A Unified Threat Model for Assessing Threat in Web Applications

Xiaohong Li and Ke He
School of Computer Science and Technology, Tianjin University, Tianjin, 300072, China
{xiaohongli,kehe}@tju.edu.cn

Abstract

This paper presents a unified threat model for assessing threat in web applications. We extend the threat tree model with more semantic and context information about threat to form the new model which is used to analyze and evaluate threat in the software design stage. We utilize historical statistical information contained in this model to design threat mitigation schemes. The threat assessing results and mitigation schemes can be used to direct secure coding and testing. This makes it possible to design threat-resistant web applications by means of detecting and mitigating threat in the early software design stage.

1. Introduction

In recent years, web applications have become tremendously popular. Plenty of new techniques have brought in much benefit, but they also increase the complexity of web applications at the same time. As the complexity increases, security issues increases as well. For example, the percentage of web-based attacks rose from 25% of the total number of entries in 2000 to 61% in 2006 according to an analysis of the CVE vulnerability database [1]. Most of these security issues are caused by the design level vulnerabilities, but little has been done to solve them [2]. Some approaches [3] [4] have been presented to assess web applications but they are not related to the threat in the software design stage.

This paper presents a unified threat model to assess threat in web applications. Different from other models, the model proposed here can help to analyze and evaluate threat in web applications from attackers' perspective in the software design stage. We extend the threat tree model [5] by adding more semantic and context attribute to it. As a result, software threat information, software deployment and software application environment information are all included in the new model. And we also add a threat evaluating algorithm set to it. The threat evaluating process focuses on the factors that attackers considered in their attack steps, such as attack cost, probability of successful attack, attack damage and whether to use special equipment or not. The evaluating metrics used in this model are dynamic, and they can be changed or combined for different evaluating purpose. The assessing results and mitigation schemes generated by the model can be used to direct secure coding and testing.

The rest of this paper is organized as follows. In section 2, we present the new unified threat model. We discuss related work in section 3 and conclude the paper in section 4.

2. Unified threat model

2.1. Model definition

Definition 1: A unified threat model is a 3-tuple , where is a unified threat tree, is a set of threat evaluating algorithms, is a set of historical statistical information. Let denotes a unified threat model.

Definition 2: A unified threat tree is a 2-tuple , it is a tree model, where is a set of nodes, is a set of relations. Let denotes a unified threat tree.

Definition 3: is a set of nodes in unified threat tree, , where is a node, let , , denotes the attributes of node .

Definition 4: is a location attribute, , where denotes root node attribute, denotes leaf node attribute, denotes intermediate node attribute. Let denotes the root node, denotes a set of leaf nodes, and denotes a set of intermediate nodes.

In definition 4, root node is a final threat goal, and can be decomposed into several sub-goals. Leaf node is a concrete threat goal and it can't be decomposed. Intermediate node between the root node and leaf nodes is an intermediate threat goal which is a sub-goal of its super-goal. Intermediate node also can be decomposed into several sub-goals.

Definition 5: is a set of semantic and context attribute of nodes, . Let denotes preconditions of a threat goal, denotes postconditions of a threat goal, denotes software deployment information, denotes software application environment information, denotes two sources of threat, , where denotes insider threat, denotes outsider threat, denotes attack scenario. are semantic attributes while is a context attribute.

In definition 5, the preconditions include assumptions that we make about the attacker or the state of the software that are necessary for an attack to succeed, such as the skills, resources, access, or knowledge that the attacker must possess, and the level of risk that the attacker must be willing to bear. The postconditions include knowledge acquired by the attacker and changes to the software state that result from successfully implementing the attack steps when the preconditions hold, such as system paralysis, system performs malicious function and to be controlled by attacker. Software deployment information contains security information in deployment, such as secure target in deployment and methods of managing secure function, etc. software application environment information contains information of environment in which the software may be applied, such as description information of software runtime parameters, runtime status and security state, etc. software deployment information and application environment information are used for researching on the relationship among software threat, software deployment and software application environment, and they can be used to direct secure deploying and applying. Insider threat includes misuse, privilege abuse and secret disclosure behavior of insider, etc. Outsider threat includes all kinds of attack from outsider, such as DoS attack, DDoS attack. Attack scenario is the situation of carrying out an attack. Security experts can obtain a lot of attack scenario information from attack case and historical statistical information.

Definition 6: is a type attribute, , where denotes a AND node attribute, denotes OR node attribute. Let denotes a set of AND nodes, denotes a set of OR nodes.

In definition 6, AND node represents a type of node that only when you achieve all the threat goals of its sub-nodes then you can achieve the threat goal of the AND node. OR node represents a type of node that only when you achieve any threat goal of its sub-nodes then you can achieve the threat goal of the OR node.

Definition 7: is a set of relations between nodes, , relation is a 6-tuple . Let and respectively denote the super node and sub node. denotes the special equipment required versus no special equipment, . denotes attack cost, . denotes the probability of successful carrying out an attack, . denotes attack damage. is directly proportional to and inversely proportional to , .

Definition 8: Let denotes a set of attack path, denotes th attack path. An attack path is a minimum cut set of . A node set is a cut set, if: (1) it is a leaf node set of ; and (2) if all the threat goals of these leaf nodes are achieved, the final threat goal of the root node in can be achieved. Let denotes a cut set. A node set is a minimum cut set, if: (1) it is cut set; and (2) if any leaf node is removed from the cut set, the cut set is not a cut set anymore [6][7]. Let denotes a minimum cut set.

Definition 9: is a set of threat evaluating algorithms, , denotes the model constructing algorithm, denotes the attack path hunting algorithm, denotes the unified threat model based threat evaluating algorithm.

Definition 10: is a set of historical statistical information which can be used to help design the mitigation schemes.

2.2. Threat evaluating algorithm

```
Algorithm 1: model constructing algorithm
Input: node set $N=\{N_i \mid L, At, K\}$, relation set $R=\{R_{ij} \mid i, j \in N\}$.
Output: unified threat model $Tr$.
1. $Tr \leftarrow \varnothing$
2. While $N \neq \varnothing$ Do
 2.1. get a node $N_i$
 2.2. If $N_i \notin N^l$ Then
         $Tr \leftarrow N_i ; Tr \leftarrow R_{ij}, j \in N$ ;
 2.3. Else $Tr \leftarrow N_i$;
3. End While
4. Return $Tr$.
Algorithm 2: attack path hunting algorithm
Input: unified threat model $Tr$.
Output: attack path set $Pa$.
1. $Pa \leftarrow \varnothing$
2. For all the nodes of $Tr$ Do get node $N_j$ inverted
 2.1. If $N_j \in N^l$ Then
         $Pa_j \leftarrow N_j$ ;( $Pa_j$ denotes attack path set where $N_j$ is the root note)
  2.2. Else If $N_j \in N^{AND}$ Then
   2.2.1. $n \leftarrow \prod_{i=1}^{k} n_i$ ;( $k$ is the number of AND node $N_j$ 's sub-nodes, $n_i$ is the number
          of sub-node $i$ 's attack path)
   2.2.2. For $l$ from $0$ to $n-1$
```

```
            $M \leftarrow \left\lfloor l \div \prod_{r=i+1}^{k} n_r \right\rfloor \bmod n_i$ ;( $i = 1,\ldots,k-1$ ,
            when $i = k$ , $M \leftarrow \lfloor l \rfloor \bmod n_k$ , $l = 0,\ldots,n-1$ )
            $Pa_j[l] \leftarrow \bigcup_{i=1}^{k} Pa_i[M]$ ;( $i = 1,\ldots,k$ , $l = 0,\ldots,n-1$ , $Pa_i[M]$ is $M$ th attack path of
            $N_j$ 's $i$ th sub-node)
      2.2.3. End For
      2.2.4. $Pa_j \leftarrow \{Pa_j[0],\ldots,Pa_j[n-1]\}$
    2.3. Else // $N_j \in N^{OR}$
            $n \leftarrow \sum_{i=1}^{k} n_i$ ;( $k$ is the number of OR node $N_j$ 's sub-nodes, $n_i$ is the number of
            $i$ th sub-node, $P_i[1],\ldots,P_i[n_i]$ are the attack paths of $N_j$ 's all the sub-nodes)
          $Pa_j \leftarrow \{P_1[1],\ldots,P_1[n_1],\ldots,P_k[1],\ldots,P_k[n_k]\}$ ;
  3. End for
  4. Return $Pa$ .
```

Algorithm 3: unified threat model based threat evaluating algorithm

Input: attack path set Pa , evaluating value assignment $Eq, Co, P, Da \in R_{ji}, i \in N^l$ of leaf nodes.

Output: threat evaluating result set $Eq', Co', P', Da' \in R_{kj}, k = N^r$, mitigation scheme set K' .

```
  1. For every attack path $Pa[M]$ in $Pa$ ( $M = 1,\ldots n, n$ is the number of attack path in $Pa$ )
         Do
    1.1. For every $N_i$ in $Pa[M]$ ( $i = 1,\ldots t, t$ is the number of leaf nodes in $Pa[M]$ ) Do
           $Eq' \leftarrow \bigvee_i Eq; (Eq \in R_{ji}, i \in Pa[M])$
           $Co' \leftarrow \sum_i Co; (Co \in R_{ji}, i \in Pa[M])$
           $P' \leftarrow \prod_i P; (P \in R_{ji}, i \in Pa[M])$
           $Da' \leftarrow P'/Co'$;
           design mitigation scheme $K$ for leaf node using historical statistical
           information $H$ ;
            $K' \leftarrow K$;
    1.2. End For
  2. End For
  3. rank the attack path in descending order according to the value of $Da'$ .
  4. Return $Eq', Co', P', Da', K'$ .
```

Figure 1 shows a unified threat model example.

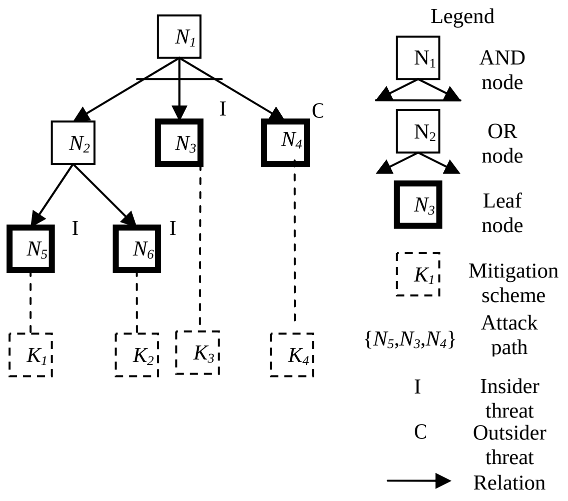

Figure 1. A unified threat model

3. Related work

Model based risk assessment methods are widely used in risk assessment of information systems. Many different models have been proposed. Fault tree [8] was presented in 1960s for the purpose of effectively analyzing the Minuteman missile system. The approach analyzed possible factors of hardware, software and environment that contribute to system fault. Attack tree [9,10,11] is an information risk assessment model for information systems. Threat tree is a software risk modeling and assessing method. These model based methods are all applied to the finished product of information systems, but seldom used to evaluate software risk in the process of software development lifecycle. CORAS project [12] presents a risk assessment framework based on the UML modeling techniques in the process of software development lifecycle. All the models above lack enough semantic information of threat and don't contain context information on carrying out attack. Also these risk assessment approaches are little used to direct secure coding and testing. In contrast, our model and threat assessment method is used in the software design stage to detect the design-level vulnerabilities and to design the mitigation schemes for secure coding and testing.

4. Conclusions

The unified threat model and model based assessment approach for web applications presented in this paper have 3 advantages: (1) More semantic and context information are provided in this model which can be used to analyze and evaluate threat precisely; (2) The historical statistics information contained in this model together with the dynamic evaluating metric can be used to generate mitigation schemes; (3) The assessing results and mitigation schemes generated in this model can be used to direct secure coding and testing.

5. Acknowledgements

This work was supported in part by the National Science Foundation of China (No.90718023), and by the National High-Tech Research and Development Plan of China (No.2007AA01Z130).

6. References

[1]Common Vulnerabilities and Exposures. http://www.cve.mitre.org/, 2006.

[2] J. D. Meier, "Web application security engineering." Security & Privacy Magazine, IEEE, vol. 4, pp. 16-24, 2006

[3] Chen, H., Wagner, D. "MOPS: an Infrastructure for Examining Security Properties of Software." In: ACM conference on computer and communication security. Washington, D.C., Nov 2002.

[4] Evans D., Larochelle, D. "Improving Security Using Extensible Lightweight Static Analysis." In: IEEE Software, Jan 2002.

[5] Amoroso, E. G. Fundamentals of Computer Security Technology. Englewood Cliffs. NJ: Prentice Hall PTR, 1994.

[6] Vaurio, J. K. "Treatment of general dependencies in system fault-tree and risk analysis." IEEE Transactions on Reliability. vol. 51, 278-287, 2002.

[7] Sinnamon R. M., Andrews, J. D. "Fault tree analysis and binary decision diagrams," In: Reliability and Maintainability Symposium, 1996 Proceedings. International Symposium on Product Quality and Integrity. Annual, 1996. 215-222.

[8] Richard J. R. "Analysis Techniques for Mechanical Reliability." Reliability Analysis Center, A DoD Information Analysis Center, 1985.

[9] Salter, C., Saydjari, O. S., Schneier, B., Wallner, J. "Toward a Secure System Engineering Methodology," In: Proceeding Of New Security Paradigms Workshop. September 1998.

[10] Schneier, B. "Attack Tree", Secrets and Lies. New York: John Wiley and Sons, 2000.

[11] Tidwell T., Larson R., Fitch K. et al. "Modeling Internet Attacks." IEEE, 2001.

[12] Aagedal, J. Ø., den Braber, F., Dimitrakos, T. et a1. "Model-based risk assessment to improve enterprise security." In: Proc. EDOC2002. 2002 IEEE Computer Society. 2002, 51~62.

Authors

Xiaohong Li is an associate professor of School of Computer Science and Technology at Tianjin University. Her research interests are in the areas of Security software engineering,trustworthy software and software agents.

Ke He is a Ph. D. student of School of Computer Science and Technology at Tianjin University. His research interests are in the areas of Security software engineering, trustworthy software and software security.