A HEURISTIC STATE SPACE SEARCH MODEL FOR SECURITY PROTOCOL VERIFICATION

Zeeshan Furqan, Ratan Guha, Shahabuddin Muhammad
Schoool of Computer Science,Universityof Central Florida, Orlando, Florida, USA

Keywords: Security Protocols, Verification, State Space Explosion, Strand Space Model, Logic of Authentication

Abstract: The vulnerability and importance of computers, robots, internet etc, demand the employment of exceedingly reliable security protocols. E-Business can not be encouraged with susceptible underlying security protocols. We present a heuristic state space search model for automatic security protocol verification. Our model exploits its knowledge of the search space and intelligently enhances the efficiency of security protocol verification process. It uses the representation of security protocols in terms of Strand Space Model (SSM) and logic of authentication. The attributes of security protocol are first represented in SSM and then interpreted into logic. This logical module is coded in the form of states. Our model accepts these states as its input and attempts to verify them. An efficient algorithm is used for the verification procedure. The goal is to avoid state space explosion problem and improve the overall efficiency by exploring maximum number of states in a given amount of time. The simplicity of our approach enables it to be translated into existing solutions for greater efficiency.

1 INTRODUCTION

Increased reliance on computers for data storage, retrieval, and communication has elevated the need for reliable security protocols. A security protocol is a sequence of messages between two or more parties in which encryption is used to provide authentication or to distribute cryptographic keys for new conversations (Roger Needham et al., 1978). A security protocol ensures the legitimate and desired working of the encapsulating object, e.g. a computer, a wireless network, a robot, e-software etc. Serious efforts have been deployed in crafting security protocols as they provide the foundation for a secure system. History has proven security protocols to be vulnerable to attacks even after circumspect design and meticulous review by the experts. These vulnerabilities have motivated the researchers to design formal methods for security protocol verification.

Many researchers have tried to formally verify the correctness of security protocols. The first work was done in this area by (Danny Dolev et al., 1983) and by (Danny Dolev et al., 1982). Most of the later work on formal analysis of cryptographic protocols is based on the Dolev-Yao model or some variant of it (Catherine Meadows, 2003). Trace-based model (TBM) was successfully applied as automatic security protocol analysis approach. However, the TBM based approaches suffer severely from the state space explosion problem as the number of states and transitions grow exponentially with the number of participants involved in the protocol (Doron Peled, 1993) (Doron Peled, 1994). Partial order and symmetry reductions techniques have been utilized to reduce the search space. These techniques, however, experience redundant state checking (Dawn Song et al., 2001). Theorem proving has been utilized but it requires more expertise and human interaction. Isabelle (Lawrence Paulson, 1997) is an example of theorem prover. It has the disadvantage of generating direct counterexamples like most of the theorem proving based techniques. NRL protocol analyzer (Catherine Meadows, 1994) is also based on theorem proving and has the advantage of proving a protocol correct for arbitrary number of participants. However, it often requires non-trivial amount of human interaction and expertise, and the running time could be much slower than in model checking approach (Catherine Meadows, 1996). Tools (Darrel kindred et al., 1996) (Stephen Brackin, 1997) (Gawin Lowe, 1995), based on the logics of knowledge and belief such as BAN logic (Martin Burrows et al., 1989) and GNY logic (Li Gong et al., 1990), have also been exploited to discover flaws in security protocols.

Furqan Z., Guha R. and Muhammad S. (2004).
A HEURISTIC STATE SPACE SEARCH MODEL FOR SECURITY PROTOCOL VERIFICATION.
In *Proceedings of the First International Conference on E-Business and Telecommunication Networks*, pages 113-118
DOI: 10.5220/0001394101130118

(Martin Burrows et al., 1989) presented logic of authentication to formally express security protocol analysis. (Javier Fabrega et al., 1999) developed a notion of strand space model (SSM) and applied the strand space formalism to prove the correctness of the Needham-Shroeder-Lowe protocol. Our technique is an extension to their work. We propose a heuristic state space search model for security protocol verification using strand space model and logic of authentication. The notion of bundle is used to represent any instance of protocol execution. The bundles are then represented in states. A state represents the collection of properties of bundles. We apply our search strategies to verify protocol correctness after representing protocol executions as states. The algorithm terminates if protocol execution is verified. Otherwise, it intelligently explores the search tree to verify correctness. A failure in finding correctness suggests that the state needs further exploration. Experiments show that our model avoids trapping into state space explosion problem and improves the overall efficiency. It can successfully be embedded in most of the SSM based approaches for an efficient state space reduction. We have designed this model to improve the overall security of the encapsulating system (e.g. e-commerce, internet, etc.) which is of primary importance in winning the trust of clients. We have focused only on authentication and secrecy of security protocols. Encompassing complete security features will be a part of our future work.

The paper is organized as follows. We first present basics of strand space in section 2. Section 3 deals with the explanation of SSM and its incorporation in our model. Section 4 describes our algorithm, experimental results, and advantages of strategies based on our model. Conclusion is followed in section 5.

2 STRAND SPACE MODEL (SSM)

(Javier Fabrega et al., 1998) proposed Strand Space Model (SSM) to prove certain security properties manually, for example, authentication and secrecy. We first briefly mention some of the basic terms used in SSM. A detailed explanation of these terms can be found in (Javier Fabrega et al., 1999) (Dawn Song et al., 2001).

The set of actions 'Act' that principals can take during the execution of a protocol include external actions such as send (denoted by +) and receive (denoted by -), and user-defined internal actions such as debit, credit, etc. An event is a pair <action, a>, where action $\in$ Act, and a $\in$ A is the argument of the action from the set of terms A. A strand is a sequence of events that a participant may engage in. For a legitimate participant, each strand is a sequence of message sends and receives; it represents the action of that party in a particular run of the protocol. A collection of strands for various legitimate protocol parties with penetrator-strands defines strand space. A strand space over A is a set Σ together with the trace mapping tr: $\Sigma \rightarrow (\pm A)^*$.

A bundle consists of a number of strands hooked together where one strand sends a message and another strand receives the same message. In other words, a bundle is a portion of a strand space large enough to represent a full protocol exchange. A node is a pair <s, i> with $s \in \Sigma$ and i an integer satisfying $1 \leq i \leq$ length (tr(s)). The set of nodes is denoted by N. Each node belongs to a unique strand. Term $(n)_i = (tr(s))_i$, i.e, the ith signed term in the trace of s. There is an edge $n_1 \rightarrow n_2$ if and only if term(n_1) = +a and term(n_2) = -a for some $a \in A$. When n_1 = <s, i> and n_2 = <s, i+1> are members of N, there is an edge $n_1 \Rightarrow n_2$.

3 SSM APPLICATION

We present the example of Needham-Shroeder-Lowe (NSL) protocol used in strand space model as in (Gawin Lowe, 1996) (Stephen Brackin, 1996). The NSL is preferred to provide a better idea of SSM application. NSL protocol is defined as follows:

1) $A \rightarrow B$: $\{ N_a . A \}K_B$
2) $B \rightarrow A$: $\{ N_a . N_b . B \}K_A$
3) $A \rightarrow B$: $\{ N_b \}K_B$

NSL defines initiator and responder roles which are represented in strand space model as follows:

Resp[A,B,N_a,N_b]	Init[A,B,N_a,N_b]
1)<−{N_a.A}K_B>	1)<+{N_a.A}K_B>
⇩	⇩
2)<+{N_a.N_b.B}K_A>	2)<−{N_a.N_b.B}K_A>
⇩	⇩
3)<−{N_b}K_B>	3)<+{N_b}K_B>

Dawn Song, Sergey Berezin, and Adrian Perrig designed a logic based on SSM that can formally express various security properties including authentication, secrecy, and properties related to electronic commerce. They developed automatic procedure for evaluating well formed formulae in this logic (Dawn Song et al., 2001). Protocol

authenticity can be verified by using security properties defined by (Gawin Lowe, 1996) (Gawin Lowe, 1997) (Gawin Lowe, 1999). Strand spaces prove his agreement properties. A protocol guarantees a participant B (say, as the responder) agreement for certain data items x if:

Each time a participant B completes a run of the protocol as responder using x, apparently with A, then there is a unique run of the protocol with the principal A as initiator using x, apparently with B.

A weaker non-injective agreement does not ensure uniqueness, but requires only:

Each time a participant B completes a run of the protocol as responder using x, apparently with A, then there exists a run of the protocol with the principal A as initiator using x, apparently with B.

The non-injective agreement property can be specified using the logic as follows:

$$\forall C.\mathrm{Resp}(x) \in C \Longrightarrow \mathrm{Init}(x) \in C$$

Secrecy property can be defined as:

$$\neg \exists C.(\mathrm{Resp}(x) \in C \wedge \mathrm{node}(+v) \in C)$$

(Adrian perrig et al., 2000) (Dawn Song, 1999) (Dawn Song et al., 2000) introduced notion of a *goal* to represent the received terms, and a notion of 'goal binding' to represent that a goal term is first sent by a node. A 'state' is a tuple of 'semi-bundle', 'unbounded-goals', relation for 'goal-bindings (->)', and reflexive and transitive closure of '->' and '=>'. They proposed the evaluation of two simple well formed formulas as:

$$\forall C.s1 \in C \Longrightarrow s2 \in C, \text{ and}$$
$$\exists C.s1 \in C$$

where s1 is a strand constant and s2 is a regular strand constant.

We build an efficient state space search approach for security protocol verification using the logic of authentication and strategy for protocol verification. Any instance of protocol execution is represented as a bundle. Bundles are represented in states, that is, state represents the collection of properties of bundles. The initial state, say l_0, specifies that any bundle it represents must satisfy the protocol P and contain all of its sets of strands. Once the protocol execution is represented in terms of states, we apply our search strategies to verify the protocol correctness. If the protocol execution is verified, our algorithm terminates and returns true. Otherwise, it intelligently explores the search tree to verify the correctness. A failure in finding the protocol correctness results in a false return value. False value suggests that the protocol needs further exploration. process.

4 HEURISTIC MODEL

We describe the working of our model in section 4.1 and analyze its experimental results, advantages, and disadvantages in section 4.2.

4.1 Algorithm

```
Initialize Δ; //Δ is a structure
While (there are states to explore
      in the structure)
{
    δ ();
    Ω ();
    Ξ ();
}
δ()
{
  evaluate = Evaluate (Δ);
  If evaluate == TRUE
    Ξ ();
}
Ω()  // this function expands the
     // optimized state
{
  ω= Minimum_Node (ξ);
  Expand (ω_state);
  Replace (ω);
}

Ξ()  // the job of this function is
     // to set the priority of best
     // candidate state for
     // evaluation
{
   i = 0;
   λ = i;
   μ = i+1;
   While (Δ != empty)
   {
      If ζλ = ζμ
         If  lλ < lμ
            Set_Priority (Δ[i]);
         Else
            Set_Priority (Δ[i+1]);
      Else if sλ=lμ //where λ ≠ μ
         If lμ-lλ > φ AND |ζμ- ζλ| < Φ
            Set_Priority (Δ[i]);
```

```
    Else
  Priority to state with
   minimum ζ.
```

setting the priority, we start the process again by calling δ function.

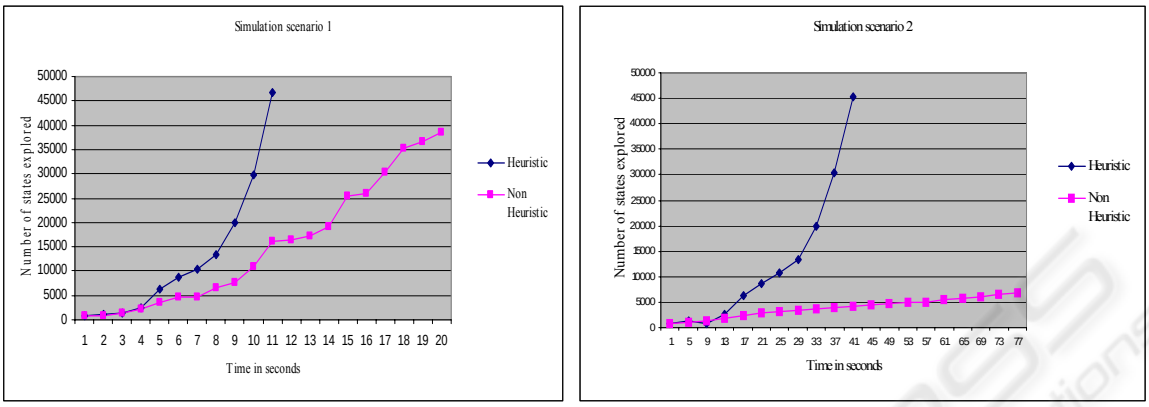

Figure 1: Search Space Explosion.

```
Else
          Priority to state with
          minimum ζ.
       i = i+1;
       λ = indexof(Δ with priority)
       μ = i+1;
    }
 }
```

We begin our search with the initial state p_o and store this state in Δ. Δ holds the values, p_i (state id), s_i (no. of strands possessed by the current state), l_i (no. of child states of p_i), ξ_i (pronounced as small xi) = $\xi_{i-1} + \zeta_i$ (ξ serves as a bound in the search space). ζ_i (pronounced as zeta) = $s_i l_i$, is used to compute the priority of individual states. Initially Δ = { p_o, s_o, l_o, $\xi_o = \zeta_o = s_o l_o$ }.

Search process starts with the initial state. We call δ function for our initial state. δ function takes our structure Δ and evaluates the current state. It returns true if current state is verified or rejected. That is, either s2 belongs to the state or unbounded goals of that state are empty (see section 3 for details). If δ returns false, it means the current state needs further expansion. Ω function is then called for further expansion. Ω selects the state with minimum ξ and expands it. The astute use of ξ makes sure that we do not go in one direction in the search space. Ω function modifies Δ array entries for the state with minimum ξ.

After Ω, we enter into Ξ (pronounced as big xi) function. The job of Ξ is to set the priority of the best candidate state. A node having fewer numbers of children and/or fewer numbers of strands is given preference. Heuristics are used to decide the trade offs between strands and children of a node. We used Φ and φ as thresholds where Φ>>φ. After

4.2 Experimental Results

We conducted extensive experiments to appraise the efficiency of our model. A java based implementation of the proposed model was simulated on Windows XP with Pentium-4 processor and 256MB RAM. The graphs in figure 1 depict the comparison between simulations of automatic protocol verification using our model and without using our model. A significant difference between the two strategies can be seen as the number of states grow exponentially with time and heuristic based approach tends to end sooner with greater number of states explored. It is clear from the graphs that non-heuristic strategies are prone to state space explosion (which is more obvious in scenario 1 of figure 1) and explore far lesser number of states in a given amount of time (which is more obvious in scenario 2 of figure 1). Figure 2 is based on results of extensive simulations. It is clear from state space explosion in figure 2 that approximately 90% of the time state space explosion problem occurred without using any state space reduction technique. A comparison between Heuristic and Non-Heuristic in figure 2 shows that our model was able to reduce the problem approximately 95% of the time. We define the probability that state space explosion will occur and Non-Heuristic strategies will suffer from the problem as P(Non-Heuristic). We call the probability that state space explosion will occur and our proposed model will suffer as P(Heuristic). The probabilities calculated corresponding to our simulations are:

P(Non-Heuristic) = (0.90) * (0.95) = (0.85)
= approximately 85%

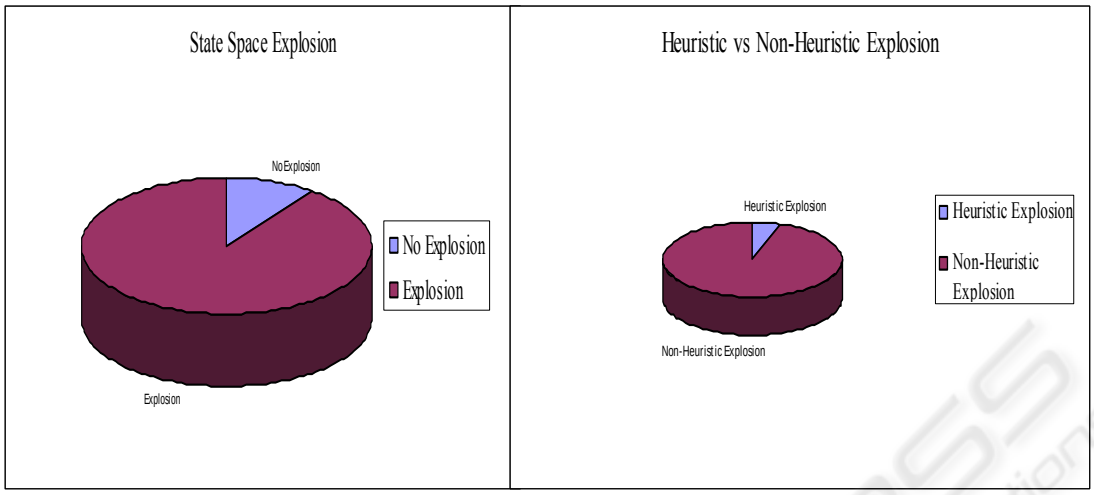

Figure 2: Comparison of Heuristic and Non Heuristic Models

P(Heuristic) =(0.90)*(0.05)=(0.05)
= approximately 5%

These results are based on our simulations. However, the excessive number of simulation is a fair measure to get an idea of practicability of our approach. The failure rate of 5% with our model, as compared to 85% without any model, is an encouraging, although not the best, achievement.

Our algorithm dramatically reduces the search space. It increases the number of protocols verified in a given amount of time and avoids falling into traps by intelligently bounding the search procedure. The approach employed to represent the protocol is simple and tries to minimize human interaction and expertise. It can easily be incorporated in most of the existing SSM approaches to avoid state space explosion problem. The simplicity of this algorithm allows its idea to be translated in strategies other than SSM based approaches. We posit that the authenticity of the beneficiary system, i.e. the system using our model for its protocol verification (e.g. a wireless network, a web site etc.), will be less vulnerable to security threats. We have confined ourselves to secrecy and authentication aspects of security protocols. We are focussed to extend our work to include other security features.

5 CONCLUSION

Authentication of Security Protocols is a big question in security community. Experience has shown that well designed security protocols can also be proven false in their later stages. The limitation of manual verification has urged the security community to find ways for automatic verification techniques for security protocols. We have presented a heuristic based model for reducing search space in automatic security protocol verification. Our approach is based on the strand space model and logic of authentication. The presented model facilitates minimum human interaction and guarantees efficiency. Our model intelligently reduces the workload on existing software for automatic security protocol verification. Experimental results support the employment of heuristic based strategies. The existing solutions can also benefit by employing our proposed logic to enhance their efficiency. At present, we limit ourselves to few security aspects. However, we plan to expand the research to include other security aspects in the future.

ACKNOWLEDGEMENTS

This work was partially supported by NSF under Grant EIA 0086251 and ARO under grant DAAD19-01-1-0502. The views and conclusions herein are those of the authors and do not represent the official policies of the funding agencies or the University of Central Florida.

REFERENCES

Needham, R., and Schroeder, M., (1978) Using encryption for authentication in large networks of computers,

Communications of the ACM, vol. 2, issue 1, 2, 993-999.

Lowe, G., (1996) Breaking and fixing the Needham-Schroeder public-key protocol using FDR, In Tools and Algorithms for the Construction and Analysis of Systems, vol. 1055 of Lecture Notes in Computer Science, 147-166, Springer-Verlag.

Burrows, M., Abadi, M., and Needham, R., (1989) A logic of authentication. Technical Report 39, DEC Systems Research Center.

Lowe, G., (1997) A hierarchy of authentication specifications, *In proceedings of 1997 IEEE Computer Society Symposium on Research in Security and Privacy*, 31-43, Rockport, MA, USA.

Gong, L., Needham, R., and Yahlalom, R., (1990) Reason about belief in cryptographic protocols, *In Proceedings of the IEEE Symposium on Research in Security and Privacy*, 234-248, Oakland, CA, USA.

Fabrega, F., Herzog, J., and Guttman, J., (1998) Strand Spaces: Why is a security protocol correct?, *In Proceedings of 1998 IEEE Symposium on Security and Privacy*, 160-171, Oakland, CA, USA.

Fabrega, F., Herzog, J., and Guttman, J., (1999) Strand Spaces: Proving security protocols correct, *Journal of Computer Security*, vol. 7, issue 2, 3, 191-230.

Song, D., Berezin, S., and Perrig, A., (2001) Athena: a novel approach to efficient automatic security protocol analysis, *Journal of Computer Security*, vol. 9, issue 1, 2, 47-74.

Perrig, A., and Song, D., (2000) An initial approach to automatic generation of security protocols, *In proceedings of NDSS'00 (Network and Distributed System Security Symposium)*, San Diego, CA, USA.

Song, D., (1999) Athena: An automatic checker for security protocol analysis, *In proceedings of the 12th computer science foundation workshop*, Mordano, Italy.

Song, D., and Perrig, A., (2000) Looking for a diamond in the dessert-extending automatic protocol generation to three party authentication and key distribution protocols, *In proceedings of IEEE Computer Security Foundation Workshop (CSFW'2000)*, Cambridge, England.

Lowe, G., (1999) Towards a completeness result for model checking security protocols, *Journal of Computer Security*, vol. 7, issue 2, 3, 89-146.

Kindred, D., and Wing, J., (1996) Fast, automatic checking of security protocols, *In USENIX 2nd workshopon Electric Commerce*, 41-52, Oakland, CA, USA.

Peled, D., (1993) All from one, one for all: on model checking using representatives, In Costas Courcoubetics, editor, *proceedings of the Fifth workshop on Computer Aided Verification*, vol. 697 of Lecture Notes in Computer Science, 409-423, Elounda, Greece, Springer-Verlag.

Peled, D., (1994) Combining partial order reductions with on-the-fly model-checking, In David L. Dill, editor, *Proceedings of the sixthworkshop on computer aided verification*, vol. 818 of Lecture Notes in Computer Science, 377-390, Stanford, CA, USA, Springer-Verlag.

Brackin, S., (1996) Automatic formal analyses of cryptographic protocols, *In Proceedings of the 19th National Conference on Inofrmation Systems Security*, 40-51, Baltimore, MD, USA.

Brackin, S., (1997) Automatic formal analyses of two large commercial protocols, *In Proceedings of the DIMACS workshop on design and formal verification of security protocols*.

Lowe, G., (1995) An attack on the Needham-Schroeder public-key authentication protocol, *Information Processing Letters*, vol. 56, issue 3, 131-136.

Meadows, C., (1994) A model of computation for the NRL protocol analyzer, *In Proceedings of the 1994 Computer Security Foundation Workshop*, 84-89, Frankonia, NH, USA.

Paulson, L., (1997) Proving properties of security protocols by induction, *In Proceedings of the 1997 IEEE Computer Society Symposium on Research in Security and Privacy*, 79-83, Rockport, MA, USA.

Meadows, C., (1996) Analyzing the Needham-Schroeder public key protocol: A comparison of two approaches, *In Proceedings of the 4th European Symposium on Research in Computer Society ESORICS*, 351-364, Rome, Italy.

Dolev, D., Yao, A., (1983) On the security of public key protocols, *IEEE Transactions on Information Theory*, vol. 29, issue 2, 198-208.

Dolev, D., Even, S., Karp, R. (1982) On the security of ping-pong protocols, *Information and Control,* 57-68.

Meadows, C., (2003) Formal methods for cryptographic protocol analysis: Emerging issues and trends, *IEEE Journal on Selected Areas in Communication*, vol. 21, issue 1, 44-54.