PROPER KEY GENERATION FOR THE IZOSIGN ALGORITHM

Loránd Szöllösi, Gábor Fehér and Tamás Marosits
High Speed Networks Laboratory, Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics, Magyar Tudósok krt. 2., Budapest, Hungary

Keywords: Graph-based digital signature algorithms, graph isomorphism.

Abstract: In the last decade using digital signatures in authentication and authorization protocols just as in e-business scenarios became more and more important and indispensable. New algorithms with different features for various applications are presented continuously. The IzoSign digital signature creation algorithm was introduced by the authors of this paper at CANS 2007. At that time, random key generation was proposed, which was later found vulnerable with high probability to a vertex matching attack (Kutylowski, 2007). We hereby analyze and generalize this kind of attacks, build a key generation algorithm that withstands such attacks, and then give a (theoretic) construction for key generation which (under the $P \neq NP$ or $NP = EXP$ assumptions) is hard to break.

1 INTRODUCTION

Digital signature algorithms provide authentication of messages using mathematical methods. Signature algorithms are based on a hard problem class for which the solution can only be generated by the signer (via additional information from the key generation phase), and message hashes select a problem from this class to be solved. If a message hash is seen with the corresponding solution, it can be assumed that the person holding the additional information from the key generation phase (called private key) intended to sign the document. Most countries define the legal consequence of digital signatures the same as that of handwritten signatures (European Parliament and Council, 1999).

One possible hard problem was discrete logarithm and factoring; RSA (Rivest et al., 1977) and DSA (Schneier, 1993) are the most popular algorithms based on this problem class. Discrete logarithm is considered hard, but it is not reduced to another hard class of problems such as EXP (Aaronson, 2008), DistNP (Aaronson, 2008; Venkatesan and Levin, 1988) or NP (Aaronson, 2008; Cormen et al., 1990). These algorithms are calculation intensive, usually requiring dedicated coprocessor in embedded systems. They provide medium signature length of 1024-2048 bits. Other algorithms either try to optimize signature length (like CFS (Courtois et al., 2001)) or signature creation time (like the broken SFLASH (Courtois et al., 2003; Dubois et al., 2006)).

Our proposed algorithm, IzoSign, is based on subgraph isomorphism (Cormen et al., 1990). The hard problem is to find a subgraph in a large graph which is isomorph with an other given graph. It is a one-time signature (Schneier, 1993) protocol, that is extended to digital signatures using a modified version of the Merkle scheme (Merkle, 1989). Subgraph isomorphism is NP hard (NP complete for the decision problem) (Cormen et al., 1990), and its simple cases are are well researched (Gupta and Nishimura, 1996b; Babai et al., 1982; Miller, 1980; Luks, 1980; Filotti and Mayer, 1980), since this problem appears in the field of both model transformation and pattern recognition.

Throughout, we use the usual notation for asymptotic relations, see (Cormen et al., 1990) for example. This notation is summarized in Table 1.

Table 1: Explanation of asymptotic notations (summarized from (Cormen et al., 1990)).

Notation	Description
$f(n) = o(g(n))$	f is dominated by g
$f(n) = O(g(n))$	f is asymptotically upper bounded by g
$f(n) = \theta(g(n))$	both $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$ holds
$f(n) = \Omega(g(n))$	f is asymptotically lower bounded by g
$f(n) = \omega(g(n))$	f dominates g

Szöllösi L., Fehér G. and Marosits T. (2008).
PROPER KEY GENERATION FOR THE IZOSIGN ALGORITHM.
In *Proceedings of the International Conference on Security and Cryptography*, pages 368-372
DOI: 10.5220/0001924403680372

In Section 2 we give a short description about the IzoSign algorithm, in Section 3 we give a proof that even random key generation with large enough key graphs and constant signature length can be made secure under some assumptions. Finally in Section 6 we conclude our work.

2 THE IZOSIGN ALGORITHM

The IzoSign algorithm (described in (Szőllősi et al., 2007) is based on subgraph isomorphism. The main algorithm is a one-time signature creation method, which is then extended to a digital signature system using a modified version of Merkle trees (Merkle, 1989). The public key consists of two graphs, a needle (G) and a (larger) haystack (H), and the private key is a function $V(G) \to V(H)$ which reveals a subgraph of H that is isomorphic to G. The hardness of this protocol relies on the strength of the key graph; this can be checked during key generation. The solving time of random generated hard graphs as a function of problem size was simulated and studied by Shuichi Ichikawa and Shoji Yamamoto (Ichikawa and Yamamoto, 2002); their results show the expected exponential growth, and regression predicts that 81 vertices take 100000 years on a software-based solution, while 264 vertices provide the same defense against key-specific custom hardware attacks.

Simple cases for key generation found in the literature are (n is the number of vertices in the graph):

- k-connected partial k-trees: having an algorithm in the order of $O(n^{k+2})$ (that is, polynomial in n but exponential in k) (Dessmark et al., 1996);
- partial k-trees of bounded degree: also having an algorithm in the order of $O(n^k)$ (Gupta and Nishimura, 1996b). If $k \sim n$, this limit becomes exponential; otherwise the k-tree would have a linear number of edges, which is impossible using our graph generation.
- trees: a subproblem of the above two cases, which is easy to match; this is avoided since we generate graphs with quadratic edge count;
- k-connected partial k-paths: (Dessmark et al., 1996; Gupta and Nishimura, 1996a), a subproblem of k-connected partial k-trees with very low edge count;
- two-connected outerplanar graphs: these graphs can be matched in cubic time (Lingas, 1989), but any planarity could be detected and thus avoided using a linear-time algorithm (Hopcroft and Tarjan, 1974). Planar graphs have a linear edge count and thus can be avoided without any further tests.
- two-connected series-parallel graphs: (Lingas and Syslo, 1988), a subproblem of outerplanar graphs;
- strongly regular graphs: are easy for isomorphism (Babai, 1995; Spielman, 1996), albeit not yet shown to be polynomial for subgraph isomorphism; the criteria of strong regularity can be checked during graph generation.

2.1 Signature Generation

Signature generation consumes a key for each message as this is a one-time signature algorithm. It requires a secure hash algorithm (denoted as $h(.)$) that maps messages to an integer between 0 and $\binom{n}{k} - 1$, where $k \leq \binom{n}{k}$ specifies the required security level. Larger k means higher level but longer signatures.

1. Calculate $h(m)$ for the message m.
2. Interpret this number as one of the possible choices of k among the n vertices of G. This is a one-to-one mapping between hash values and k-vertex subgraphs of G. The chosen vertices will be denoted as $v_i, \quad 0 < i \leq k$, the set of these vertices is denoted as S.
3. Present the maps of these vertices in H. This step takes k memory lookups, and produces a subgraph of H that is isomorphic to the selected subgraph of G.

The verification algorithm then simply compared whether the isomorphism holds; this is a simple bit matrix comparison as the vertices are revealed in the same order in G and H.

The first proposal for IzoSign algorithm simply required a random unidegree graph as G, and a random extension to H. While the first is currently thought of being secure, the latter was broken by professor Miroslaw Kutylowski (Kutylowski, 2007). The break used a signature to forge other ones using the same key. It is based on the observation that a randomly chosen vertex c of G that is not adjacent to a given vertex v_i of the signature cannot be mapped to a vertex in H that is adjacent to the map of v_i in H, because of the isomorphism. In random graphs, when $4|S| = 2|G| = |H|$, the probability of this event is $\frac{1}{4}$. By inspecting all the possible v_i vertices of the signature, there's a high probability that one will find the above condition to hold for one of them for each candidate map of c, except the actual map. Therefore, by recursion it is possible (with high probability) to break a random graph given a signature. We will generalize the break shown above in Section 2.2.

2.2 Defense Against Limited Horizon Algorithms

The algorithm used to break the original, random key graph generation considered the set of vertices that are non-adjacent to a given vertex. One can generalize this scheme to consider the subset of vertices that are not accessible from a given vertex in exactly z steps. We will call these limited horizon algorithms, as they only "look into" the graph at a given depth. Direct implementation of these algorithms require $\sim n^z$ steps in dense graphs where n is the size of the graph. By taking the limit $z \rightarrow \infty$ we arrive at the general subgraph isomorphism, which is NP-hard.

To avoid this type of attack we propose to extend the graph in a well-defined manner instead of the original random extension. For the original break, this would be to have an additional vertex v_i' in H for each vertex v_i in G that is connected to the same vertices as v_i, except for one vertex. This way, the break can only eliminate a candidate map if the non-adjacency condition holds for this vertex, but not for the map candidate, which has a probability of $\frac{1}{4}$. Therefore only one quarter of the candidates can be eliminated for each vertex on average, so the attack remains exponential.

For the general case of limited horizon algorithms, we mainly use the same concept, but add additional z vertices $v_{i,\ z}$ to H for each vertex v_i in G (plus the copy of v_i). The first vertex $v_{i,\ 1}$ is added the same way as written above, but the extra vertex it needs to be connected to will be $v_{i,\ 2}$. This new vertex will have the same connections as the vertex that is not connected to $v_{i,\ 1}$, but is a neighbor of v_i (i.e., it will "simulate" the missing connection of $v_{i,\ 2}$), and so on with the other vertices up to $v_{i,\ z}$. Due to this construction, it is impossible to distinguish v_i from $v_{i,\ 1}$ using any algorithm that considers the z-step limited horizon of v_i. This extension results in a linear increase of key size in z, whereas the attack is exponential in z.

3 THEORETICALLY SECURE EXTENSIONS UNDER $P \neq NP$ AND $NP = EXP$

While in the previous section we have shown that the key generation of our algorithm can be made secure concerning a certain class of attacks, we are interested in defending it against any possible attack. We will hereby give a proof that even random key generation with large enough key graphs and constant signature length can be made secure if $P \neq NP$. We will also prove that, if the signature size is a $O(\frac{n}{\lg n\cdot\omega(1)})$ function of the key size n, and $NP = EXP$, then the algorithm is secure. The latter is a strong assumption, but no algorithm is known to disprove it. This second proof basically relies on the assumption that any algorithm capable of solving NP-hard problems is exponential. This is not the same as $P \neq NP$, since a runtime that is neither exponential nor polynomial is theoretically plausible, albeit improbable.

4 EXTENSION UNDER $P \neq NP$ ASSUMPTION

We first give a proof for the case if the assumption $P \neq NP$ is made and the signature size being a constant. An attack against the key generation of IzoSign is equivalent to an algorithm solving the informed version of the subgraph isomorphism problem. The additional input of the algorithm is k vertices in the pattern graph G and their maps in the search graph H. Without this information, subgraph isomorphism is NP-hard. Let n denote the number of vertices in G, k the vertex count of the signature (that is, the number of revealed vertices), and let $\frac{|H|}{|G|}$ be fixed to a constant c_0. Then the non-informed algorithm will have a runtime of $\omega(n^{c_1})$ for any constant c_1. The indirect assumption that a polynomial attack exists means that the informed algorithm will have a runtime of $O((n-k)^{c_2})$ if $k \geq k_0$, the latter being a constant of the informed algorithm. The maps of the k vertices can be chosen in

$$c_0n\cdot(c_0n-1)\cdot(c_0n-2)\cdot\ldots(c_0n-k+1)=\frac{(c_0n)!}{(c_0n-k)!}$$

ways, which can be bounded from above by $(c_0n)^k$. Therefore, given the informed algorithm, one could construct a non-informed algorithm that choses k distinct vertices in G, then runs $(c_0n)^k$ instances of the informed algorithm in parallel. If any of the instances finishes, we need to test the proof of the isomorphism returned, which if holds, is the solution of the non-informed problem as well. Since one of the $(c_0n)^k$ parallel instances has correct input, at least one instance will return a correct answer not slower than $O((n-k)^{c_2})$. So the runtime of the non-informed algorithm is

$$O((c_0n)^k(n-k)^{c_2}) = O(c_0^k n^k (n-k)^{c_2})$$

. We might chose $k = k_0$ as the signature size in the key generation, and a considerably large n, which leads to the runtime

$$O(c_0^{k_0} n^{k_0}(n-k)^{c_2}) = O(n^{k_0+c_2}) < \omega(n^{c_1})$$

, so we arrive at the contradiction that a polynomial algorithm exists for NP.

5 EXTENSION UNDER $NP = EXP$ ASSUMPTION

For the much stronger assumption $NP = EXP$, the runtime of the non-informed algorithm can be written as $\theta(c_3^n)$ - where c_3 is a constant - and a constant signature size is no longer needed. Instead, one might set the signature size as any such function of the key graph size n that is contained in $\mathcal{O}(\frac{n}{\lg n \cdot \omega(1)})$. This sublinear function will be denoted as $\sigma(n)$. We construct the same parallel algorithm as before, which will have a runtime

$$\begin{aligned}\mathcal{O}(c_0^k n^k (n-k)^{c_2}) &= \mathcal{O}(c_0^{\sigma(n)} n^{\sigma(n)} (n-\sigma(n))^{c_2}) \\ &= \mathcal{O}(n^{\sigma(n)}) = \mathcal{O}(n^{\frac{n}{\lg n \cdot o(1)}}) \\ &= \mathcal{O}((e^n)^{\frac{1}{o(1)}}) = o(c_4^n)\end{aligned}$$

for any c_4. Should we let $c_3 = c_4$, we arrive at the contradiction $o(c_3^n) = \theta(c_3^n)$.

This proof leads us to two conclusions: first, even the random key generation can be made secure if the key graph is large compared to the signature size, since the informed problem cannot be solved $2^{\{bits\ of\ information\}}$-times faster than the non-informed variant; and second, our originally proposed $4|S| = 2|G| = |H|$ ratio was wrong, and $|S| \ll |G|$ is necessary.

6 SUMMARY

In our paper we have shortly described the IzoSign signature algorithm and the original key generation algorithm that was broken by professor Miroslaw Kutylowski (Kutylowski, 2007). We then presented a new key generation method that is defended against the type of attack used to break the same procedure in the first version of the protocol; and given a proof of the security of the random key generation for proper parameter choices under the assumptions $P \neq NP$ or $NP = EXP$. We believe that our new key generation is safe, however, we encourage breaking attempts in order to fully understand the underlying Subgraph Isomorphism problem.

REFERENCES

Aaronson, S. (2008). Complexity zoo. http://qwiki.caltech.edu/wiki/Complexity_Zoo.

Babai, L. (1995). Automorphism groups, isomorphism reconstruction. In Graham, R., Grötschel, M., and Lovász, L., editors, *Handbook of Combinatorics*, chapter 27, pages 1447–1540. Elsevier Science.

Babai, L., Grigoryev, D. Y., and Mount, D. M. (1982). Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, pages 310–324. ACM.

Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990). *Introduction to Algorithms*. MIT Press, Cambridge, MA, USA.

Courtois, N. T., Finiasz, M., and Sendrier, N. (2001). How to achieve a McEliece-based digital signature scheme. In *Advances in Cryptology – ASIACRYPT 2001*, pages 157–174. Springer.

Courtois, N. T., Goubin, L., and Patarin, J. (2003). SFLASHv3, a fast asymmetric signature scheme. Cryptology ePrint Archive, Report 2003/211. http://eprint.iacr.org/.

Dessmark, A., Lingas, A., and Proskurowski, A. (1996). Faster algorithms for subgraph isomorphism of k-connected partial k-trees. In *European Symposium on Algorithms*, pages 501–513. Springer.

Dubois, V., Fouque, P.-A., Shamir, A., and Stern, J. (2006). Breaking SFLASH. http://www.ecrypt.eu.org-/webnews/webnews1206.htm#sflash.

European Parliament and Council (1999). Directive 1999/93/ec on a community framework for electronic signatures. http://europa.eu.int-/ISPO/legal/en/ecommerc/digsig.html, http://www.legi-internet.ro/diresignature.htm.

Filotti, I. S. and Mayer, J. N. (1980). A polynomial time algorithm for determining isomorphism of graphs of fixed genus. In *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*, pages 236–243. ACM.

Gupta, A. and Nishimura, N. (1996a). Characterizing the complexity of subgraph isomorphism for graphs of bounded path-width. In *STACS '96: Proceedings of the 13th Annual Symposium on Theoretical Aspects of Computer Science*, pages 453–464. Springer-Verlag.

Gupta, A. and Nishimura, N. (1996b). The complexity of subgraph isomorphism for classes of partial k-trees. *Theoretical Computer Science*, 164:287–298.

Hopcroft, J. and Tarjan, R. (1974). Efficient planarity testing. *Journal of the ACM*, 21(4):549–568.

Ichikawa, S. and Yamamoto, S. (2002). Data dependent circuit for subgraph isomorphism problem. In *Proceedings of 12th International Conference on Field Programmable Logic and Applications*, pages 1068–1071. Springer-Verlag.

Kutylowski, M. (2007). personal communication regarding IzoSign algorithm.

Lingas, A. (1989). Subgraph isomorphism for biconnected outerplanar graphs in cubic time. *Theoretical Computer Science*, 63(3):295–302.

Lingas, A. and Syslo, M. M. (1988). A polynomial-time algorithm for subgraph isomorphism of two-connected series-parallel graphs. In *ICALP '88: Proceedings of the 15th International Colloquium on Automata, Languages and Programming*, pages 394–409. Springer-Verlag.

Luks, E. M. (1980). Isomorphism of graphs of bounded valence can be tested in polynomial time. In *Proceedings of 21st IEEE FOCS Symposium*, pages 42–49.

Merkle, R. C. (1989). A certified digital signature. In *Proceedings on Advances in Cryptology*, pages 218–238. Springer-Verlag.

Miller, G. (1980). Isomorphism testing for graphs of bounded genus. In *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*, pages 225–235. ACM.

Rivest, R. L., Shamir, A., and Adelman, L. M. (1977). A method for obtaining digital signatures and public-key cryptosystems. Technical Report MIT/LCS/TM-82.

Schneier, B. (1993). *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* John Wiley & Sons, Inc., New York, NY, USA.

Spielman, D. A. (1996). Faster isomorphism testing of strongly regular graphs. In *STOC '96: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 576–584. ACM Press.

Szőllősi, L., Marosits, T., Fehér, G., and Recski, A. (2007). Fast digital signature algorithm based on subgraph isomorphism. In *LNCS 4856: Proceedings of the 6th International Conference on Cryptology and Network Security*, pages 34–46. Springer.

Venkatesan, R. and Levin, L. (1988). Random instances of a graph coloring problem are hard. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of computing*, pages 217–222. ACM Press.