Defining Privacy is Supposed to be Easy*

Sebastian Mödersheim[1], Thomas Groß[2], and Luca Viganò[3]

[1] DTU Compute, Lyngby, Denmark
[2] School of Computing Science, Newcastle University, UK
[3] Department of Informatics, King's College London, UK

Abstract. Formally specifying privacy goals is not trivial. The most widely used approach in formal methods is based on the static equivalence of frames in the applied *pi*-calculus, basically asking whether or not the intruder is able to distinguish two given worlds. A subtle question is how we can be sure that we have specified all pairs of worlds to properly reflect our intuitive privacy goal. To address this problem, we introduce in this paper a novel and declarative way to specify privacy goals, called α-β privacy, and relate it to static equivalence. This new approach is based on specifying two formulae α and β in first-order logic with Herbrand universes, where α reflects the intentionally released information and β includes the actual cryptographic ("technical") messages the intruder can see. Then α-β privacy means that the intruder cannot derive any "non-technical" statement from β that he cannot derive from α already. We describe by a variety of examples how this notion can be used in practice. Even though α-β privacy does not directly contain a notion of distinguishing between worlds, there is a close relationship to static equivalence of frames that we investigate formally. This allows us to justify (and criticize) the specifications that are currently used in verification tools, and obtain partial tool support for α-β privacy.

1 Introduction

Context and motivation. Several formal notions of privacy have been proposed over the last decade, e.g., [1, 3, 5–7, 9, 13, 17]. Although these notions are quite different, we can probably agree that defining privacy is actually quite subtle and not as easy as it is supposed to be. One of the main reasons is that classical secrecy notions do not apply for data that are not themselves secrets, e.g., a vote is not itself a secret value like a private key. Rather, the information we would like to protect is the *relation* between the (usually non-secret) values, e.g., which voter has cast what vote.

For this reason, the vast majority of the popular approaches to formalizing privacy is based not on the question of what the intruder can deduce from a set of

* This work was partially supported by the EU FP7 Projects no. 318424, "FutureID: Shaping the Future of Electronic Identity" (futureid.eu), and no. 257876, "SPaCIoS: Secure Provision and Consumption in the Internet of Services" (spacios.eu) and the PRIN 2010-11 project "Security Horizons". Much of this work was carried out while L. Viganò was at Dipartimento di Informatica, Università di Verona, Italy.

known messages, but rather whether he can *distinguish* two different worlds.[4] An interesting follow-up question is thus: what is the "right" set of distinguishability questions to define privacy? For instance, in a voting protocol where each user can just vote yes or no, we may check that the intruder cannot distinguish the world where a given voter voted yes from the one where this voter voted no. However, this is not enough: even if the intruder cannot determine the votes, he should also not be able to tell whether two voters have voted the same.

When we look at privacy-friendly identity management, we have even more different kinds of data and possible relations between them, such as date of birth, home address, or different uses of the same credentials. So, how can we ever be confident that a given set of distinguishability questions is sufficient for privacy, i.e., that we have not overlooked some possible connection the intruder could make that we prefer him not to be able to make?

Contributions. In this paper, we take a step back and approach the problem from a different angle. Our main goal is to find a formal description that reflects the idea of privacy in a "natural" and less technical way and that can then be related to the existing privacy notions, supporting or criticizing them. In fact, ultimately we want to use the existing results in this field, but we take the scientific liberty to first think in a slightly different direction.

More specifically, in this paper, we introduce a novel, simple and declarative approach to specify privacy goals, called *α-β privacy*, which is based on specifying two formulae α and β in First-Order Logic with Herbrand Universes [12].

α formalizes the intentionally released information, i.e., the information that we can legitimately give to the intruder, which we also refer to as *payload*. For instance, in a privacy-friendly zero-knowledge credential system (such as IBM's *Idemix* [13]) a user may prove that she is a female older than 18 years (according to an electronic passport she owns), without releasing any more information, such as her name or the precise date of birth. Hence, we have an immediate specification of the data that the user deliberately released, i.e., the statement proved in the zero-knowledge proof, and it is intuitive that we then have a violation of privacy whenever the server who verified the zero-knowledge proof can derive more about the user than the user deliberately released by the proof. Of course, we must exclude from this definition everything that is already entailed by the proved statement, e.g., the fact that the user is also over 15 years old is entailed by the proved statement, so that is not a violation of privacy, but if the server is able to derive that the user is actually over 21 years, then there is a violation. It is thus quite natural to formalize such statements as formulae in some logic and to define privacy as the inability of the intruder to derive statements that are not entailed by what the users have released.

As a counterpart to the "ideal knowledge" provided by the payload α, we also need the *technical information* β, which represents the "actual knowledge" that the intruder has, describing the information (e.g., names, keys ...) that he initially

[4] This is not unlike the earlier paradigm shift in cryptographic definitions from deducibility questions (such as: can the intruder obtain the plaintext of an encrypted message?) to distinguishability questions (such as: can the intruder distinguish the encryption of different chosen values?).

knows, which actual cryptographic messages he has observed and what he knows about these messages. For instance, he may be unable to decrypt a message but anyway know that it has a certain format and contains certain (protected) information, like a vote.

α-β privacy then means that the intruder cannot derive any "non-technical" statement from β that he cannot derive from α already. We believe that this is indeed a simple way to define privacy, and is a more declarative way to talk about privacy than distinguishability of frames. Essentially, the modeler should not think about what technical information the intruder could exploit, but rather what information he is fine to release (α) and what messages are actually exchanged (β).

Another interesting and very declarative feature of our approach is that it is straightforward to model what happens when two intruders collaborate and share their knowledge. α-β privacy allows us to formalize this simply by taking the logical conjunction of the formulae describing the knowledge that the two intruders have, reflecting in a natural way what we can ask the system to provide: The best technology cannot prevent dishonest agents from pooling all the information that they were intentionally given and deriving all possible conclusions from that—but we can ask that they cannot derive more than that.

We describe by a variety of examples how α-β-privacy can be used in practice, and define transition systems based on it. Even though α-β privacy does not directly contain a notion of distinguishing between worlds, there is a close relationship to static equivalence of frames that we investigate formally. This allows us to justify (and criticize) the specifications that are currently used in verification tools and obtain partial tool support for α-β privacy (but we do not discuss these two issues in full detail in this paper). We also prove several results that help in reasoning about α-β privacy in general and give a decision procedure for a fragment of it.

Organization. Section 2 provides the basis for our approach: we discuss First-Order Logic with Herbrand Universes, messages and frames. In Section 3, we formalize α-β-privacy and consider some concrete examples. In Section 4, we discuss automation and the relation of α-β privacy to static equivalence and in Section 5, we draw conclusions. In the accompanying technical report [14], we provide additional examples of how α-β privacy may be employed to model randomized and deterministic encryption, non-determinism, strong secrecy, guessing attacks, anonymous credential systems and pooling of knowledge.

We introduce primitives of our new α-β privacy approach step by step, where Table 1 gives an overview of where they are introduced.

2 Preliminaries

2.1 Herbrand Logic

To formalize our approach, we need to choose an appropriate logic. An obvious candidate is *first-order logic (FOL)*, but this has one difficulty when it comes to the interpretation of the constants and the cryptographic operators. As it is standard in security protocol verification, we would like to interpret these operators either

Table 1. Roadmap of the primitives introduced.

Σ, $\mathcal{V}$, $\mathcal{T}_\Sigma(\mathcal{V})$	§2.1/p.3	Finite alphabet, disjoint set of variables, and terms of our Herbrand Logic (FOL with Herbrand Universes)
F_i	§2.3/p.7	Frame (as in static equivalence), adapted to Herbrand Logic
m_i	§2.3/p.7	Memory location i, storing a piece of intruder knowledge
α	§3/p.8	*Payload*, information that the intruder may legitimately obtain, over $\mathcal{V}_0 \subseteq \mathcal{V}$ and $\Sigma_0 \subseteq \Sigma$
β	§3/p.8	*Technical information* of and about observed protocol messages, over $\mathcal{V}$ and Σ
concr	§2.3/p.7	Encoding of concrete intruder knowledge, ground terms from $\mathcal{T}_\Sigma$
eval	§3.2/p.9	Encoding of structural intruder knowledge, terms from $\mathcal{T}_\Sigma(\mathcal{V})$
ϕ_{axiom}	Table 3	Axioms for generable terms, concrete and structural knowledge

in the free algebra or in the initial algebra induced by a set of algebraic equations; we also call this the *Herbrand Universe*.[5] In general, we cannot enforce the desired interpretation by axioms in FOL (see, e.g., Example 2). There are some workarounds for this, e.g., [4, 11, 16, 18] use first-order Horn theories that are inconsistent (in standard FOL) iff there is an attack in the least Herbrand model, but this construction is not possible for our work because we want to talk about deductions that hold in all Herbrand models of a formula (which does not necessarily have a unique least Herbrand model).

As proposed in [12], FOL with Herbrand universes, or *Herbrand Logic* for short, can be seen as a logic in its own right—as justified, e.g., by Example 2 below. We define *Herbrand Logic* as follows (discussing differences with respect to the definition of [12] below).

Definition 1 (Syntax of Herbrand Logic). *Let $\Sigma = \Sigma_f \uplus \Sigma_i \uplus \Sigma_r$ be an alphabet that consists of a set Σ_f of* free function symbols*, a set Σ_i of* interpreted function symbols *and a set Σ_r of* relation symbols*, all with their arities. To distinguish notation, we write $\mathsf{f}(t_1, \ldots, t_n)$ when $\mathsf{f} \in \Sigma_f$ and $f[t_1, \ldots, t_n]$ when $f \in \Sigma_i$, and we denote the set of considered cryptographic operators by the subset $\Sigma_{op} \subseteq \Sigma_f$.* Constants *are the special case of free function symbols with arity* 0*.*

Let $\mathcal{V}$ be a countable set of variable symbols*, disjoint from Σ. We denote with $\mathcal{T}_\Sigma(\mathcal{V})$ the set of all* terms *that can be built from the function symbols in Σ and the variables in $\mathcal{V}$. We simply write $\mathcal{T}_\Sigma$ when $\mathcal{V} = \emptyset$, and call its elements* ground terms *(over signature Σ).*

We define the set $\mathcal{L}_\Sigma(\mathcal{V})$ of formulae *over the alphabet Σ and the variables $\mathcal{V}$ as usual: relations and equality of terms are* atomic formulae*, and* composed formulae *are built using conjunction $\wedge$, negation $\neg$, and existential quantification $\exists$.*

[5] Note that it is common to define the Herbrand Universe as the free term algebra but for our purposes it is crucial to also include algebraic properties of the operators, as illustrated in Example 1.

We employ the standard syntactic sugar and write, for example, $\forall x.\, \phi$ for $\neg\exists x.\, \neg\phi$. We also write $x \in \{t_1, \ldots, t_n\}$ to abbreviate $x = t_1 \vee \ldots \vee x = t_n$. The function *fv* returns the set of *free variables* of a formula as expected.

Definition 2 (Herbrand Universe and Algebra). *Formulae in Herbrand logic are always interpreted with respect to a given fixed set Σ_f of free symbols (since this set may contain symbols that do not occur in the formulae) and a congruence relation $\approx$ on $\mathcal{T}_{\Sigma_f}$. We may annotate all notions of the semantics with Σ_f and $\approx$ when it is not clear from the context.*

We write $[t]_\approx = \{t' \in \mathcal{T}_{\Sigma_f} \mid t \approx t'\}$ to denote the equivalence class *of a term $t \in \mathcal{T}_{\Sigma_f}$ with respect to $\approx$. Further, let $U = \{[t]_\approx \mid t \in \mathcal{T}_{\Sigma_f}\}$ be the set of all equivalence classes. We call U the* Herbrand universe *(since it is freely generated by the function symbols of Σ_f modulo $\approx$). Based on U, we define a Σ_f-*algebra *$\mathcal{A}$ that interprets every n-ary function symbol $\mathsf{f} \in \Sigma_f$ as a function $\mathsf{f}^\mathcal{A} : U^n \to U$ in the following standard way. $\mathsf{f}^\mathcal{A}([t_1]_\approx, \ldots, [t_n]_\approx) = [\mathsf{f}(t_1, \ldots, t_n)]_\approx$, where the choice of the representatives $t_1, \ldots, t_n$ of the equivalence classes is irrelevant because $\approx$ is congruent. $\mathcal{A}$ is sometimes also called the* quotient algebra *(in the literature sometimes denoted with $\mathcal{T}_{\Sigma_f} / \approx$).*

Example 1. As an example, suppose the congruence relation $\approx$ is given by a set of equations like $\forall x, y.\, x{+}y \approx y{+}x$ for some binary function symbols $+$ and $-$ in Σ_f. Then we have in the quotient algebra $\mathsf{5{+}3} \approx \mathsf{3{+}5}$ but still $\mathsf{3{+}5} \not\approx \mathsf{(7{-}4){+}5}$. Thus, the quotient algebra is the *finest* (or "free-est") interpretation still compatible with the given algebraic properties. □

Definition 3 (Semantics of Herbrand Logic). *An* interpretation *$\mathcal{I}$ maps every interpreted function symbol $f \in \Sigma_i$ of arity n to a function $\mathcal{I}(f) : U^n \to U$ on the Herbrand universe, every relation symbol $r \in \Sigma_r$ of arity n to a relation $\mathcal{I}(r) \subseteq U^n$ on the Herbrand universe, and every variable $x \in \mathcal{V}$ to an element of U.*

We extend $\mathcal{I}$ to a function on $\mathcal{T}_\Sigma(\mathcal{V})$ as expected: $\mathcal{I}(\mathsf{f}(t_1, \ldots, t_n)) = \mathsf{f}^\mathcal{A}(\mathcal{I}(t_1), \ldots, \mathcal{I}(t_n))$ for $\mathsf{f} \in \Sigma_f$ and $\mathcal{I}(\mathsf{f}[t_1, \ldots, t_n]) = \mathcal{I}(\mathsf{f})(\mathcal{I}(t_1), \ldots, \mathcal{I}(t_n))$.

We define that $\mathcal{I}$ is a model *of formula ϕ, in symbols $\mathcal{I} \models \phi$, as follows:*

$$\begin{array}{ll}
\mathcal{I} \models s = t & \textit{iff } \mathcal{I}(s) = \mathcal{I}(t) \\
\mathcal{I} \models r(t_1, \ldots, t_n) & \textit{iff } (\mathcal{I}(t_1), \ldots, \mathcal{I}(t_n)) \in \mathcal{I}(r) \\
\mathcal{I} \models \phi \wedge \psi & \textit{iff } \mathcal{I} \models \phi \textit{ and } \mathcal{I} \models \psi \\
\mathcal{I} \models \neg\phi & \textit{iff not } \mathcal{I} \models \phi \\
\mathcal{I} \models \exists x.\, \phi & \textit{iff there is a } c \in U \textit{ such that } \mathcal{I}[x \mapsto c] \models \phi
\end{array}$$

where $\mathcal{I}[x \mapsto c]$ denotes the interpretation that is identical to $\mathcal{I}$ except that x is mapped to c. Entailment *$\phi \models \psi$ is defined as $\mathcal{I} \models \phi$ implies $\mathcal{I} \models \psi$ for all interpretations $\mathcal{I}$. We write $\phi \equiv \psi$ when both $\phi \models \psi$ and $\psi \models \phi$. We also use $\equiv$ in the definitions of formulae.*

Example 2. Similar to [12], we can axiomatize arithmetic in Herbrand logic; simply let $\Sigma_f = \{\mathsf{z}/0, \mathsf{s}/1\}$, representing 0 and (+1), let $\approx$ be syntactic equality on $\mathcal{T}_{\Sigma_f}$, and let $\Sigma_i = \{add/2, mult/2\}$ and $\Sigma_r = \{<\}$ with the following formula:

$$\begin{aligned}
\phi \equiv \forall x, y.\; & add[\mathsf{z}, y] = y \;\wedge\; add[\mathsf{s}(x), y] = add[x, \mathsf{s}(y)] \;\wedge\; mult[\mathsf{z}, y] = \mathsf{z} \;\wedge \\
& mult[\mathsf{s}(x), y] = add[y, mult[x, y]] \;\wedge\; x < \mathsf{s}(x) \;\wedge\; x < y \implies x < \mathsf{s}(y)
\end{aligned}$$

Table 2. Example set Σ_{op}: standard cryptographic constructors, destructors and verifiers.

Constructors	**Destructors**	**Verifiers**	**Meaning**
$\mathsf{crypt}(k, r, t)$	$\mathsf{dcrypt}(k, t)$	$\mathsf{vcrypt}(k, t)$	Asymmetric encryption of t with public key k and randomness r. Decryption with private key k.
$\mathsf{scrypt}(k, r, t)$	$\mathsf{dscrypt}(k, t)$	$\mathsf{vscrypt}(k, t)$	Symmetric encryption of t with secret key k and randomness r.
$\mathsf{sign}(k, t)$	$\mathsf{retrieve}(t)$	$\mathsf{vsig}(k, t)$	Signature of t with private key k; verification with public key k.
$\mathsf{pub}(s), \mathsf{priv}(s)$			Asymmetric key pair generated from seed s.
$\mathsf{pair}(t_1, t_2)$	$\mathsf{proj}_i(t)$	$\mathsf{vpair}(t)$	Concatenation of messages t_1 and t_2.
$\mathsf{h}(t)$			Hash of message t.

Then $\phi \models \psi$ iff ψ is a true arithmetic statement. It is well-known that (as a consequence of Löwenheim-Skolem's theorem or of Gödel's incompleteness theorem, see [10]) an equivalent axiomatization cannot be achieved in standard FOL. □

We note the following three differences with respect to the definition of Herbrand logic in [12]. First, in [12] and as is standard, the Herbrand universe is the free term algebra, forbidding one to model algebraic properties of the free operators. Our definition is a generalization to equivalence classes modulo the $\approx$ relation (and $\approx$ can simply be set to be the syntactic equality on $\mathcal{T}_{\Sigma_f}$ to get the free algebra). Second, the logic in [12] treats free variables as implicitly universally quantified, which is quite non-standard. In our definition, an interpretation of a formula includes the interpretation of the free variables as is standard. This is, of course, without loss of expressiveness since one can quantify variables when this is what one wants to express. Third, the logic in [12] does not have interpreted functions and, in fact, these are syntactic sugar: an interpreted n-ary function symbol f can be modeled by an $n+1$-ary relation R_f symbol with the axiom $\forall x_1, \ldots, x_n . \exists y.\, R_f(x_1, \ldots, x_n, y) \wedge \forall y'.\, R_f(x_1, \ldots, x_n, y') \implies y = y'$.

2.2 Messages, Operators and Algebraic Properties

We adopt the common black-box ("Dolev-Yao style" [8]) algebraic model of the cryptographic operations. We consider, in this paper, the example set Σ_{op} of standard operators given, together with their intuitive meanings, in Table 2. Let $\approx$ be the smallest relation so that for all terms s, r, t, t_1, t_2 in $\mathcal{T}_{\Sigma_f}$ and for $i \in \{1, 2\}$:

$$\begin{array}{ll}
\mathsf{dcrypt}(\mathsf{priv}(s), \mathsf{crypt}(\mathsf{pub}(s), r, t)) \approx t & \mathsf{vcrypt}(\mathsf{priv}(s), \mathsf{crypt}(\mathsf{pub}(s), r, t)) \approx \mathsf{yes} \\
\mathsf{retrieve}(\mathsf{sign}(\mathsf{priv}(s), t)) \approx t & \mathsf{vsig}(\mathsf{pub}(s), \mathsf{sign}(\mathsf{priv}(s), t)) \approx \mathsf{yes} \\
\mathsf{dscrypt}(k, \mathsf{scrypt}(k, r, t)) \approx t & \mathsf{vscrypt}(k, \mathsf{scrypt}(k, r, t)) \approx \mathsf{yes} \\
\mathsf{proj}_i(\mathsf{pair}(t_1, t_2)) \approx t_i & \mathsf{vpair}(\mathsf{pair}(t_1, t_2)) \approx \mathsf{yes}
\end{array}$$

The equations induce a *congruence relation* $\approx$ on terms, and we interpret all functions in the Herbrand universe modulo this congruence as explained above, i.e., two terms are equal iff that is a consequence of $\approx$ with respect to Σ_{op}.

2.3 Frames

Frames and the notion of their static equivalence are a standard way to formalize privacy goals in formal methods, e.g., [5–7]. We define them here in a slightly non-standard way that is more convenient to directly formalize them in Herbrand logic and later relate them to our concept of α-β privacy (we point the reader to [14] for a detailed discussion on the differences between the standard definition of frames and the one we consider here). *Frames* are written as

$$\mathcal{F} = \{\mathsf{m}_1 \mapsto t_1, \ldots, \mathsf{m}_l \mapsto t_l\}$$

where the m_i are distinguished constants and the t_i are ground terms that do not contain any m_i. This frame represents that the intruder *knows* l messages $t_1, \ldots, t_l$ that he can "refer to" as $\mathsf{m}_1, \ldots, \mathsf{m}_l$. In contrast to the standard Dolev-Yao intruders, we thus do not model the intruder knowledge by a set of messages $\{t_1, \ldots, t_l\}$, but we give each message a unique label m_i. This allows us to talk about checks that the intruder can make, e.g., whether hashing the value at m_1 gives the same value as the one stored at m_2. We may thus refer to the m_i as *memory locations* in the intruder's memory.

We define *the terms that the intruder can generate from his knowledge* as the least set that contains $\mathsf{m}_1, \ldots, \mathsf{m}_l$ and is closed under all the cryptographic operators that the intruder can employ. For the example operators of Σ_{op} shown in Table 2, we can formalize this in Herbrand Logic with a formula $\phi_{gen}(l)$, which uses a new predicate $gen(t)$ to represent that the intruder can generate t. Hence, in contrast to the standard Dolev-Yao definition, the intruder does not directly compose the terms he knows but rather he builds what is sometimes called *recipes* by applying operators to the memory locations he has.

The axiom $\phi_{gen}(l)$ is shown in Table 3, together with the other axioms that we will employ in α-β privacy. For a different set Σ_{op} of cryptographic operators the definition is analogous: using semi-formal notation, $\phi_{gen}(l)$ would have the form

$$\begin{aligned}\phi_{gen}(l) \equiv \forall x.\ gen(x) \iff (x \in \{\mathsf{m}_1, \ldots, \mathsf{m}_l\} \vee \\ \textstyle\bigvee_{\mathsf{f} \in \Sigma_{op}} \exists x_1 \ldots x_n.\ x = \mathsf{f}(x_1, \ldots, x_n) \wedge gen(x_1) \ldots gen(x_n))\end{aligned}$$

The axiom $\phi_{Fr}(\mathcal{F})$ in Table 3 allows us to encode the frame $\mathcal{F} = \{\mathsf{m}_1 \mapsto t_1, \ldots, \mathsf{m}_l \mapsto t_l\}$ into Herbrand logic using an interpreted function symbol $concr[\cdot]$ that yields the concrete message stored for a memory location, and the axiom ϕ_{concr} extends the definition of $concr[\cdot]$ congruently for the application of cryptographic operators, so that $concr[t]$ is determined for all terms t that the intruder can generate.

In the following, we use examples with two frames $\mathcal{F}_0$ and $\mathcal{F}_1$, both with the same length l. We use functions $concr_0[t]$ and $concr_1[t]$ for their respective encodings (and denote the above axiom as ϕ_{concr_0} and ϕ_{concr_1} as expected).

Example 3. Consider the frame (from [6]): $\mathcal{F}_0 = \{\mathsf{m}_1 \mapsto \mathsf{scrypt}(\mathsf{k}, \mathsf{r}_1, \mathsf{n}_1), \mathsf{m}_2 \mapsto \mathsf{pair}(\mathsf{n}_1, \mathsf{n}_2), \mathsf{m}_3 \mapsto \mathsf{k}\}$. We have, for instance, that the intruder can obtain n_1. Let $\Phi \equiv \phi_{Fr}(\mathcal{F}_0) \wedge \phi_{concr_0} \wedge \phi_{gen}(3)$. Then we have, e.g., $\Phi \models gen(\mathsf{dscrypt}(\mathsf{m}_3, \mathsf{m}_1)) \wedge concr_0[\mathsf{dscrypt}(\mathsf{m}_3, \mathsf{m}_1)] = \mathsf{n}_1$. Note that we have $\Phi \models concr_0[\mathsf{dscrypt}(\mathsf{m}_3, \mathsf{m}_1)] =$

Table 3. Axioms used in α-β privacy (for the example set Σ_{op}).

$$\begin{aligned}
\phi_{gen}(l) \equiv\ & \forall x.\ gen(x) \iff (x \in \{\mathsf{m}_1,\ldots,\mathsf{m}_l\} \vee \\
& (\exists x_1,x_2,x_3.\ x = \mathsf{crypt}(x_1,x_2,x_3) \wedge gen(x_1) \wedge gen(x_2) \wedge gen(x_3)) \vee \\
& (\exists x_1,x_2.\ x = \mathsf{dcrypt}(x_1,x_2) \wedge gen(x_1) \wedge gen(x_2)) \ \vee \ldots \vee \\
& (\exists x_1.\ x = \mathsf{h}(x_1) \wedge gen(x_1))) \quad \text{for a length } l \\
\phi_{Fr}(\mathcal{F}) \equiv\ & concr[\mathsf{m}_1] = t_1 \wedge \ldots \wedge concr[\mathsf{m}_l] = t_l \quad \text{for a frame } \mathcal{F} \text{ of length } l \\
\phi_{concr} \equiv\ & \forall x_1,x_2,x_3,y_1,y_2,y_3.\,(concr[x_1] = y_1 \ \wedge concr[x_2] = y_2 \wedge concr[x_3] = y_3) \implies \\
& (concr[\mathsf{crypt}(x_1,x_2,x_3)] = \mathsf{crypt}(y_1,y_2,y_3) \ \wedge \\
& concr[\mathsf{dcrypt}(x_1,x_2)] = \mathsf{dcrypt}(y_1,y_2) \ \wedge \ldots \wedge concr[\mathsf{h}(x_1)] = \mathsf{h}(y_1)) \\
\phi_{eval} \equiv\ & \forall x_1,x_2,x_3,y_1,y_2,y_3.\,(eval[x_1] = y_1 \ \wedge eval[x_2] = y_2 \wedge eval[x_3] = y_3) \implies \\
& (eval[\mathsf{crypt}(x_1,x_2,x_3)] = \mathsf{crypt}(y_1,y_2,y_3) \ \wedge \\
& eval[\mathsf{dcrypt}(x_1,x_2)] = \mathsf{dcrypt}(y_1,y_2) \ \wedge \ldots \wedge eval[\mathsf{h}(x_1)] = \mathsf{h}(y_1)) \\
\phi_{struct} \equiv\ & \forall x,y.\,(concr[x] = concr[y] \iff eval[x] = eval[y])
\end{aligned}$$

$concr_0[\mathsf{proj}_1(\mathsf{m}_2)]$, i.e., the intruder can check that the decrypted term is equal to the first component of m_2. $\square$

Definition 4 (Static Equivalence of Frames). *Two frames $\mathcal{F}_0$ and $\mathcal{F}_1$ of the same length l are* statically equivalent *(in symbols, $\mathcal{F}_0 \sim \mathcal{F}_1$) iff for any pair of generable terms either both frames give the same result or both frames give a different result. Formally, $\mathcal{F}_0 \sim \mathcal{F}_1$ iff*

$$\begin{aligned}
& \phi_{gen}(l) \ \wedge \ \phi_{Fr}(\mathcal{F}_0) \ \wedge \ \phi_{Fr}(\mathcal{F}_1) \ \wedge \ \phi_{concr_0} \ \wedge \ \phi_{concr_1} \models \\
& \quad \forall x,y.\ (gen(x) \ \wedge \ gen(y)) \implies (concr_0[x] = concr_0[y] \iff concr_1[x] = concr_1[y])
\end{aligned}$$

Example 4. We can distinguish $\mathcal{F}_0$ of Example 3 from the frame $\mathcal{F}_1 = \{\mathsf{m}_1 \mapsto \mathsf{scrypt}(\mathsf{k},\mathsf{r}_1,\mathsf{n}_3), \mathsf{m}_2 \mapsto \mathsf{pair}(\mathsf{n}_1,\mathsf{n}_2), \mathsf{m}_3 \mapsto \mathsf{k}\}$ since the check $concr_1[\mathsf{dscrypt}(\mathsf{m}_3,\mathsf{m}_1)] = concr_1[\mathsf{proj}_1(\mathsf{m}_2)]$ fails, whereas it succeeds for $concr_0$. $\square$

3 A New Privacy Model: α-β Privacy

We introduce α-β privacy step by step: First, in Section 3.1 we introduce the distinction between payload formulae α and technical formulae β as well as the notion of *interesting* derivation from β. Second, in Section 3.2, we establish the methodology to reason over such formulae, introducing a further function $eval[\cdot]$ similar to $concr[\cdot]$ that represents the structural information the intruder has about his knowledge. Third, in Section 3.3 we extend the privacy notion to transition systems, and, finally, in Section 3.4 we discuss further examples of α-β privacy.

3.1 Payload and Technical Information

Our model is inspired by zero-knowledge proofs for privacy (as they are used, e.g., in IBM's Idemix [13]). The following points are characteristic for such proofs:

– The prover (intentionally) conveys some information to the verifier, i.e., the statement being proved to the verifier. We call this statement the *payload* α.

– The participants also (inevitably) convey some cryptographic information (e.g., commitments, challenges, and responses) that, if the scheme is secure, do *not* reveal anything "interesting" besides α; this, of course, is the very reason why such a scheme is called zero-knowledge. We call this kind of information the *technical information* β.

Here the term "interesting" is often defined in the cryptographic world by the fact that it is computationally easy to produce a fake transcript of zero-knowledge proofs that is statistically indistinguishable from a real transcript. Hence, whatever information could possibly be obtained from β one may have created oneself. This kind of definition is, however, quite unhandy in logical reasoning, and it applies only to (some types of) zero-knowledge proofs.

In this paper, we show that it is fortunately possible to define the term "interesting" on a logical basis that makes sense for many actual situations in which we want to talk about privacy. The key idea is that the payload α may be formulated over a restricted alphabet $\Sigma_0 \subsetneq \Sigma$, whereas the technical information β may talk about the full alphabet Σ. For instance, all cryptographic operators are part of $\Sigma \setminus \Sigma_0$.

Definition 5. *Let $\Sigma_0 \subsetneq \Sigma$. Given a payload formula $\alpha \in \mathcal{L}_{\Sigma_0}(\mathcal{V})$ and a technical formula $\beta \in \mathcal{L}_{\Sigma}(\mathcal{V})$, where $\beta \models \alpha$ and $fv(\alpha) = fv(\beta)$ and both α and β are consistent, we say that a statement $\alpha' \in \mathcal{L}_{\Sigma_0}(fv(\alpha))$ is an* interesting derivation from β (with respect to α) *if $\beta \models \alpha'$ but $\alpha \not\models \alpha'$. We say that β* respects the privacy of α *if the intruder cannot derive any interesting statement from β, and that β* violates the privacy of α *otherwise.*

We have defined the notion of an interesting derivation α' as anything the intruder may be able to derive from his observations β as long as it is a non-technical statement (i.e., of $\mathcal{L}_{\Sigma_0}$) and it does not follow from α alone, i.e., from what he is permitted to know anyway. This allows us to capture that the intruder may well see a few technical details, e.g., that two messages come from the same IP address, but that in itself is not very interesting as long as he cannot tie that to a relevant information α'.

Another aspect of this definition is that by the information α that we gave out, also all information that can be derived from α is given out, because the best cryptographic systems cannot protect us from the intruder drawing conclusions. In general, the weaker α is (i.e., the less information we deliberately release to the intruder) and the stronger β is (i.e., the more information we assume the intruder might actually have), the stronger is the notion of privacy. So, as a rule of thumb, when a modeler is in doubt, one should be restrictive on α and generous on β.

3.2 Privacy on Messages

We look at a fixed state of a complex system and ask whether the intruder can violate privacy in this state. Let us start with an example:

Example 5. Let the payload alphabet be $\Sigma_0 = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ and let us model that users choose values x from Σ_0. This is the only information we want to give the intruder. Suppose there is a protocol in place where each user sends out a message $\mathsf{h}(\mathsf{pair}(\mathsf{n}, x))$ that the intruder can observe, that is, a hash of the choice x and a fixed number n (that is a secret from $\Sigma \setminus \Sigma_0$). Obviously, using such a fixed number, even though secret from the intruder, is a risk for "guessing attacks". Suppose further that the intruder has previously observed the message $\mathsf{h}(\mathsf{pair}(\mathsf{n}, \mathsf{a}))$ and thus that he knows that the choice in this case was a. Let us finally assume that a user has chosen $x = \mathsf{b}$ and thus sent out $\mathsf{h}(\mathsf{pair}(\mathsf{n}, \mathsf{b}))$. □

We want to reflect that, in this example, the intruder knows not only the concrete message $\mathsf{h}(\mathsf{pair}(\mathsf{n}, \mathsf{b}))$, but also the *structural information* that this message has the form $\mathsf{h}(\mathsf{pair}(\mathsf{n}, x))$ where x is the choice we are interested in.

For this reason, we use the *concr* function as before to represent concrete knowledge and further introduce, as a fundamental part of α-β privacy, an interpreted unary function symbol *eval* that works similar to *concr* and *maps memory locations to the structural information that the intruder has about the terms in his knowledge.* Here is one possible way to model Example 5 in Herbrand logic:

$$\begin{aligned}
&\alpha \equiv x \in \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}\\
&\beta \equiv \alpha \ \wedge\ \phi_{gen}(5) \ \wedge\ \phi_{concr} \ \wedge\ \phi_{eval} \ \wedge\ \phi_{struct} \ \wedge\ \mathit{concr}[\mathsf{m}_1] = \mathit{eval}[\mathsf{m}_1] = \mathsf{a} \ \wedge\\
&\qquad \mathit{concr}[\mathsf{m}_2] = \mathit{eval}[\mathsf{m}_2] = \mathsf{b} \ \wedge\ \mathit{concr}[\mathsf{m}_3] = \mathit{eval}[\mathsf{m}_3] = \mathsf{c} \ \wedge\\
&\qquad \mathit{concr}[\mathsf{m}_4] = \mathit{eval}[\mathsf{m}_4] = \mathsf{h}(\mathsf{pair}(\mathsf{n}, \mathsf{a})) \ \wedge\ \mathit{concr}[\mathsf{m}_5] = \mathsf{h}(\mathsf{pair}(\mathsf{n}, \mathsf{b})) \ \wedge\\
&\qquad \mathit{eval}[\mathsf{m}_5] = \mathsf{h}(\mathsf{pair}(\mathsf{n},x))
\end{aligned}$$

where the axioms ϕ_{eval} and ϕ_{struct} are as defined in Table 3 (we will explain them in detail below).

For most part, the structural information is identical to the concrete information, only for the field m_5 we have a difference between *eval* and *concr*. This is indeed a major point for our model: for the choice $x = \mathsf{b}$ (i.e., "what really happened"), and only for this choice, we have that $\mathit{concr}[\mathsf{m}_5] = \mathit{eval}[\mathsf{m}_5]$ but the intruder a priori has no way to check that. However, the axiom ϕ_{eval} allows him to derive the structure of terms he can generate, and most importantly ϕ_{struct} tells us that two generable terms have the same concrete value iff they have the same structure. In this example, we can exploit ϕ_{struct}: from $\mathit{concr}[\mathsf{m}_4] \neq \mathit{concr}[\mathsf{m}_5]$ (remember that all terms are interpreted in the Herbrand universe) we can conclude $\mathit{eval}[\mathsf{m}_4] \neq \mathit{eval}[\mathsf{m}_5]$, so that $\mathsf{h}(\mathsf{pair}(\mathsf{n}, \mathsf{a})) \neq \mathsf{h}(\mathsf{pair}(\mathsf{n}, x))$ and thus $x \neq \mathsf{a}$ (again since terms are interpreted in the Herbrand universe). Hence, the intruder can derive from β the Σ_0-formula $\alpha' \equiv x \in \{\mathsf{b}, \mathsf{c}\}$ that does not follow from α. Thus, in this example, β does not respect the privacy of α. Note that the intruder cannot derive more, which is—very declaratively—because β has both a model in which $x = \mathsf{b}$, and one where $x = \mathsf{c}$, so the intruder was not even able to determine the choice x, he was only able to exclude one interpretation, namely $x = \mathsf{a}$.

Message Analysis The form of α and β that we have used for Example 5 is at the core of many specifications, namely, when the intruder has observed a set of messages and knows their structure. For this reason, we define a particular fragment of α-β privacy (for which we give some decidability results in Section 4.2) that deals only with *combinatoric* α and only with the analysis of messages similar to the previous example.[6]

Definition 6. *We call $\alpha \in \mathcal{L}_{\Sigma_0}(\mathcal{V})$* combinatoric *if Σ_0 is finite and consists only of free constants. Let α be combinatoric and σ a substitution of the free variables of α to elements of Σ_0 so that $\sigma(\alpha)$ is consistent. We say that β is a* message-analysis problem *(with respect to α and σ) iff there are $t_1, \ldots, t_l \in \mathcal{T}_\Sigma(fv(\alpha))$ such that*

$$\beta \equiv \alpha \wedge \phi_{gen}(l) \wedge \phi_{concr} \wedge \phi_{eval} \wedge \phi_{struct} \wedge \bigwedge_{i=1}^{l} concr[\mathsf{m}_i] = \sigma(t_i) \wedge eval[\mathsf{m}_i] = t_i$$

In general, such a β allows us to model a system where messages t_i have been exchanged that depend on some payload values $fv(\alpha)$ and the intruder has seen the concrete instantiations $\sigma(t_i)$ of these messages. Typically, the intruder knowledge will contain all the values of Σ_0 but he does not know the substitution σ, i.e., how the payload variables were actually chosen from Σ_0. What he knows, however, is the structure of the terms, i.e., where these variables occur in the t_i, because this structural information is usually part of a publicly available protocol description. He can try to exploit comparisons (ϕ_{struct}) with the actual terms $\sigma(t_i)$ and their compositions (ϕ_{concr} and ϕ_{eval}).

Some Variants of Example 5 One may, of course, consider a similar use of variables for non-payload secrets, like the value n in Example 5. However, since we require that α and β have the same set of free variables, one would then existentially quantify that value; for instance, for Example 5:

$$\beta \equiv \exists y.\ \ldots concr[\mathsf{m}_4] = \mathsf{h}(\mathsf{pair}(\mathsf{n},\mathsf{a})) \ \wedge\ eval[\mathsf{m}_4] = \mathsf{h}(\mathsf{pair}(y,\mathsf{a})) \ \wedge \\ concr[\mathsf{m}_5] = \mathsf{h}(\mathsf{pair}(\mathsf{n},\mathsf{b})) \ \wedge\ eval[\mathsf{m}_5] = \mathsf{h}(\mathsf{pair}(y,x))$$

Without the existential quantifier (if y were left free), the intruder could derive, e.g., that $y \neq \mathsf{a}$ (by generating $\mathsf{h}(\mathsf{pair}(\mathsf{m}_1,\mathsf{m}_1))$ and comparing the result with m_4). The $\exists$ thus intuitively expresses that we are not interested in the concrete value of y—the goal is not the protection of the nonces in the hash-values, so if they are found out, then it is *not* in itself a violation of privacy (but may lead to one).

Let us briefly also consider three variants of the example. First, if the intruder also knows n, say, $concr[\mathsf{m}_6] = eval[\mathsf{m}_6] = \mathsf{n}$, then he can indeed derive $x = \mathsf{b}$, because he can verify that $\mathsf{h}(\mathsf{pair}(\mathsf{m}_6,\mathsf{m}_2))$ gives the same concrete value as m_4.

Second, if users use different nonces that the intruder does not know, i.e., $\beta \equiv \ldots concr[\mathsf{m}_4] = eval[\mathsf{m}_4] = \mathsf{h}(\mathsf{pair}(\mathsf{n}_1,\mathsf{a})) \wedge concr[\mathsf{m}_5] = \mathsf{h}(\mathsf{pair}(\mathsf{n}_2,\mathsf{b})) \wedge eval[\mathsf{m}_5] = \mathsf{h}(\mathsf{pair}(\mathsf{n}_2,x))$, then β indeed preserves the privacy of α. To see this, note that β has models with $x = \mathsf{a}$, with $x = \mathsf{b}$, and with $x = \mathsf{c}$. Thus, every Σ_0-formula α' that follows β also follows from α.

Third, we have so far seen the message in m_4 as a message that was sent previously by some agent and we are not interested in protecting that, and, in fact,

[6] We could consider other forms of "combinatoric" α, e.g., such that Σ_0 may contain infinitely many free constants and function symbols as long as α admits only finitely many models (up to isomorphism). We leave a detailed investigation to future work.

we had assumed that the intruder already knows that it contained the choice a. We can now also model that we are interested in protecting both choices as follows:

$$\begin{aligned}\alpha &\equiv x_1 \in \{\mathsf{a},\mathsf{b},\mathsf{c}\} \ \wedge \ x_2 \in \{\mathsf{a},\mathsf{b},\mathsf{c}\}\\ \beta &\equiv \ldots \mathit{concr}[\mathsf{m}_4] = \mathsf{h}(\mathsf{pair}(\mathsf{n}_1,\mathsf{a})) \ \wedge \ \mathit{eval}[\mathsf{m}_4] = \mathsf{h}(\mathsf{pair}(\mathsf{n}_1,x_1)) \ \wedge\\ &\quad\ \mathit{concr}[\mathsf{m}_5] = \mathsf{h}(\mathsf{pair}(\mathsf{n}_2,\mathsf{b}) \ \wedge \ \mathit{eval}[\mathsf{m}_5] = \mathsf{h}(\mathsf{pair}(\mathsf{n}_2,x_2))\end{aligned}$$

Here again β respects the privacy of α because we can find a model for each combination of values for $x_1, x_2 \in \{\mathsf{a},\mathsf{b},\mathsf{c}\}$. In contrast, if we had used the same nonce (replacing both n_1 and n_2 with n), we would have that $\mathit{concr}[\mathsf{m}_4] \neq \mathit{concr}[\mathsf{m}_5]$ and thus $x_1 \neq x_2$, which does not follow from α. Again the intruder does not find out x_1 or x_2 but only that the two users voted differently. The crucial point here (and the strength of α-β privacy) is that we do not have to specify checks for all the different things that the intruder may be able to figure out, or even think about them, but simply just specify a formula α that describes what he is cleared to know and a formula β containing all information that may be available to him.

3.3 α-β-Privacy in Transition Systems

We now show how we can extend α-β-privacy to transition systems. The key idea is that we can define an α-β *state* as the pair (α, β) of formulae and privacy as reachability in the resulting transition system. Formally, with Σ, $\Sigma_0 \subseteq \Sigma$, $\mathcal{V}$ and $\approx$ as before:

Definition 7. *An α-β* state *is a pair (α, β) of formulae where $\alpha \in \mathcal{L}_{\Sigma_0}(\mathcal{V})$ and $\beta \in \mathcal{L}_{\Sigma}(\mathcal{V})$. Let $\mathcal{S}$ denote the set of all α-β-states. An α-β* transition system *is a pair (I, R) where $I \in \mathcal{S}$ and $R \subseteq \mathcal{S} \times \mathcal{S}$. As is standard, the set of* reachable states *is the smallest set that contains I and that is closed under R, i.e.: if S is reachable and $(S, S') \in R$, then also S' is reachable. We say that an α-β-transition system* satisfies privacy *iff in every reachable state (α, β), β respects the privacy of α.*

As an example of privacy as reachability, consider a simple transition system with an initial state that has no information, and four successor states $S_{i,j}$ with $i, j \in \{0, 1\}$ depending on two independent choices i and j of the user. In all four states, we have $\alpha \equiv x \in \{0, 1\}$. Let now $\beta_{i,j} \equiv \alpha \ \wedge \ \phi_{gen}(2) \ \wedge \ \phi_{concr} \ \wedge \ \phi_{eval} \ \wedge \ \phi_{struct} \ \wedge \ \mathit{concr}[\mathsf{m}_1] = \mathsf{scrypt}(\mathsf{k}_j, \mathsf{r}_j, i) \wedge \mathit{eval}[\mathsf{m}_1] = \mathsf{scrypt}(\mathsf{k}_j, \mathsf{r}_j, x) \ \wedge \ \mathit{concr}[\mathsf{m}_2] = \mathit{eval}[\mathsf{m}_2] = \mathsf{k}_1$, where $\mathsf{k_j}$ and $\mathsf{r_j}$ are new constants. In the states with $j = 0$, the intruder cannot deduce anything interesting as he does not have the key needed for decryption, but in the states with $j = 1$ we have $\beta_{i,0} \models x = i$. Thus, there are reachable states in which the intruder can find out more than he is supposed to.

3.4 Modeling Further Example Scenarios

We chose the following three major areas to model further examples of α-β privacy, which are discussed in [14]: randomized vs. non-randomized encryption including non-determinism and the notion of strong secrecy, guessing attacks (in which we discuss different approaches to encode passwords and guessing in α-β privacy and show unique features of our logic), and privacy-friendly identity management including pooling of knowledge.

In particular, in [14], we discuss in detail an example of how to model anonymous credential systems, which highlights two interesting aspects of our approach: (i) we can have formulae α that talk also about relations between data (e.g., $y < \mathsf{1996}$ to specify that a user if at least 18 years old), and (ii) we can easily model that two dishonest agents collaborate and pool their knowledge. To that end, suppose we have individual privacy specifications α_1 and α_2 (i.e., the information that was deliberately given to the two agents individually) and their actual knowledge is β_1 and β_2, respectively, where we further assume that all free variables that occur in both α_1 and α_2 actually refer to the same values. Then, in our α-β privacy approach, we simply use logical *conjunction* and ask whether $\beta_1 \wedge \beta_2$ respects the privacy of $\alpha_1 \wedge \alpha_2$. The rationale is that two agents can always pool their actual knowledge and draw conclusions from it, i.e., we should consider $\beta_1 \wedge \beta_2$ to be available to them, and even the best credential system cannot prevent that they can derive everything that can be derived from what we gave them individually, i.e., we have to at least allow them to derive $\alpha_1 \wedge \alpha_2$.

4 Automation and the Relation to Static Equivalence

The concept of α-β-privacy is very expressive, because Herbrand logic is. Considering Example 2, we recall that we can axiomatize arithmetic (of natural numbers) by a Herbrand formula α so that $\alpha \models \gamma$ iff γ is a true sentence of arithmetic. Let *valid* be a further nullary relation symbol in Σ_0 and $\beta \equiv \alpha \wedge (\gamma \implies \mathit{valid})$; then β respects the privacy of α iff γ is a true sentence of arithmetic. Thus, in general, α-β privacy (or its complement) is not even semi-decidable.

We see this expressive power as a feature, because it allows us to think about privacy without the tight corset imposed by automated methods. In this section, we explore a decidable fragment and the relation to static equivalence of frames for which many decidability results already exist. Because of its expressive power, it is no surprise that α-β-privacy subsumes static equivalence of frames:

Theorem 1. *Let $\mathcal{F}_0$ and $\mathcal{F}_1$ be two frames, Σ_0 consist of the nullary relation symbol neq, $\alpha \equiv \mathsf{true}$ and $\beta \equiv \alpha \wedge \phi_{gen}(l) \wedge \phi_{Fr}(\mathcal{F}_0) \wedge \phi_{Fr}(\mathcal{F}_1) \wedge \phi_{concr_0} \wedge \phi_{concr_1} \wedge (\neg neq \implies (\forall x, y.\ (gen(x) \wedge gen(y)) \implies (concr_0[x] = concr_0[y] \iff concr_1[x] = concr_1[y])))$. Then, β respects the privacy of α iff $\mathcal{F}_0 \sim \mathcal{F}_1$.*

Proof. From the definition of $\sim$ in Herbrand logic it follows that *neq* is derivable from β iff the frames are not statically equivalent. If *neq* is not derivable, there is no Σ_0-formula that follows from β and not from α. □

The simple argument of this theorem may seem a bit unfair towards static equivalence of frames, since we are not truly using α for the high-level payload information available to the intruder, but rather considering everything as technical, and then just exploit the expressive power of Herbrand logic. In addition, we show in [14] that a large fragment of the static-equivalence problem for frames can be encoded into the message-analysis fragment of α-β privacy (cf. Def. 6).

Looking deeper at the two concepts, we observe the following situation. Static equivalence of frames is essentially the question whether the intruder can distinguish two concrete worlds. For instance, the frames $\mathcal{F}_0$ and $\mathcal{F}_1$ in the Examples 3 and 4 represent two concrete worlds that the intruder can distinguish: $\mathcal{F}_0 \not\sim \mathcal{F}_1$. In contrast, α-β privacy expresses with α all possible worlds (there may be more than two) and with β one concrete world, asking whether the intruder can exclude some of the worlds of α. This, in particular, requires a distinction between high-level payload information and low-level technical information that frames do not have.

4.1 Limiting the Interesting Derivations

In order to show that many α-β-privacy problems can indeed be reduced to static equivalence of frames, we need to overcome one obstacle: α-β-privacy asks for *any* Σ_0-formula α' that can be derived from β but not from α. In general, there is a (countably) infinite choice for α' to consider. Recall that we call α *combinatoric* if Σ_0 is a finite set of free constants. Then the Herbrand Universe for α is finite and so there are finitely many possible different interpretations of the free variables of α. We can use this to limit the number of α' we need to consider:

Theorem 2. *Consider an (α, β) pair where α is combinatoric and consistent. Then, there is a finite number $n > 0$ of satisfying interpretations of the free variables of α, and we can give $N = 2^n - 2$ formulae $\alpha'_1, \ldots, \alpha'_N \in \mathcal{L}_{\Sigma_0}(fv(\alpha))$ such that $\alpha \not\models \alpha'_i$ for all $i \in \{1, \ldots, N\}$ and β violates the privacy of α iff $\beta \models \alpha'_i$ for some $i \in \{1, \ldots, N\}$.*

Before we prove Theorem 2, let us recall that when α is combinatoric, then Σ_0 is a finite set of free constants, so that the Herbrand Universe for α is finite and thus there are finitely many possible different interpretations of the free variables of α. The key observation is that we can use this to limit the number of α' we need to consider. For example, if $\alpha \equiv x \in \{0, 1, 2\}$ then it obviously suffices to check the following six candidates for α':

$$\begin{array}{lll} \alpha'_1 \equiv x = 0 & \alpha'_2 \equiv x = 1 & \alpha'_3 \equiv x = 2 \\ \alpha'_4 \equiv x \in \{0, 1\} & \alpha'_5 \equiv x \in \{0, 2\} & \alpha'_6 \equiv x \in \{1, 2\} \end{array}$$

In other words, any of the proper, non-empty subsets of the original choice $\{0, 1, 2\}$ are candidates to check—the empty set is excluded because x must be one of the values, and the whole choice $\{0, 1, 2\}$ is excluded because that already follows from α. In fact, all other possible α' that one could come up with (with the same set of free variables) must be equivalent to one of the above candidates, e.g., $\alpha' \equiv x \in \{0, 1\} \implies x \notin \{0\}$ is equivalent to α'_6.

Proof (Theorem 2). The Herbrand universe for α is simply Σ_0, so every model of α must map the free variables of α to elements of Σ_0, which gives us a finite set of choices since Σ_0 is finite. In fact, this set of choices can be effectively be computed, since α can only consists of variables, constants of Σ_0, equality, Boolean connectives and quantifiers (so, basically, Quantified Boolean Logic). We can effectively write each model in the form $\gamma \equiv x_1 = \mathsf{c}_{i_1} \wedge \ldots \wedge x_k = \mathsf{c}_{i_k}$. Let $G = \{\gamma_1, \ldots, \gamma_n\}$ be the set of all possible models that satisfy α, i.e., $\alpha \equiv \gamma_1 \vee \ldots \vee \gamma_n$. Consider the set $G_P = \{G_0 \mid \emptyset \not\subset G_0 \subsetneq G\}$ of proper, non-empty subsets of G. G_P has $N = 2^n - 2$ elements $\{g_1, \ldots, g_N\}$. Define now the α'_i to be the disjunction of all formulae in g_i for each $i \in \{1, \ldots, N\}$, i.e., $\alpha'_i = \bigvee_{\phi \in g_i} g_i$. For any $i \in \{1, \ldots, N\}$, $\alpha \not\models \alpha'_i$ since one of the possible valuations of the free variables of α is not satisfied (since we chose only *proper* subsets of G; note that we could exclude the empty set as at least one valuation will true).

Suppose now that β violates the privacy of α. Then, there is a formula $\alpha' \in \mathcal{L}_{\Sigma_0}(fv(\alpha))$ such that $\beta \models \alpha'$ and $\alpha \not\models \alpha'$. From Definition 5, it follows that $fv(\alpha') \subseteq fv(\beta)$: suppose $x \in fv(\alpha') \setminus fv(\beta)$, then $\beta \models \forall x.\ \alpha'$ and still $\alpha \not\models \forall x.\ \alpha'$. Since $\alpha \not\models \alpha'$ there is a valuation γ_i of the free variables of α so that $\gamma_i \models \alpha$ but $\gamma_i \not\models \alpha'$. Also there must be some γ_j with $\gamma_j \models \alpha$ and, since $\beta \models \gamma$, also $\gamma_j \models \alpha'$. Thus, the set of models of α' is a proper, non-empty subset of the G, so some $g_i \in G_P$ describes exactly the models of α', and therefore, finally, $\alpha'_i \equiv \alpha'$. □

4.2 Reduction to Frames and Decidability

We now reduce message-analysis problems (cf. Def. 6) to finitely many static equivalence problems of frames. Note that α in a message-analysis problem is by definition combinatoric, and thus, by Theorem 2, there are finitely many satisfying interpretations of the free variables of α (and nothing else is to interpret since Σ_0 does not contain non-constant function or relation symbols). We denote these models simply as substitutions σ_i (that map from $fv(\alpha)$ to Σ_0).

Theorem 3. *Consider (α, β) in the message-analysis problem fragment of α-β privacy (i.e., according to Def. 6), with terms $t_1, \ldots, t_l$. Let $\{\sigma_1, \ldots, \sigma_n\}$ be the models of α, and define $F_i = \{\mathsf{m}_1 \mapsto \sigma_i(t_1), \ldots,\ \mathsf{m}_l \mapsto \sigma_i(t_l)\}$. Then, β respects the privacy of α iff $F_1 \sim F_2 \sim \ldots \sim F_n$.*

Proof (Theorem 3). We proceed by proving that β respects the privacy of α iff $\forall i. F_i \sim F_1$, which is equivalent as $\sim$ is an equivalence relation. Let $eq([x_1 \mapsto t_1, \ldots, x_j \mapsto t_j])$ for some j denote the formula $x_1 = t_1 \wedge \ldots \wedge x_j = t_j$. Then $\alpha \equiv \bigvee_{i=1}^{n} eq(\sigma_i)$. Consider the formula

$$\alpha' \equiv \bigvee_{\{i \mid F_i \sim F_1\}} eq(\sigma_i)$$

Let further $\alpha_i \equiv eq(\sigma_i) \vee eq(\sigma_1)$, i.e., the restriction of α to the choice between σ_1 and σ_i. It follows, for every $i \in \{1, \ldots, n\}$, that $F_i \sim F_1$ iff β respects the privacy of α_i (see [14] for a proof of this claim). Therefore, $\beta \models \alpha'$. The conjunction of α' has at least one element, since $\phi_i \sim \phi_1$ at least for $i = 1$. There are then two possible cases:

- If there is also at least one $i \in \{2, \ldots, n\}$ such that $F_i \not\sim F_1$, then $\alpha \not\models \alpha'$, and thus β violates the privacy of α.

– Otherwise (note: trivially $\alpha \models \alpha'$ in this case), by Theorem 2, there is no α' that follows from β but not from α, thus β respects the privacy of α. $\square$

Since this result is independent of the considered set Σ_{op} of cryptographic operations and algebraic theory, we immediately have that if we can decide static equivalence for a given theory (e.g., [2, 5]), then we can decide the message-analysis problem fragment of α-β privacy for that theory.

Note that, instead of relying on static equivalence, we could have also given a direct decision procedure for our example theory, without an enumeration of all models. In a nutshell, the key idea of such a proof is that in the restricted form of α and β considered in the message-analysis problem, we can find a violation of α-β privacy iff we can make use of the axiom $\mathit{concr}[s] = \mathit{concr}[t] \iff \mathit{eval}[s] = \mathit{eval}[t]$. Then, we can show that there is a violation of α-β privacy iff β has a *witness*, i.e., there are terms $s, t \in \mathcal{T}_\Sigma$ such that concr and eval are defined for s and t, and $\mathit{concr}[s] = \mathit{concr}[t]$ while $\mathit{eval}[s] \neq \mathit{eval}[t]$. Then, we can remove all analysis steps (i.e., decryptions and decompositions) from β by encoding them in additional memory positions. The resulting β' preserves the privacy of α iff β does, and has a witness iff it has one in the free algebra, for which it is straightforward to find witness or to prove their absence, and thus conclude the proof. This argument is, of course, similar to what one does to decide static equivalence in frames. However, static equivalence looks at the more basic problem to compare a pair of frames, while α-β privacy asks to look at all models of α (as did the above reduction).

5 Concluding Remarks

We have introduced α-β privacy as, we believe, a simple and declarative way to specify privacy goals: the intruder should not be able to derive any "non-technical" statement from the technical information β that he cannot derive from the intentionally released information α already. We have given a variety of examples that describe how α-β privacy can be used in practice and investigated formally its close relationship to static equivalence of frames, which allows to use existing methods for deciding a fragment of α-β privacy.

α-β privacy bears some similarities with the non-interference approach (e.g., [15]) since it also distinguishes (at least) two levels of information, usually low-level and high-variables. These are, however, fundamentally different from our payload α and technical information β since they are formulae that express relations between values (rather than directly being public or private values). We actually do not mind that the intruder gets hold of (some) technical information as long as he cannot use it to obtain anything interesting besides the payload.

There are also privacy notions building on database abstractions. The two predominant notions are the *k-anonymity* family [17], asking whether an intruder is unable to reduce the anonymity set below a threshold of k users, and *differential privacy* [9], asking whether an intruder can detect significant changes in a probability distribution on statistical data released by a curator on data sets differing in one element. For k-anonymity, we observe that the property that α has at least k

models, and that the intruder cannot deduce an α' with less choices, is encodable in α-β privacy and will be part of future work. As differential privacy is a property established on the information release function of the curator, a relation to our notion is not straightforward.

We have mentioned above and in the previous sections a few directions for future work. In addition to these, we have already started to consider further examples, to formalize a language for specifying α-β transition systems, and to generalize our decidability results to larger fragments of α-β privacy.

References

1. M. Abadi. Private authentication. In *PET*, LNCS 2482, pages 27–40. Springer, 2003.
2. M. Abadi and V. Cortier. Deciding knowledge in security protocols under (many more) equational theories. In *CSFW*, pages 62–76. IEEE CS, 2005.
3. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *POPL*, pages 104–115. ACM, 2001.
4. B. Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *CSFW*, pages 82–96. IEEE CS, 2001.
5. B. Blanchet, M. Abadi, and C. Fournet. Automated verification of selected equivalences for security protocols. *JLAP*, 75(1):3–51, 2008.
6. V. Cortier, M. Rusinowitch, and E. Zalinescu. Relating two standard notions of secrecy. *Logical Methods in Computer Science*, 3(3), 2007.
7. S. Delaune, M. D. Ryan, and B. Smyth. Automatic verification of privacy properties in the applied pi-calculus. In *IFIPTM*, pages 263–278. Springer, 2008.
8. D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198 – 208, 1983.
9. C. Dwork. Differential Privacy: A Survey of Results. In *TAMC*, LNCS 4978, pages 1–19. Springer, 2008.
10. H.-D. Ebbinghaus, J. Flum, and W. Thomas. *Mathematical logic*. Springer, 1994.
11. J. Goubault-Larrecq. Finite models for formal security proofs. *J. Comput. Secur.*, 18(6):1247–1299, 2010.
12. T. Hinrichs and M. Genesereth. Herbrand logic. Technical Report LG-2006-02, Stanford University, CA, USA, 2006. http://logic.stanford.edu/reports/LG-2006-02.pdf.
13. IBM Research – Zurich. Specification of the identity mixer cryptographic library. version 2.3.4. Technical report, IBM Research, 2012.
14. S. Mödersheim, T. Groß, and L. Viganò. Defining Privacy is Supposed to be Easy (Extended Version). Technical Report 2013-21, DTU Compute, Lyngby, Denmark, 2013.
15. P. Ryan and S. Schneider. Process algebra and non-interference. In *CSFW*. IEEE CS, 1999.
16. P. Selinger. Models for an Adversary-Centric Protocol Logic. *Electronic Notes in Theoretical Computer Science*, 55(1):69–84, 2003.
17. L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.
18. C. Weidenbach. Towards an Automatic Analysis of Security Protocols in First-Order Logic. In H. Ganzinger, editor, *CADE 16*, LNCS 1632, pages 314–328. Springer, 1999.