Minimum Disclosure as Boolean Optimization: New Results

Alessandro Armando[1,2], Angela Contento[1], Daniele Costa[1], and Marco Maratea[1]

[1] DIST, University of Genova, Italy
armando@dist.unige.it,{angela.contento,daniele.costa}@studenti.unige.it,
marco@dist.unige.it
[2] Security & Trust Unit, FBK-irst, Trento, Italy
armando@fbk.eu

Abstract. In advanced credential-based, distributed applications, clients gain access to sensitive resources by proving possession of some properties to the server. Yet, user's properties (or the combination thereof) may be sensitive and the user may want to minimize their disclosure whenever possible. Given a quantitative estimate of the sensitivity of the properties contained in the credentials, the *Minimum Disclosure (MIN-DISCL) problem* is the problem of determining a disclosure of information that allows the user to obtain the service, while minimizing the overall sensitivity of the exposed disclosure. Previous work provided a reduction of MIN-DISCL to the Weighted Max-SAT problem and showed the practical viability of the approach through experimentation with the YICES SMT solver. In this paper we present a simplified and optimized problem formulation that leads to much smaller encodings and smaller solving times than the original formulation on randomly generated MIN-DISCL problems.

1 Introduction

The use of cryptographic credentials to regulate client-server interactions on the web has received considerable attention in the last decade. The basic idea is that in order to access a sensitive resource, a client must preliminarily prove possession of some properties to the server. This is achieved by an exchange of credentials from the client to the server. This provides a robust and flexible solution to the problem. Yet, user's properties (or the combination thereof) may be sensitive in themselves and the user may want to minimize their disclosure whenever possible. For instance, the combination of the user's family name and zip-code may enable the identification of the user and therefore their disclosure should be avoided if the access to the same resource can be obtained by other means, e.g. by disclosing the user's email address only. Since this problem must be tackled at run-time, performance of the solver is key.

The use of logic-based languages in this context originated in [5], and than other approaches, e.g. [6, 10, 14], provided the user with the ability to specify privacy preferences and ways to reason about them. This problem has been recently formalized as the

Proceedings of the 19^{th} RCRA workshop on *Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion* (RCRA 2012).
In conjunction with AI*IA 2012, Rome, Italy, June 14–16, 2012.

minimum disclosure (MIN-DISCL) problem [3, 2, 1], i.e. determining a disclosure of information that allows the user to obtain the service, while minimizing the overall sensitivity of the disclosure. In particular, [2, 1] propose a heuristic graph-based approach to tackle the problem, while in [3] an exact approach is presented where (i) information items are associated with weights representing their sensitivity; (ii) the whole problem is encoded as a logical Weighted Max-SAT problem, an extension of the propositional satisfiability (SAT) problem, in which a subset of the clauses is associated with weights and the goal is to maximize the sum of the weights of satisfied clauses, and (iii) solving is done with the YICES [7] solver, a Satisfiability Modulo Theories (SMT) solver that can also solve Max-SAT problems.

In this paper we first present a simplified and optimized problem formulation, in particular w.r.t. how disclosure limitations (i.e. that at most k out of n information items can be exposed) are expressed. Then, we show a preliminary experimental analysis on randomly generated MIN-DISCL problems using a wide range of SMT, Pseudo-Boolean (PB), and Max-SAT encodings and solvers that can express and solve the problem. The results of our experiments indicate that (i) our model can generate much smaller formulas than the original formulation, and solving times obtained with YICES are greatly reduced, and (ii) while our formulas are generally solved quickly by a wide set of solvers, the Max-SAT solver MINIMAXSAT shows best performance among the ones considered.

The paper is structured as follow. In Section 2 we present our formulation of the problem and discuss the differences w.r.t. the formulation given in [3]. In Section 3 we present and discuss the experimental results we have carried out. We conclude in Section 4 with some final remarks and a brief discussion of the work ahead.

2 Problem Formulation

The starting point of our formulation is [3] where the interested reader can find more details and examples. The information that a client can provide to acquire services from a server is collected within a *portfolio*. The portfolio contains both certified and uncertified *properties*. Certified properties are organized within *credentials*, signed by third parties, while uncertified properties are self-signed statements that form a declaration.

Credentials are organized by types, where the type of a credential identifies the properties that the credential certifies. There possibly can be a hierarchy of types (defined by a partial order relationship $\preceq$ on the credential types). A credential instance c is characterized by an identifier c, a set of certified properties $\mathit{properties}(c)$, a type $type(c)$ and an issuer. Credential can be atomic, i.e. can only be released as a whole, or non-atomic. A property p is characterized by an identifier p, a type $type(p)$, organized in a 1-level hierarchy, i.e. no multiple levels are allowed, and a value.

Let C be a set of credentials, and P a set of properties. Given a credential $c \in C$, $\mathit{properties}(c)$ is the set of properties certified by c.

A *disclosure* is defined as follows.

Definition 1. *Let C and P be the set of credentials and properties, respectively, in the client portfolio. A disclosure D of the portfolio is a set $\{d_1, \ldots, d_j\}$ of elements of the form $d = c.\{p_1, \ldots, p_i\}$ such that the following conditions hold:*

1. $\forall d \in D \Longrightarrow \{p_1, ..., p_i\} \subseteq \mathit{properties}(c)$ *(certifiability);*
2. $\forall d \in D$ *s.t. c is atomic* $\Longrightarrow \{p_1, \ldots, p_i\} = \mathit{properties}(c)$ *(atomicity).*

2.1 Modeling the client portfolio

In this subsection we represent the conditions mentioned above as Boolean formulas, plus other constraints that a user may want to express on disclosed information. We will outline relevant differences w.r.t. the formulation in [3].

Certifiability. Each disclosed property must be certified by (at least) a credential. This is modeled with the following clause: for each atomic $p \in P$

$$p \rightarrow \vee_{c \in C, p \in \mathit{properties}(c)}\ c \tag{1}$$

i.e. or the property p is not disclosed, or at least a credential has to certify p.

Atomicity. If an *atomic* credential is disclosed, all of its properties are disclosed. This is modeled with the following SAT formula: for each $c \in C$

$$c \rightarrow \wedge_{p \in \mathit{properties}(c)}\ p \tag{2}$$

i.e. either the credential is not disclosed or all its properties are disclosed.

Both (1) and (2) are slightly simplified w.r.t. [3], e.g. the correspondent formula for (1) is

$$\neg p \vee (p \wedge (\vee_{c \in C, p \in \mathit{properties}(c)}\ c))$$

and similarly for (2).

Forbidden view. The user may want to specify that some information, i.e. credentials or properties, can not be released simultaneously. Let V be such a set of forbidden views. This is modeled with the following clause: for each $v \in V$, $v \subseteq C \cup P$

$$\vee_{x \in v} \neg x \tag{3}$$

Disclosure limitation. The user may want to specify some limitations on the disclosure in term of cardinality constraints, i.e., at most n among a given number of information items can be released. Let L be such a set of disclosure limitation, each $l \in L$ having a bound of $n_l < |l|$ elements. This is modeled with the following SAT formula: for each $l \in L$ (l' is a subset of l):

$$\wedge_{l' \subseteq l, |l'| = n_l + 1}\ \vee_{x \in l'}\ \neg x \tag{4}$$

i.e. it is explicitly forbidden to release $n_l + 1$ elements (the cases with $|l'| > n_l + 1$ are implied).

This formulation is different in spirit to the one employed in [3]: consider S to be the power set of the element in l limited to have at most n_l elements, i.e.

$$S = \{s \in 2^l : |s| \leq n_l\}$$

the same constraint can expressed with

$$\vee_{s \in S}(\wedge_{x \in S}\, x \wedge \wedge_{x \notin S}\, \neg x) \tag{5}$$

i.e. by explicitly enumerating all assignments to the elements in l that satisfy the disclosure limitation.

Even if both (4) and (5) are exponential (in n_l), (4) is tighter than (5): on the practical side, it is thus clear that the resulting size of (5) can be very large, much larger than (4).

2.2 Modeling the server request

We have seen that, in order to grant a service to an user, a server asks the user to prove possession of some properties.

More formally, a server request $\mathcal{R}$ is a disjunctively intended set of simple request, where each simple request R is a conjunctively intended set of simple term, a simple term r being of the form $type.\{pt_1, \ldots, pt_m\}$ where $type$ is a credential type (denoted $type(r)$), and $\{pt_1, \ldots, pt_m\}$ is a set of properties types (denoted $properties(r)$). In the following we define the concept of *term satisfaction*

Definition 2. *Let C and P be the set of credentials and properties, respectively, in the client portfolio, and R be a request. Credential $c \in C$ satisfies a term r in R iff*

- $type(c) \preceq type(r)$*; and*
- $\forall pt \in properties(r), \exists p \in properties(c) : type(p) = pt$

Term satisfaction. Given the definition, we modeled term satisfaction with the following SAT formula, for each r in R

$$TermSAT(r) = \vee_{c \in C, type(c) \preceq\, type(r)} c \wedge (\wedge_{p \in properties(c), pt \in properties(r), type(p) = pt}\, p) \tag{6}$$ [3]

Server request. By relying on (6), we can model the server request $\mathcal{R}$ with the following SAT formula:

$$\vee_{R \in \mathcal{R}}\, \wedge_{r \in R}\; TermSAT(r) \tag{7}$$

[3] This formula can be reduced with a preliminary check: if c does not certify at least one property for each of the types specified in r, the related disjunct is not included.

Sensitivity. A quantitative estimate of the sensitivity of the information contained in a disclosure can be easily represented by a function λ that maps credential, properties and their combination to an integer.

Given a set of information items, e.g. a disclosure, its sensitivity includes the sum of the sensitivity of each credential and each property in the disclosure. Other than that, the user may want to express that the association of some information items is more (or less) sensitive than the sum of the weights of the information items in it. This is the case of *sensitivity views*, or *dependencies* that can be defined on the portfolio. As an example consider to have the properties *Address* and *Country* – assuming they are exposed together, it is clear that the two information items together do not provide more information than the *Address* (*Country* is a part of a full address). The sensitivity of dependencies is expressed e.g., by assigning a weight to the association equals to the negation of the sensitivity of the dependent property, i.e. $\lambda(\{Address, Country\}) = -\lambda(\{Country\})$. On the other hand, sensitivity views contain information items that, if exposed together, bring to a sensitivity which is higher than the sum of each individual item in the view: in this case, a positive weight is associated to the sensitivity view.

The *minimum disclosure (MIN-DISCL) problem* is defined as follows.

Definition 3. *Given a set C of credentials and a set P of properties composing the client portfolio, a set A of associations, a set V of forbidden views, a set L of disclosure limitations, a labeling function λ, and a server request R, find a minimum disclosure D of the portfolio w.r.t. R that satisfies the following requirements:*

- *D is valid w.r.t. V and L;*
- *D satisfies R;*
- *$\nexists$ a valid disclosure D' s.t. D' satisfies R and $\lambda(D') < \lambda(D)$.*

3 Implementation and Experimental Analysis

Several Boolean Optimization representations can be used to encode the MIN-DISCL problem as formalized in the previous section, e.g. Max-SAT, SMT and Pseudo-Boolean (PB), and then a wide range of Max-SAT, PB and SMT solvers with different heuristics and optimization techniques can be used to solve the problem. In [3], a Partial Weighted Max-SAT encoding is employed and than solving is done with the YICES SMT solvers.

In our analysis we have considered the following solvers: the Max-SAT solvers MINIMAXSAT ver. 1.0 [9] and WMAXSATZ [11] ver. 2.5 submitted to the 2010 Max-SAT evaluation (a first version of the solver has been presented in [4]), the PB solvers BSOLO [12] ver. 3.2 submitted to the PB 2011 Competition and PBCLASP, based on the solver for Answer Set Programming CLASP [8], and the SMT solver YICES. The benchmarks have been generated as follows: we have considered $|C|$ credentials, whose hierarchy of credential types is randomly generated, and $|P|$ properties, each property being certified by a randomly selected set of credentials. The fraction of atomic credentials has been fixed to 0.7 . The number $|V|$ and $|L|$ of forbidden views and disclosure limitations has been fixed to 1 and 2, respectively: the set of credentials and properties

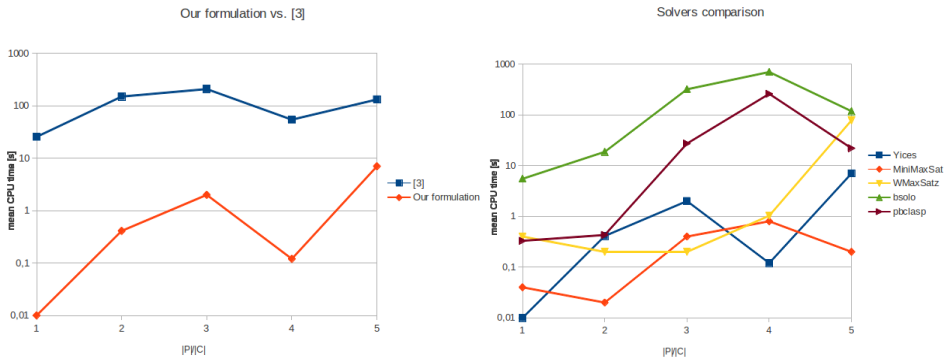

Fig. 1. Results on randomly generated MIN-DISCL problem: (Left) Comparison of our formulation and the one in [3]; (Right) Comparison of Max-SAT, SMT and PB solvers on problems encodings related to our formulation.

on which they are defined is randomly generated among all the available with a minimum of 2, as well as the bound of each disclosure. Also the server request is randomly generated by fixing the number of terms to 2, the number of simple requests is taken from $\{1, 2, 3, 4\}$ for each term, and simple terms are randomly generated.[4] Finally, we randomly assigned to each credential and property a weight w s.t. $1 \leq w \leq 20$. Moreover, we randomly generated a dependency of 2 information, whose weight is $-\lambda(a)$, where a is the information in the dependency having lower score. Finally, two sensitivity views are generated by randomly selecting from 2 to 5 information, and by assigning a weight w s.t. $1 \leq w \leq 10$ to each sensitivity view.

More in details we considered $|C| = 20$, $|P| = \{20, 40, 60, 80, 100\}$, and we randomly generated 10 instances for each configuration as we explained.

Experiments have been run on an Intel Core(TM)2 Duo CPU P8400, 2.26GHz processor. The timeout has been set to 900s and the memory limit to 2GB.

Our first experiment aims at comparing our encoding with the encoding in [3]. Figure 1 (Left) presents the results: fixing $|C| = 20$, the x-axis contains the ratio $r = |P|/|C|$ between properties and credentials, $r = \{1, 2, 3, 4, 5\}$, while the y-axis is the mean CPU time of the 10 instances for YICES on the SMT encodings related to our formulation and the formulation in [3]. As it is clear from the figure, our formulation allows YICES to solve the problems much more efficiently, by orders of magnitude. The main reason seems to be our optimized formulation of the disclosure limitations, e.g. considering the instances with $r = 5$, the mean size of the formula representing the disclosure limitations is around 835K literals for our formulation, and more than 3M literals for [3].

[4] In [3] a preliminary check is performed: a credential c, even if it represents an instance of $type(r)$, is not included in the clause if c does not certify at least one property for each of the types specified in r. In our experiments we have not done this check that, of course, can be also applied to our encoding.

The second experiment, instead, aims at comparing the performance of the SMT, PB and Max-SAT solvers mentioned above, each employing its heuristic and optimization techniques, run on related encodings of our formulation. Results are presented in Figure 1 (Right). Here we can see that, globally, the Max-SAT solver MINIMAXSAT has top performance, even if not on all values of r, followed by YICES and WMAXSATZ. PB solvers, which we remind are not restricted to SAT clauses and weights assigned to clauses, do not perform well on our experiment: in this light, the performance of YICES are remarkable, given it can deal with a much wider range of formulas than those solved by Max-SAT solvers. Still, by using appropriate and tuned solvers for our formulation, we can further improve MIN-DISCL solving by more than one order of magnitude than using YICES.

4 Conclusions and Future Work

In this paper we have presented improvements in the modeling and solving of the MIN-DISCL problem, which minimizes the quantity of information that an user is willing to expose in credential-based, client-server interactions. We have presented (i) a simplified and optimized formulation of the problem w.r.t. the original proposal; and (ii) an experimental analysis on randomly generated MIN-DISCL problems involving a wide range of Boolean Optimization encodings and solvers, i.e. Max-SAT, PB and SMT. Our results indicate that much smaller formulas can be obtained, that they can be solved considerably faster than those obtained with the original formulation, and even better performance (for some values of r) can be obtained by using MINIMAXSAT.

As part of the future work, we plan to improve our work both in the modeling and in the solving. On the modeling side, we first would like to consider polynomial encoding of the cardinality constraints that express disclosure limitations. In case the encoding still produces set of clauses, one option is to rely on [13]; otherwise, if, e.g. the PB format is employed, a cardinality constraint can be expressed with a single PB constraint. On the solving side, we plan to extend our experimental analysis, by considering other Boolean Optimization formats and solvers, e.g. 0-1 Linear Programing with CPLEX, as well as evaluate other solvers for the formats already considered.

References

1. C. Ardagna, S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, and P. Samarati. Minimising disclosure of client information in credential-based interactions. *International Journal of Information Privacy, Security and Integrity*, 1(2/3):205–233, 2012.
2. C. A. Ardagna, S. D. C. di Vimercati, S. Foresti, S. Paraboschi, and P. Samarati. Minimizing disclosure of private information in credential-based interactions: A graph-based approach. In A. K. Elmagarmid and D. Agrawal, editors, *Proc. of the IEEE 2nd International Conference on Social Computing (SocialCom 2010) and IEEE International Conference on Privacy, Security, Risk and Trust (PASSAT 2010)*, pages 743–750. IEEE Computer Society, 2010.
3. C. A. Ardagna, S. D. C. di Vimercati, S. Foresti, S. Paraboschi, and P. Samarati. Supporting privacy preferences in credential-based interactions. In E. Al-Shaer and K. B. Frikken, editors, *Proc. of the 2010 ACM Workshop on Privacy in the Electronic Society (WPES 2010)*, pages 83–92. ACM, 2010.

4. J. Argelich, C. M. Li, , and F. Manyà. An improved exact solver for partial Max-SAT. In *Proc. of the International Conference on Nonconvex Programming: Local and Global Approaches (NCP-2007)*, pages 230–231, 2007.
5. P. A. Bonatti and P. Samarati. A uniform framework for regulating service access and information release on the web. *Journal of Computer Security*, 10(3):241–272, 2002.
6. W. Chen, L. Clarke, J. F. Kurose, and D. F. Towsley. Optimizing cost-sensitive trust-negotiation protocols. In *Proc. of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, pages 1431–1442. IEEE, 2005.
7. B. Dutertre and L. de Moura. The Yices SMT solver. Technical report, Stanford Research Institute, 2006.
8. M. Gebser, B. Kaufmann, A. Neumann, and T. Schaub. Conflict-driven answer set solving. In M. M. Veloso, editor, *Proc. of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 386–392. Morgan Kaufmann Publishers, 2007.
9. F. Heras, J. Larrosa, and A. Oliveras. MiniMaxSAT: A new weighted Max-SAT solver. *Journal of Artificial Intelligence Research*, 31:1–32, 2008.
10. P. Kärger, D. Olmedilla, and W.-T. Balke. Exploiting preferences for minimal credential disclosure in policy-driven trust negotiations. In W. Jonker and M. Petkovic, editors, *5th VLDB Workshop on Secure Data Management (SDM 2008)*, volume 5159 of *Lecture Notes in Computer Science*, pages 99–118. Springer, 2008.
11. C. M. Li, F. Manyà, N. O. Mohamedou, and J. Planes. Exploiting cycle structures in Max-SAT. In O. Kullmann, editor, *Proc. of 12th International Conference on Theory and Applications of Satisfiability Testing (SAT 2009)*, volume 5584 of *Lecture Notes in Computer Science*, pages 467–480. Springer, 2009.
12. V. M. Manquinho and J. P. Marques-Silva. On using cutting planes in pseudo-Boolean optimization. *Journal on Satisfiability, Boolean Modeling and Computation*, 2:209–219, 2006.
13. C. Sinz. Towards an optimal cnf encoding of boolean cardinality constraints. In P. van Beek, editor, *Proc. of the 11th International Conference on Principles and Practice of Constraint Programming (CP 2005)*, volume 3709 of *Lecture Notes in Computer Science*, pages 827–831. Springer, 2005.
14. D. Yao, K. B. Frikken, M. J. Atallah, and R. Tamassia. Private information: To reveal or not to reveal. *ACM Transactions on Information and System Security*, 12(1), 2008.