof trees for transitions in Theorem 2

By Rule^n below we assume n applications of the transition rule Rule from Fig. 4. In case of n consecutive applications of rules $\mathsf{Par\text{-}l}$, $\mathsf{Par\text{-}r}$ we write Par^n. Notice that α-conversion is often used: in particular when the rule $\mathsf{Extrusion}$ is applied. We define $\chi_l(\vec{Y}, M)$ as the list of message terms obtained by the replacement of lth entry in $\vec{Y}$ with M. In *Case* 5, σ', θ' are the frames accumulated at the point of input of Y_l. In the proof trees presented below we use the following abbreviations

$$S \triangleq \nu c.\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, ch, c)$$

$$I \triangleq \nu c.!\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, ch, c)$$

$$\dfrac{\dfrac{pk_s \mathrel{\#} out, s, !S \quad out =_E out}{\overline{out}\langle \mathtt{pk}(s)\rangle.!S \xrightarrow{\overline{out}(pk_s)} \left(\left\{{}^{\mathtt{pk}(s)}/_{pk_s}\right\}\right) \mid !S}\,\text{Out} \qquad s \mathrel{\#} out, pk_s}{UPD_{\text{spec}} \xrightarrow{\overline{out}(pk_s)} \nu s.\left(\left\{{}^{\mathtt{pk}(s)}/_{pk_s}\right\}\right) \mid !S}\,\text{Res}$$

Case 1. Transition $UPD_{\text{spec}} \xrightarrow{\overline{out}(pk_s)} UPD^{\varnothing}_{\text{spec}}(\varnothing)$.

$$\dfrac{\dfrac{pk_s \mathrel{\#} out, s, !I \quad out =_E out}{\overline{out}\langle \mathtt{pk}(s)\rangle.!I \xrightarrow{\overline{out}(pk_s)} \left(\left\{{}^{\mathtt{pk}(s)}/_{pk_s}\right\}\right) \mid !I}\,\text{Out} \qquad s \mathrel{\#} out, pk_s}{UPD_{\text{impl}} \xrightarrow{\overline{out}(pk_s)} \nu s.\left(\left\{{}^{\mathtt{pk}(s)}/_{pk_s}\right\}\right) \mid !I}\,\text{Res}$$

Case 1. Transition $UPD_{\text{impl}} \xrightarrow{\overline{out}(pk_s)} UPD^{\varnothing,\varnothing}_{\text{impl}}(\varnothing)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{l} u_{L+1} \mathrel{\#} card, ch, C_{\text{upd}}(s, c_{L+1}, ch_{L+1}), \sigma \\ card\sigma =_E card \end{array}}{\begin{array}{l}\overline{card}\langle ch_{L+1}\rangle.C_{\text{upd}}(s, c_{L+1}, ch_{L+1}) \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \sigma\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^{L+1}(ch_{L+1})\end{array}}\,\text{Out} \quad \begin{array}{l} c_{L+1}, ch_{L+1} \mathrel{\#} \\ card, u_{L+1}, \sigma \end{array}}{\begin{array}{l}\sigma \mid S \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu c_{L+1}, ch_{L+1}.(\sigma\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^{L+1}(ch_{L+1}))\end{array}}\,\text{Extrusion}^2 \quad \begin{array}{l} c_{L+1}, ch_{L+1}, \\ u_{L+1} \mathrel{\#} S \end{array}}{\begin{array}{l}\sigma \mid !S \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu c_{L+1}, ch_{L+1}.(\sigma\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^{L+1}(ch_{L+1}) \mid !S)\end{array}}\,\text{Rep-act} \quad \begin{array}{l} c_{L+1}, ch_{L+1}, u_{L+1} \mathrel{\#} \\ C_i, i \leq L \end{array}}{\begin{array}{l}\sigma \mid C_1 \mid \cdots \mid C_L \mid !S \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu c_{L+1}, ch_{L+1}.(\sigma\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid C_1 \mid \cdots \mid C_L \mid \mathcal{E}^{L+1}(ch_{L+1}) \mid !S)\end{array}}\,\text{Par}^{L} \quad \begin{array}{l} s, c_i, ch_i, a_k \\ i \leq L, k \in \beta \cup \gamma \cup \delta \mathrel{\#} \\ card, u_{L+1} \end{array}}{UPD^{\Psi}_{\text{spec}}(\vec{Y}) \xrightarrow{\overline{card}(u_{L+1})} \nu s, c_1, \cdots, c_L, c_{L+1}, ch_1, \cdots, ch_L, ch_{L+1}, a_{l_1}, \cdots, a_{l_K}.(\sigma\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid C_1 \mid \cdots \mid C_L \mid \mathcal{E}^{L+1}(ch_{L+1}) \mid !S)}\,\text{Res}^{1+2L+K}$$

Case 2. Transition $UPD^{\Psi}_{\text{spec}}(\vec{Y}) \xrightarrow{\overline{card}(u_{L+1})} UPD^{\{\alpha\cup\{L+1\},\beta,\gamma,\delta\}}_{\text{spec}}((Y_1, \cdots, Y_L, \varnothing))$.

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\begin{array}{l} u_{L+1} \# card, ch_{L+1}, C_{\text{upd}}(s, c_d, ch_{L+1}), \theta \\ card\theta =_E card \end{array}
}{
\begin{array}{l} \theta \mid \overline{card}\langle ch_{L+1}\rangle.C_{\text{upd}}(s, c_d, ch_{L+1}) \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^d(ch_{L+1}) \end{array}
}\,\text{Out}
\quad
\begin{array}{l} ch_{L+1} \# \\ card, u_{L+1}, \theta \end{array}
}{
\begin{array}{l} \theta \mid \nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_d, ch) \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu ch_{L+1}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^d(ch_{L+1}) \end{array}
}\,\text{Extrusion}
\quad
\begin{array}{l} ch_{L+1}, u_{L+1} \# \\ \nu ch.\overline{card}\langle ch\rangle. \\ C_{\text{upd}}(s, c_d, ch) \end{array}
}{
\begin{array}{l} \theta \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_d, ch) \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu ch_{L+1}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^d(ch_{L+1}) \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_d, ch)) \end{array}
}\,\text{Rep-act}
\quad
\begin{array}{l} ch_{L+1}, u_{L+1} \# \\ C_j^i, i \leq D, j \leq \max\limits_{i\leq D} L_i; \\ \nu ch.\overline{card}\langle ch\rangle. \\ C_{\text{upd}}(s, c_i, ch), \\ i \leq D, i \neq d; !I \end{array}
}{
\begin{array}{l} \theta \mid \cdots \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_d, ch) \mid \cdots \mid !I \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu ch_{L+1}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \cdots \mid \mathcal{E}^d(ch_{L+1}) \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_d, ch) \mid \cdots \mid !I) \end{array}
}\,\text{Par}^{D+L}
\quad
\begin{array}{l} s, c_i, ch_j, a_k, \\ i \leq D, j \leq L, k \in \beta\cup\gamma\cup\delta \# \\ card, u_{L+1} \end{array}
}{
UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{\overline{card}(u_{L+1})} \nu s, c_1, \cdots, c_D, ch_1, \cdots, ch_L, ch_{L+1}, a_{l_1}, \cdots, a_{l_K}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \cdots \mid \mathcal{E}^d(ch_{L+1}) \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_d, ch) \mid \cdots \mid !I)
}\,\text{Res}^{1+D+L+K}
$$

Case 2. Transition $UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{\overline{card}(u_{L+1})} UPD^{\{\alpha\cup\{L+1\},\beta,\gamma,\delta\},\{\cdots,\zeta^d\cup\{L+1\},\cdots\}}_{\text{impl}}((Y_1, \cdots, Y_L, \varnothing))$: card d starts new session.

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\begin{array}{l} u_{L+1} \# card, ch_{L+1}, C_{\text{upd}}(s, c_{D+1}, ch_{L+1}), \theta \\ card\theta =_E card \end{array}
}{
\begin{array}{l} \theta \mid \overline{card}\langle ch_{L+1}\rangle.C_{\text{upd}}(s, c_{D+1}, ch_{L+1}) \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^{D+1}(ch_{L+1}) \end{array}
}\,\text{Out}
\quad
\begin{array}{l} ch_{L+1} \# \\ card, \\ u_{L+1}, \theta \end{array}
}{
\begin{array}{l} \theta \mid \nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_{D+1}, ch) \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu ch_{L+1}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^{D+1}(ch_{L+1}) \end{array}
}\,\text{Extrusion}
\quad
\begin{array}{l} ch_{L+1}, u_{L+1} \# \\ \nu ch.\overline{card}\langle ch\rangle. \\ C_{\text{upd}}(s, c_{D+1}, ch) \end{array}
}{
\begin{array}{l} \theta \mid !\nu ch.\overline{card}\langle ch_{L+1}\rangle.C_{\text{upd}}(s, c_{D+1}, ch) \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu ch_{L+1}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^{D+1}(ch_{L+1}) \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_{D+1}, ch)) \end{array}
}\,\text{Rep-act}
\quad
\begin{array}{l} c_{D+1} \# \\ card, \\ u_{L+1}, \theta \end{array}
}{
\begin{array}{l} \theta \mid I \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu c_{D+1}, ch_{L+1}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^{D+1}(ch_{L+1}) \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_{D+1}, ch)) \end{array}
}\,\text{Extrusion}
\quad
\begin{array}{l} c_{D+1}, \\ ch_{L+1}, \\ u_{L+1} \# I \end{array}
}{
\begin{array}{l} \theta \mid !I \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu c_{D+1}, ch_{L+1}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \mathcal{E}^{D+1}(ch_{L+1}) \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_{D+1}, ch) \mid !I) \end{array}
}\,\text{Rep-act}
\quad
\begin{array}{l} c_{D+1}, ch_{L+1}, \\ u_{L+1} \# C_j^i, \\ i \leq D, j \leq \max\limits_{i\leq D} L_i; \\ \nu ch.\overline{card}\langle ch\rangle. \\ C_{\text{upd}}(s, c_d, ch) \end{array}
}{
\begin{array}{l} \theta \mid \cdots \mid !I \\ \xrightarrow{\overline{card}(u_{L+1})} \\ \nu c_{D+1}, ch_{L+1}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \cdots \mid \mathcal{E}^{D+1}(ch_{L+1}) \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_{D+1}, ch) \mid !I) \end{array}
}\,\text{Par}^{D+L}
\quad
\begin{array}{l} s, c_i, ch_j, a_k, \\ i \leq D, j \leq L, \\ k \in \beta\cup\gamma\cup\delta \# \\ card, u_{L+1} \end{array}
}{
UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{\overline{card}(u_{L+1})} \nu s, c_1, \cdots, c_D, c_{D+1}, ch_1, \cdots, ch_L, ch_{L+1}, a_{l_1}, \cdots, a_{l_K}.(\theta\circ\left\{{}^{ch_{L+1}}/_{u_{L+1}}\right\} \mid \cdots \mid \mathcal{E}^{D+1}(ch_{L+1}) \mid !\nu ch.\overline{card}\langle ch\rangle.C_{\text{upd}}(s, c_{D+1}, ch) \mid !I)
}\,\text{Res}^{1+D+L+K}
$$

Case 2. Transition $UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{\overline{card}(u_{L+1})} UPD^{\{\alpha\cup\{L+1\},\beta,\gamma,\delta\},\Omega\cup\{\{L+1\}\}}_{\text{impl}}((Y_1, \cdots, Y_L, \varnothing))$: a new card is created.

$$\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{l} v_l \# u_l, a_l, \mathcal{F}^l(ch_l, a_l), \sigma \\ u_l\sigma =_E ch_l \end{array}}{\sigma \mid \overline{ch_l}\langle \phi(a_l, \phi(c_l, \mathsf{g}))\rangle.\mathcal{F}^l(ch_l, a_l) \xrightarrow{\overline{u_l}(v_l)} \sigma \circ \left\{^{\phi(a_l,\phi(c_l,\mathsf{g}))}/_{v_l}\right\} \mid \mathcal{F}^l(ch_l, a_l)}\,\text{Out} \qquad a_l \# u_l, v_l, \sigma}{\sigma \mid \nu a.\overline{ch_l}\langle \phi(a, \phi(c_l, \mathsf{g}))\rangle.\mathcal{F}^l(ch_l, a) \xrightarrow{\overline{u_l}(v_l)} \nu a_l.(\sigma \circ \left\{^{\phi(a_l,\phi(c_l,\mathsf{g}))}/_{v_l}\right\} \mid \mathcal{F}^l(ch_l, a_l))}\,\text{Extrusion} \qquad a_l, v_l \# C_i, i \leq L, i \neq l; !S}{\sigma \mid C_1 \mid \cdots \mid \mathcal{E}^l(ch_l) \mid \cdots \mid C_L \mid !S \xrightarrow{\overline{u_l}(v_l)} \nu a_l.(\sigma \circ \left\{^{\phi(a_l,\phi(c_l,\mathsf{g}))}/_{v_l}\right\} \mid \cdots \mid C_K \mid \cdots \mid \mathcal{F}^l(ch_l, a_l) \mid \cdots \mid !S)}\,\text{Par}^L \qquad \begin{array}{l} s, c_i, ch_i, a_k \\ i \leq L, k \in \beta \cup \gamma \cup \delta \# u_l, v_l \end{array}}{UPD^{\Psi}_{\text{spec}}(\vec{Y}) \xrightarrow{\overline{u_l}(v_l)} \nu s, c_1, \cdots, c_L, ch_1, \cdots, ch_L, a_{l_1}, \cdots, a_{l_K}, a_l.(\sigma \circ \left\{^{\phi(a_l,\phi(c_l,\mathsf{g}))}/_{v_l}\right\} \mid \cdots \mid C_K \mid \cdots \mid \mathcal{F}^l(ch_l, a_l) \mid \cdots \mid !S)}\,\text{Res}^{1+2L+K}$$

Case 3. Transition $UPD^{\Psi}_{\text{spec}}(\vec{Y}) \xrightarrow{\overline{u_l}(v_l)} UPD^{\{\alpha\setminus\{l\}, \beta\cup\{l\}, \gamma, \delta\}}_{\text{spec}}(\vec{Y})$, $l \in \alpha$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{l} v_l \# u_l, a_l, \mathcal{F}^d(ch_l, a_l), \theta \\ u_l\theta =_E ch_l \end{array}}{\theta \mid \overline{ch_l}\langle \phi(a_l, \phi(c_d, \mathsf{g}))\rangle.\mathcal{F}^d(ch_l, a_l) \xrightarrow{\overline{u_l}(v_l)} \theta \circ \left\{^{\phi(a_l,\phi(c_d,\mathsf{g}))}/_{v_l}\right\} \mid \mathcal{F}^d(ch_l, a_l)}\,\text{Out} \qquad a_l \# u_l, v_l, \theta}{\theta \mid \nu a.\overline{ch_l}\langle \phi(a, \phi(c_d, \mathsf{g}))\rangle.\mathcal{F}^d(ch_l, a) \xrightarrow{\overline{u_l}(v_l)} \nu a_l.(\theta \circ \left\{^{\phi(a_l,\phi(c_d,\mathsf{g}))}/_{v_l}\right\} \mid \mathcal{F}^d(ch_l, a_l))}\,\text{Extrusion} \qquad \begin{array}{l} a_l, v_l \# C^i_j, \\ i \leq D, j \leq \max\limits_{i \leq D} L_i, \\ j \neq l; !I \end{array}}{\theta \mid \cdots \mid \mathcal{E}^d(ch_l) \mid \cdots \mid !I \xrightarrow{\overline{u_l}(v_l)} \nu a_l.(\theta \circ \left\{^{\phi(a_l,\phi(c_d,\mathsf{g}))}/_{v_l}\right\} \mid \cdots \mid \mathcal{F}^d(ch_l, a_l) \mid \cdots \mid !I)}\,\text{Par}^{D+L} \qquad \begin{array}{l} s, c_i, ch_j, a_k \\ i \leq D, j \leq L, k \in \beta \cup \gamma \cup \delta \# \\ u_l, v_l \end{array}}{UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{\overline{u_l}(v_l)} \nu s, c_1, \cdots, c_D, ch_1, \cdots, ch_L, a_{l_1}, \cdots, a_{l_k}, a_l.(\theta \circ \left\{^{\phi(a_l,\phi(c_d,\mathsf{g}))}/_{v_l}\right\} \mid \cdots \mid \mathcal{F}^d(ch_l, a_l) \mid \cdots \mid !I)}\,\text{Res}^{1+D+L+K}$$

Case 3. Transition $UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{\overline{u_l}(v_l)} UPD^{\alpha\setminus\{l\}, \beta\cup\{l\}, \gamma, \delta\}, \Omega}_{\text{impl}}(\vec{Y})$, $l \in \alpha$.

$$\dfrac{\dfrac{\dfrac{u_l\sigma =_E ch_l}{\sigma \mid ch_l(y).\mathcal{G}^l(ch_l, a_l, y) \xrightarrow{u_l\,Y_l} \sigma \mid \mathcal{G}^l(ch_l, a_l, Y_l\sigma)}\,\text{Inp}}{\sigma \mid C_1 \mid \cdots \mid \mathcal{F}^l(ch_l, a_l) \mid \cdots \mid C_L \mid !S \xrightarrow{u_l\,Y_l} \sigma \mid \cdots \mid \mathcal{G}^l(ch_l, a_l, Y_l\sigma) \mid \cdots \mid !S}\,\text{Par}^L \qquad \begin{array}{l} s, c_i, ch_i, a_k \\ i \leq L, k \in \beta \cup \gamma \cup \delta \# u_l, Y_l \end{array}}{UPD^{\Psi}_{\text{spec}}(\vec{Y}) \xrightarrow{u_l\,Y_l} \nu s, c_1, \cdots, c_L, ch_1, \cdots, ch_L, a_{l_1}, \cdots, a_{l_k}.\{\sigma \mid \cdots \mid \mathcal{G}^l(ch_l, a_l, Y_l\sigma) \mid \cdots \mid !S\}}\,\text{Res}^{1+2L+K}$$

Case 4. Transition $UPD^{\Psi}_{\text{spec}}(\vec{Y}) \xrightarrow{u_l\,Y_l} UPD^{\{\alpha, \beta\setminus\{l\}, \gamma\cup\{l\}, \delta\}}_{\text{spec}}(\chi_l(\vec{Y}, Y_l))$, $l \in \beta$.

$$\dfrac{\dfrac{\dfrac{u_l\theta =_E ch_l}{\theta \mid ch_l(y).\mathcal{G}^d(ch_l, a_l, y) \xrightarrow{u_l\,Y_l} \theta \mid \mathcal{G}^d(ch_l, a_l, Y_l\sigma)}\,\text{Inp}}{\theta \mid \cdots \mid \mathcal{F}^d(ch_l, a_l) \mid \cdots \mid !I \xrightarrow{u_l\,Y_l} \theta \mid \cdots \mid \mathcal{G}^d(ch_l, a_l, Y_l\sigma) \mid \cdots \mid !I}\,\text{Par}^{D+L} \qquad \begin{array}{l} s, c_i, ch_j, a_k \\ i \leq D, j \leq L, k \in \beta \cup \gamma \cup \delta \# \\ u_l, Y_l \end{array}}{UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{u_l\,Y_l} \nu s, c_1, \cdots, c_D, ch_1, \cdots, ch_L, a_{l_1}, \cdots, a_{l_K}.\{\theta \mid \cdots \mid \mathcal{G}^d(ch_l, a_l, Y_l\sigma) \mid \cdots \mid !I\}}\,\text{Res}^{1+D+L+K}$$

Case 4. Transition $UPD^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{u_l\,Y_l} UPD^{\{\alpha, \beta\setminus\{l\}, \gamma\cup\{l\}, \delta\}, \Omega}_{\text{impl}}(\chi_l(\vec{Y}, Y_l))$ if there is a card at the stage $\mathcal{F}$.

$$\dfrac{\dfrac{\dfrac{\begin{array}{l} w_l \# u_l, m^l(a_l, Y_l\sigma'), \sigma \\ u_l\theta =_E ch_l \end{array}}{\sigma \mid \overline{ch_l}\left\langle m^l(a_l, Y_l\sigma')\right\rangle \xrightarrow{\overline{u_l}(w_l)} \sigma \circ \left\{^{m^l(a_l, Y_l\sigma')}/_{w_l}\right\} \mid \mathcal{H}^l}\,\text{Out}}{\sigma \mid C_1 \mid \cdots \mid \mathcal{G}^l(ch_l, a_l, Y_l\sigma') \mid \cdots \mid C_L \mid !S \xrightarrow{\overline{u_l}(w_l)} \sigma \circ \left\{^{m^l(a_l, Y_l\sigma')}/_{w_l}\right\} \mid \cdots \mid \mathcal{H}^l \mid \cdots \mid !S}\,\text{Par}^L \qquad \begin{array}{l} s, c_i, ch_i, a_k \\ i \leq L, k \in \beta \cup \gamma \cup \delta \# u_l, w_l \end{array}}{UPD^{\Psi}_{\text{spec}}(\vec{Y}) \xrightarrow{\overline{u_l}(w_l)} \nu s, c_1, \cdots, c_L, ch_1, \cdots, ch_L, a_{l_1}, \cdots, a_{l_K}.\{\sigma \circ \left\{^{m^l(a_l, Y_l\sigma')}/_{w_l}\right\} \mid \cdots \mid \mathcal{H}^l \mid \cdots \mid !S\}}\,\text{Res}^{1+2L+K}$$

Case 5. Transition $UPD^{\Psi}_{\text{spec}}(\vec{Y}) \xrightarrow{\overline{u_l}(w_l)} UPD^{\{\alpha, \beta, \gamma\setminus\{l\}, \delta\cup\{l\}\}}_{\text{spec}}(\vec{Y})$, $l \in \gamma$.

$$\dfrac{\dfrac{\dfrac{\begin{array}{l} w_l \;\#\; u_l, m^d(a_l, Y_l\theta'), \theta \\ u_l\theta =_E ch_l \end{array}}{\theta \mid \overline{ch_l}\left\langle m^d(a_l, Y_l\theta')\right\rangle \xrightarrow{\overline{u_l}(w_l)} \theta \circ \left\{ {}^{m^d(a_l, Y_l\theta')}\!/_{w_l} \right\} \mid \mathcal{H}^d}\;\text{Out}}{\theta \mid \cdots \mid \mathcal{G}^d(ch_l, a_l, Y_l\theta') \mid \cdots \mid !I \xrightarrow{\overline{u_l}(w_l)} \theta \circ \left\{ {}^{m^d(a_l, Y_l\theta')}\!/_{w_l} \right\} \mid \cdots \mid \mathcal{H}^d \mid \cdots \mid !I}\;\text{Par}^{D+L} \quad \begin{array}{l} s, c_i, ch_j, a_k \\ i <\leq D, j \leq L, k \in \beta \cup \gamma \cup \delta \;\# \\ u_l, w_l \end{array}}{\mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{\overline{u_l}(w_l)} \nu s, c_1, \cdots, c_D, ch_1, \cdots, ch_L, a_{l_1}, \cdots, a_{l_K}.\{\theta \circ \left\{ {}^{m^d(a_l, Y_l\theta')}\!/_{w_l} \right\} \mid \cdots \mid \mathcal{H}^d \mid \cdots \mid !I\}}\;\text{Res}^{1+D+L+K}$$

Case 5. Transition $\mathit{UPD}^{\Psi,\Omega}_{\text{impl}}(\vec{Y}) \xrightarrow{\overline{u_l}(w_l)} \mathit{UPD}^{\{\alpha,\beta,\gamma\backslash\{l\},\delta\cup\{l\}\},\Omega}_{\text{impl}}(\vec{Y})$, $l \in \gamma$.