Unification modulo a property of the El Gamal Encryption Scheme

Serdar Erbatur[1], Santiago Escobar[2], and Paliath Narendran[1]

[1] University at Albany–SUNY (USA), {se,dran}@cs.albany.edu
[2] Universidad Politécnica de Valencia (Spain), sescobar@dsic.upv.es

1 Introduction

Equational Unification has recently been applied in the field of formal analysis of cryptographic protocols. Formal methods have been very useful in detecting nontrivial flaws in protocols and also to verify their correctness; see Meadows [7] for a survey of formal verification of cryptographic protocols. Terms in this approach are often assumed to be in the *free algebra,* i.e., the function symbols are not interpreted; in particular algebraic properties of function symbols are ignored. Thus two terms are equal only if they are syntactically equal; thus the analysis only requires standard unification. However, it is possible to extend this analysis by also considering algebraic properties of terms, to get a deeper analysis against attacks that can possibly exploit those properties [4, 5]. Therefore, E-unification algorithms provide tools to achieve this goal.

In this work, we consider an axiom which is observed in El Gamal encryption. We briefly explain how a message is encrypted within the El Gamal scheme. Let p be a prime, g a generator of Z_p and x the private key obtained from the range 1 to $p-2$. Define $h = g^x \, mod \, p$. The public key is the tuple (p, g, h). A message m is encrypted by first selecting a random integer r s.t. $gcd(r, p-1) = 1$ and then computing

$$a \equiv g^r \, mod \, p, \;\; b \equiv m * h^r \, mod \, p$$

The ciphertext of m is the pair (a, b). Let us define $B(m, r) = m * h^r$. When using this functionality, there is an important property of B that could be taken into account. This property is unfolded as follows:

$$B(m_1, r_1) * B(m_2, r_2) \; = \; m_1 * h^{r_1} * m_2 * h^{r_2}$$

and

$$B(m_1, r_1) * B(m_2, r_2) \; = \; m_1 * m_2 * h^{r_1 + r_2} \;\; (mod \, q)$$

where $h^{r_1+r_2}$ can be written (abstracted) as $r_1 \circledast r_2$. Therefore the equality of interest is

$$B(m_1, r_1) * B(m_2, r_2) \; = \; B(m_1 * m_2, \; r_1 \circledast r_2)$$

In the rest of the paper, we consider the theory with this equality, which we will denote by $\mathcal{E}$. In Section 2 we give an algorithm and prove decidability of

$\mathcal{E}$-unification. A novel feature of our approach is the use of types in detecting non-termination. We follow the standard notation in the literature, see [2] for more details.

2 Unification modulo $\mathcal{E}$

We present decidability of $\mathcal{E}$-unification by constructing an algorithm along with a proof of correctness. First, we define a set of inference rules based on standard forms and observe that those rules are complete and sound similarly to the case in [1]. This is not very different from our approach in earlier papers, but the novelty is that termination of the algorithm (i.e., inference rules) is obtained by introducing a type system for function symbols of $\mathcal{E}$. Note that $\mathcal{E}$ is not defined as a typed theory. However, using types as described later in this section allows us to identify a set of equivalence classes that does not grow. Then through a series of lemmas we show how to detect infinite splitting, which is the hardest type of failure to detect since new variables are continuously introduced into an $\mathcal{E}$-unification problem. Thus, the algorithm either transforms an $\mathcal{E}$-unification problem to dag-solved form or returns failure by finding (i) a function clash, (ii) (extended) cycle induced by relations among the variables, or (iii) variables which split indefinitely.

The function symbols B, $*$ and $\circledast$ are cancellative, i.e., if s_1, t_1, s_2, t_2 are ground terms in normal form, then $B(s_1, t_1) =_{\mathcal{E}} B(s_2, t_2)$ if and only if $s_1 =_{\mathcal{E}} s_2$ and $t_1 =_{\mathcal{E}} t_2$; similarly for the other symbols. This can be shown using the fact that $\mathcal{E}$ can be oriented either way to get a convergent rewrite system.

We now define several relations among variables:

- $U \succ_{r_*} V$ iff there is an equation $U = T * V$
- $U \succ_{l_*} V$ iff there is an equation $U = V * T$
- $U \succ_{r_\circledast} V$ iff there is an equation $U = T \circledast V$
- $U \succ_{l_\circledast} V$ iff there is an equation $U = V \circledast T$
- $U \succ_{r_B} V$ iff there is an equation $U = B(T, V)$
- $U \succ_{l_B} V$ iff there is an equation $U = B(V, T)$
- $U \succ_{\circledast} V$ iff $U \succ_{r_\circledast} V$ or $U \succ_{l_\circledast} V$
- $U \succ_{*} V$ iff $U \succ_{r_*} V$ or $U \succ_{l_*} V$
- $U \succ_{B} V$ iff $U \succ_{r_B} V$ or $U \succ_{l_B} V$

Let $\sim_{lp(*)}$ and $\sim_{lp(B)}$ be the reflexive, symmetric and transitive closures of $\succ_{l_*}$ and $\succ_{l_B}$ respectively. Also, let $\succ = \succ_{\circledast} \cup \succ_{*} \cup \succ_{B}$.

(a) *Variable Elimination*:

$$\frac{\{X =^? V\} \uplus \mathcal{EQ}}{\{X =^? V\} \cup [V/X](\mathcal{EQ})} \qquad \text{if } X \text{ occurs in } \mathcal{EQ}$$

(b) *Cancellation on B*:

$$\frac{\mathcal{EQ} \uplus \{X =^? B(V, Y),\ X =^? B(W, T)\}}{\mathcal{EQ} \cup \{X =^? B(V, Y),\ V =^? W,\ Y =^? T\}}$$

(c) *Cancellation on '$*$'*:

$$\frac{\mathcal{EQ} \ \uplus \ \{X =^? V * Y, \ X =^? W * T\}}{\mathcal{EQ} \ \cup \ \{X =^? V * Y, \ V =^? W, \ Y =^? T\}}$$

(d) *Cancellation on '$\circledast$'*:

$$\frac{\mathcal{EQ} \ \uplus \ \{X =^? V \circledast Y, \ X =^? W \circledast T\}}{\mathcal{EQ} \ \cup \ \{X =^? V \circledast Y, \ V =^? W, \ Y =^? T\}}$$

(e) *Splitting*:

$$\frac{\mathcal{EQ} \ \uplus \ \{X =^? B(V,Y), \ X =^? W * Z\}}{\mathcal{EQ} \ \cup \ \{X =^? W * Z, \ V =^? V_0 * V_1, \ Y =^? Y_0 \circledast Y_1, \ W =^? B(V_0,Y_0), \ Z =^? B(V_1,Y_1) \ \}}$$

(f) *Failure Rule 1*:

$$\frac{\mathcal{EQ} \ \uplus \ \{X =^? B(V,Y), \ X =^? W \circledast T\}}{FAIL}$$

(g) *Failure Rule 2*:

$$\frac{\mathcal{EQ} \ \uplus \ \{X =^? V * Y, \ X =^? W \circledast T\}}{FAIL}$$

(h) *Occur-Check*:

$$\frac{\mathcal{EQ}}{FAIL} \qquad \text{if } X \ \succ^+ \ X \text{ for some } X$$

A set of equations (i.e., a unification problem) is said to be in *dag-solved form* (or *d-solved form*) if and only if they can be arranged as a list

$$x_1 =^? t_1, \ \ldots, \ x_n =^? t_n$$

where (a) each left-hand side x_i is a distinct variable, and (b) $\forall\, 1 \leq i \leq j \leq n$: x_i does not occur in t_j [6].

The variable X in the splitting rule is called an *e-peak*. That is, an *e-peak* is a variable X such that $X \succ_{l_*} W$, $X \succ_{r_*} Z$, $X \succ_{l_B} V$ and $X \succ_{r_B} Y$ for some variables V, Y, W, Z. The rules (a) – (h) can be applied in any order but we propose the following strategy for efficiency in reaching the dag-solved form. Rules (a), (f), (g) and (h) have the highest priority, followed by (b), (c) and (d). Finally the rule (e) has the lowest priority.

Lemma 1. *Rules (a) – (h) are sound and complete for $\mathcal{E}$-unification.*

Proof. The result is obtained in a similar way to that of [1]. □

Apart from rules (f), (g) and (h), another failure case is infinite splitting. A necessary and sufficient condition for this, along with a failure rule, will be given later in this paper. Two example cases where rule (e) applies infinitely because of a variable shared between the first argument of B and an argument of $*$ are shown below. We underline those equations that give rise to a new e-peak in the conclusion of the inference rule.

(1) *Infinite Splitting Case 1*:

$$\frac{\mathcal{EQ} \ \uplus \ \{X =^? B(V,Y), \ X =^? V * Z\}}{\mathcal{EQ} \ \cup \ \{X =^? V * Z, \ \underline{V =^? V_0 * V_1}, \ Y =^? Y_0 \circledast Y_1, \ \underline{V =^? B(V_0,Y_0)}, \ Z =^? B(V_1,Y_1) \ \}}$$

(2) *Infinite Splitting Case 2*:

$$\frac{\mathcal{EQ} \uplus \{X =^? B(V,Y),\ X =^? W * V\}}{\mathcal{EQ} \cup \{X =^? W * V,\ \underline{V =^? V_0 * V_1},\ Y =^? Y_0 \circledast Y_1,\ W =^? B(V_0,Y_0),\ \underline{V =^? B(V_1,Y_1)}\ \}}$$

Note that if a variable is shared between the second argument of B and the first or second argument of $*$, this does not lead to infinite splitting:

(3) *Example*:

$$\frac{\mathcal{EQ} \uplus \{X =^? B(V,Y),\ X =^? W * Y\}}{\mathcal{EQ} \cup \{X =^? W * Y,\ V =^? V_0 * V_1,\ \underline{Y =^? Y_0 \circledast Y_1},\ W =^? B(V_0,Y_0),\ \underline{Y =^? B(V_1,Y_1)}\}}$$

By failure rule (f), case (3) results with a Function Clash. In contrast, note that cases (1) and (2) cause infinite splitting since they both give rise to *e-peaks* repeatedly. To explain this, we first assign the set of types $\{\alpha,\ \gamma\}$ to arguments of function symbols as follows:

$$B : \alpha \times \gamma \to \alpha,\ \ * : \alpha \times \alpha \to \alpha,\ \ \circledast : \gamma \times \gamma \to \gamma$$

This type mechanism is not strict: in fact one may consider $\{\alpha,\ \gamma\}$ as a set of attributes. Note that a term such as $B(X, X)$ would be problematic for strict typing but it is reasonable to assume that X has both α and γ as "attributes" in this case. This also explains how types are assigned to existing variables. As an example, the equation $U_1 =^? V * W$ and $U_2 =^? V \circledast W$ are typed properly in this discipline: U_1 with α, U_2 with γ and both V and W with α and γ, respectively.

We, in general, assign types to variables as follows. A variable V is assigned type α if and only if there exists a variable U such that one of $U \succ_* V, U \succ_{l_B} V$, $V \succ_* U$, or $V \succ_{l_B} U$ holds. Likewise, a variable V is assigned type γ if and only if there exists a variable U such that $U \succ_{\circledast} V$ or $U \succ_{r_B} V$ or $V \succ_{\circledast} U$ or $V \succ_{r_B} U$.

We can now observe that in (1) and (2) the new peaks have type α. However, the set $Var(\mathcal{S})$ for a problem $\mathcal{S}$ may get larger because of splitting. Also, if V is the representative of an equivalence class of variables with respect to a relation $R \in \{\sim_{lp(*)}, \sim_{lp(B)}\}$, i.e., $[V]_R$, then obviously all variables in $[V]_R$ have the same type as V.

As seen in rule (e), a variable T can split in two ways: either as $T = T_0 * T_1$ or as $T = T_0 \circledast T_1$. The splitting rule (e) may be applied further to new variables, and in general we may obtain a variable T_ω where $\omega \in \{0,1\}^*$ is a string of 0's and 1's. Therefore we adopt the general discipline for creating new variables as: $T_\omega = T_{\omega 0} * T_{\omega 1}$ **or** $T_\omega = T_{\omega 0} \circledast T_{\omega 1}$ [3]. Also, if $\omega = \lambda$, the empty string, then $T_\omega = T$, i.e., T is an original variable. For a variable V, define $\overline{V} = \{V_\omega \mid \omega \in \{0,1\}^*\}$. Note that $\overline{V}$ may be infinitely large.

Lemma 2. *Let V be of type α and $|\overline{V}|$ be infinite. Then any descendant V_ω, where $\omega \in \{0,1\}^*$ and $V \succ_*^+ V_\omega$, joins an originally existing $\sim_{lp(B)}$-equivalence class which has type α.*

Our main observations are (i) if a variable splits, then its descendants have the same type, see Lemma 2 and (ii) in case of infinite splitting the new e-peaks

[3] Using the type discipline given earlier, we assume that T is α-typed in the former case and γ-typed in the latter.

are always α-typed. We justify (ii) later in this section. Thus (i) and (ii) allow us to effectively leave γ-typed variables out.

Definition 1. *Let $\mathcal{V} = \{X_w \mid X$ is α-typed and $\omega \in \{0,1\}^*\}$ i.e., the set of variables which have type α in a given problem. In other words, $\mathcal{V}$ includes the original variables with type α and the new variables that α is assigned as type.*

Lemma 3. *Let $\mathcal{S}$ be an $\mathcal{E}$-unification problem and $X \in Var(\mathcal{S})$ such that X is an e-peak. Then X has type α, i.e., $X \in \mathcal{V}$.*

Let us consider grouping elements of $\mathcal{V}$ with respect to the equivalence relation $\sim_{lp(B)}$. That is, we write $\mathcal{V} = \biguplus [X]_{\sim_{lp(B)}}$ where $X \in \mathcal{V}$. Therefore, we have a set of $\sim_{lp(B)}$-equivalence classes such that the number of them remains the same even if splitting applies infinitely.

Lemma 4. *The number of $\sim_{lp(B)}$-equivalence classes in $\mathcal{V}$ does not increase.*

Let us define $\beta = \sim_{lp(B)} \circ \succ_* \circ \sim_{lp(B)}$. Then the following result gives us a way to detect infinite splitting.

Lemma 5. *If rule (e) applies infinitely, then there exists a β-cycle among $\sim_{lp(B)}$-equivalence classes in $\mathcal{V}$.*

We define an interpretation which gives a valid model for $\mathcal{E}$. If B is interpreted as projection to its first argument, i.e., left projection, then we get $m*n = m*n$ out of $\mathcal{E}$. This is useful since the unification problem is solvable only if its interpreted version is also solvable.

Lemma 6. *Let $\mathcal{P}$ be an $\mathcal{E}$-unification problem. If β is cyclic, then $\mathcal{P}$ is not solvable.*

Therefore we can define the following failure rule which deals with infinite splitting.

(i) *Infinite Splitting*:

$$\frac{\mathcal{EQ}}{FAIL} \qquad \text{if } \beta \text{ is cyclic in } \mathcal{V}$$

Lemma 7. *Unification modulo $\mathcal{E}$ is decidable using rules (a)-(i) above.*

3 Conclusion

We have shown decidability of unification modulo a theory with a single axiom which is a property of the El-Gamal public key cryptosystem.

In [3], we show decidability of unification for a theory with symbols B and $*$ through a similar outline of the results using the fact that the number of $\sim_{lp(B)}$-classes remain same. On the other hand, the number of $\sim_{lp(B)}$-classes here does not remain the same in general; only the number of $\sim_{lp(B)}$-classes in a special subclass, that we identified through a type system, is non-decreasing. Introducing types, which are not originally defined for the equational theory, proved useful to show termination of the algorithm. Furthermore, the version of $\mathcal{E}$ which has the same multiplication operator on the right, has an undecidable unification problem as shown recently in [1].

References

1. S. Anantharaman, S. Erbatur, C. Lynch, P. Narendran, M. Rusinowitch. "Unification modulo Synchronous Distributivity". Technical Report SUNYA-CS-12-01, Dept. of Computer Science, University at Albany—SUNY. Available at `www.cs.albany.edu/~ncstrl/treports/Data/README.html` (An abridged version to be presented at *IJCAR 2012.*)
2. F. Baader, W. Snyder. "Unification Theory". *Handbook of Automated Reasoning*, pp. 440–526, Elsevier Sc. Publishers B.V., 2001.
3. S. Erbatur, C. Lynch, P. Narendran. "Unification in Blind Signatures". Presented at FTP 2011. Available at `www.cs.albany.edu/~se/blindsig_ftp2011.pdf`
4. S. Escobar, C. Meadows, J. Meseguer. "Maude-NPA: Cryptographic Protocol Analysis Modulo Equational Properties". In: *Foundations of Security Analysis and Design V, FOSAD 2007/2008/2009 Tutorial Lectures* (A. Aldini, G. Barthe, and R. Gorrieri, eds.) LNCS 5705, pages 1–50.
5. S. Escobar, C. Meadows, D. Kapur, C. Lynch, C. Meadows, J. Meseguer, P. Narendran, R. Sasse. "Protocol analysis in Maude-NPA using unification modulo homomorphic encryption". In: *Proceedings of the 13th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming, July 20-22, 2011, Odense, Denmark* (P. Schneider-Kamp and M. Hanus, eds.), pages 65–76.
6. J.-P. Jouannaud, C. Kirchner. "Solving equations in abstract algebras: a rule-based survey of unification." *Computational Logic: Essays in Honor of Alan Robinson*, pp. 360–394, MIT Press, Boston (1991).
7. C. Meadows. "Formal Verification of Cryptographic Protocols: A Survey" *ASIACRYPT* , pp. 135–150 (1994).