Producing and Evaluating Crowdsourced Computer Security Attack Trees

Dan Bogaard[1], Sanjay Goel[2], Shreshth Kandari[1], Daryl Johnson[3]
George Markowsky[4] , Bill Stackpole[3]
[1]Department of Information Sciences and Technology, Rochester Institute of Technology, Rochester NY
[2]Information Technology Department, SUNY, Albany NY
[3]Department of Computing Security, Rochester Institute of Technology, Rochester NY
[4]School of Computing and Information Science, University of Maine, Orono ME

***Abstract*—We describe the recent developments of an open-source project called RATCHET that can be used by groups of users to collectively construct attack trees. We present the RATCHET framework as well as a model for testing and evaluation of the produced attack trees. RATCHET has been tested in classroom settings with positive results and this paper presents the plans for expanding its outreach to the community at large and building attack trees through crowdsourcing. This paper gives an overview of RATCHET and an introduction to its use.**

***Index Terms*—crowdsourcing, attack tree, security, attack surface, evaluation**

I. Introduction

Attack trees and their analysis, as described by Bruce Schneier in the book *Secrets and Lies* [1], is the process of modeling possible attacks on systems. Attack trees encourage users to understand possible threats against systems, visualize those threats and assign various metrics to determine which threats are most likely to occur. Fault tree analysis (FTA), similar to attack tree analysis, has been used since the early 1960s to perform safety and reliability evaluations in high-hazard industries including originally the U.S. Air Force Ballistic Systems Division [2], [3]. While attack tree analysis and fault tree analysis are used primarily in information technology and industrial engineering respectively, the two methods have much in common. Fault tree analysis is more closely tied to risk assessment and relies more on statistical techniques. The dividing line between attack trees and fault trees is not very precise.

One of the major concerns in building an attack tree is overlooking a vulnerability in a system. Crowdsourcing the development of attack trees will help harness the contributions of experts who can bring a rich background of experience, perspective, and expertise to this problem. Having a group work together increases the likelihood that, not only, the most likely attack scenarios will be explored, but that a more complete attack surface will be developed.

Attack trees are useful only if they are sufficiently comprehensive to include most, if not all, credible threats to a computer system. These may include vulnerabilities for software and services, multi-step attacks, social engineering, physical security, network device security, etc. The number of attack vectors is too large for a typical organization to understand all of them. Crowdsourcing can address this problem by distributing the effort required to build a complete tree and making it easier to access the required expertise. Collaboratively built attack trees can be more complete and accurate, increasing potential benefits to the computing security community.

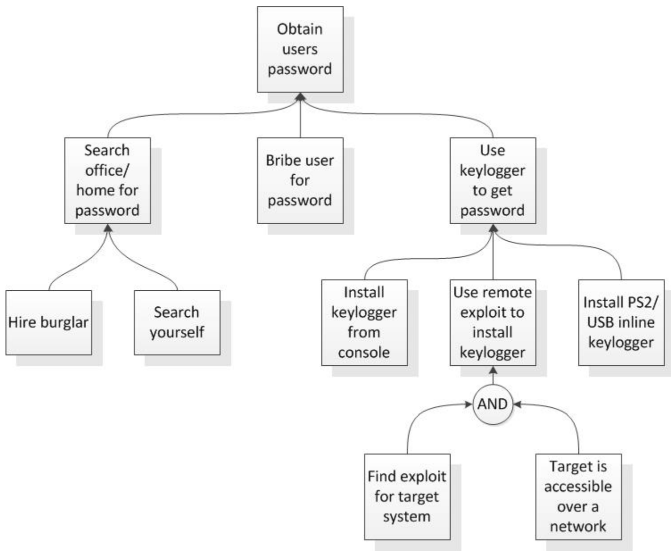

Obtain users password
Search office/ home for password
Bribe user for password
Use keylogger to get password
Hire burglar
Search yourself
Install keylogger from console
Use remote exploit to install keylogger
Install PS2/ USB inline keylogger
AND
Find exploit for target system
Target is accessible over a network

Fig. 1. Simple Attack Tree for Compromising Passwords

Over the past several years a team of faculty and students have implemented a web-based system, RATCHET [4], to allow an online community of users to create attack trees viewable to the general public. The online community has the ability to promote better ideas, voting down the ones perceived as less valuable. RATCHET (http://ratchet.csec.rit.edu/) permits people to visualize attack trees, share branches, and vote on relevant information and update the descriptions of the nodes of the attack tree.

II. Related Work

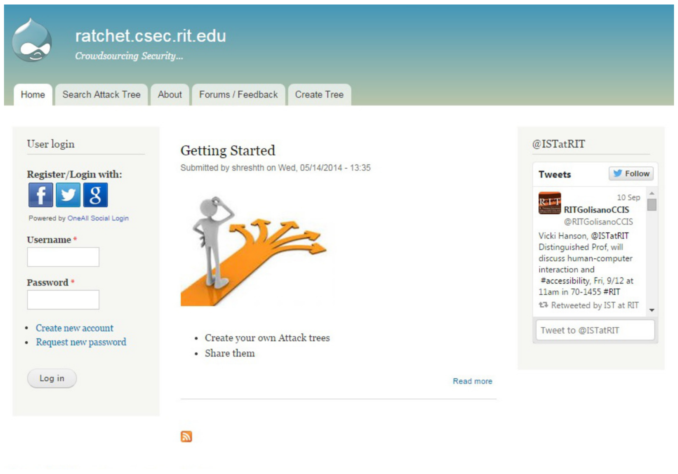
Figure 1 shows a simple attack tree. References [5]–[7] illustrate the advantages of using attack tree analysis and fault tree analysis to model security threats. Zhang et al. [8] show how to supplement fault tree analysis models by adding privilege escalation metrics into the models. Edge et al. [9] introduce the idea of expanding the attack tree concept into a protection tree. They first create an attack tree, calculate the appropriate metrics, and then they create a protection tree to help planners allocate resources and add controls to defend against specific attacks. Roy et al. [10] take the idea of protection trees a step further, proposing attack countermeasure trees where qualitative metrics, defense mechanisms, and probabilistic analysis can be applied to nodes directly within the tree. Bauer [11] discusses scenarios that can be turned into attack trees. The open-source project called Seamonster [12] produces attack trees. Seamonster is designed for individual use and does not provide a crowd sourcing capability at this time.

III. Evaluation of Attack Trees and Ratchet

While a lot of security professionals portray themselves as information security experts, true experts are few and far between. With the complexity of information systems increasing staying abreast of constantly emerging new attacks is getting harder. Crowd sourcing is one way to obtain input from security experts to increase the collective knowledge of emerging threats. This research is specifically focused on creating a model of crowd sourcing for collection and validation of threat information. Specifically, we use crowd sourcing to generate accurate attack trees for different scenarios and to make them publicly available both for research and use by professionals in analyzing information security risks. There are two fundamental problems in generating and refining attack trees using crowd sourcing: 1) how to determine the level of expertise of any contributor and 2) how to reconcile conflicting contributions. We address both these two problems below through a rating system that we have developed.

The rating system that we have adopted is based on the convergence of opinion of experts on different elements of the attack tree. To participate in this crowdsourcing endeavor, contributors register to be a part of the team and then work on specific problems. Each participant gets points based on his/her performance; the more people that converge on a solution, the more points the participants will get. Based on the consensus on any element of the tree a decision is automatically made by our algorithm for a specific node. We thus create a rational system of sorting through multiple alternate solutions for the problem.

To validate this approach, we plan to conduct an experiment where we pose multiple threat scenarios and allow users to build attack trees by adding nodes collectively. The population will consist of users with different levels of expertise. We will then attribute points to the users based on their contribution to the problem. We will statistically examine whether the ratings based on their performance matches with the expertise that they exhibit. As an initial pilot, we will run this experiment in a classroom setting where students will be graded based on their performance on an exam that requires creation and analysis of attack trees along with other information security problems.



Getting Started

Submitted by shreshth on Wed, 05/14/2014 - 13:35

- Create your own Attack trees
- Share them

10 Sep RITGolisanoCCIS @RITGolisanoCCIS

Vicki Hanson, @ISTatRIT Distinguished Prof, will discuss human-computer interaction and #accessibility, Fri, 9/12 at 11am in 70-1455 #RIT

Retweeted by IST at RIT

Fig. 2. RATCHET Homepage

Measuring accuracy of attack trees requires specialized security knowledge and recruiting security specialists is not easy. We will recruit security specialists from AMT and use an assessment instrument to determine whether or not they have deep expertise in security as described in [13]. We also have access to security specialists through various associations such as ISACA, New York State Information Security Officers group, and hacking competitions. While visual appeal is one of the attributes we want to work on, our primary focus will be on the accuracy of the attack trees.

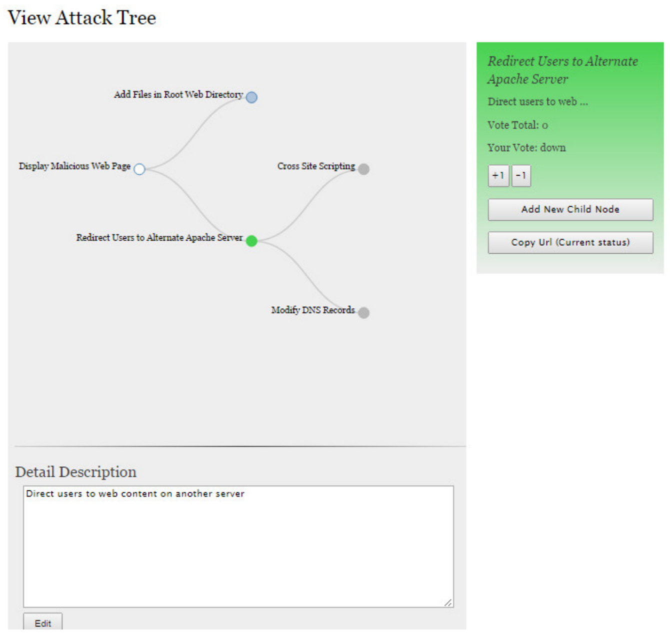
Once we have a validated system in place for measuring the expertise, we will be able to make a decision engine when deciding what changes to accept in the decision tree in case of conflict among contributors. As a further test, we will run several cases of development of an attack tree based on different scenarios and compare them with ideal trees that are developed by experts to evaluate the quality of the attack trees produced. Our metrics would include, incorporation of all the nodes, extraneous nodes, and logical placement of the nodes in the tree.

IV. RATCHET

The RATCHET web application is a system that leverages crowdsourcing for the development of attack trees. The ubiquity of the web allows for a convenient way for anyone with access to a computer to elect to view, visit or participate in the sharing of the data. Figure 2 shows the home page of RATCHET.

The researchers and students involved in this project all collaborated in gathering ideas for what the system should accomplish, how it might be used and what issues it might face. The application was designed for a collaborative audience of users who would have sufficient knowledge and the ability to help the system grow. The system can handle both simple attack trees, e.g., a password attack, and complex attack trees, e.g., attacking an operating system.

The system can allow registered contributors to build a new tree, node by node, or extend an existing tree. Every


View Attack Tree
Add Files in Root Web Directory
Display Malicious Web Page
Cross Site Scripting
Redirect Users to Alternate Apache Server
Modify DNS Records
+1
-1
Add New Child Node
Copy Url (Current status)
Detail Description
Edit

Redirect Users to Alternate Apache Server
Direct users to web ...
Vote Total: 0
Your Vote: down

Direct users to web content on another server

Fig. 3. Simple Attack Tree in RATCHET

node in the tree has an area for comments and the capability of allowing users to vote in favor or against it. Each node and comment area is editable by the creator of the node and the administrators of the system to ensure their validity. Unregistered users will be allowed to interact with the system and view existing trees. Some users will be allowed to add a vote (up/down) to nodes to indicate their view of the relevancy or usefulness of that node.

Each attack tree is displayed visually using a combination of Scalable Vector Graphics (SVG - an HTML5 technology that natively allows for dynamically drawing within a browser) and D3.js (a web standards compliant JavaScript library that allows for data driven manipulation of documents). The collapsible tree layout gives users the ability to zoom, pan or select any of the created nodes, view the comments, vote on a node's relevance, or add a new child node if they have the appropriate authorization.

V. Classroom Feedback

A. Previous Classroom Testing

In the 2014 spring semester, Professors Stackpole and Johnson tested RATCHET in a class titled "Penetration Testing Methodologies." The goal of this class was to provide students with a realistic experience with tools, techniques, and goals that face a typical penetration tester or ethical hacker. Students were introduced to the offensive side with items such as Open Source Intelligence (OSINT), scanning, penetration testing framework and collaboration tools, stepping stones, and password cracking. On the defensive side, items such as firewalls, IDS/IPS, antivirus, software updates, and the like were introduced. The need for effective and complete planning and documentation was stressed.

Attack trees were presented as a tool for organizing a penetration testing exercise. The concept of attack trees was raised as the topic for an initial discussion with students. A thirty-minute training session on how to use RATCHET was presented. Students were broken into four teams with six students per team. Each team was asked to prepare two attack trees using RATCHET: one attack tree focused on compromising an operating system, and the other focused on the exploitation of an application. Each team was instructed to perform a pre-event survey capturing their preconceived notions and opinions on the use and value of attack trees.

Each team was tasked with reviewing the attack trees created by two other teams, adding content where appropriate. The student teams were tasked with using their attack trees in performing an attack on a specific target in the Security Lab. Finally, the students were asked to provide feedback on the experience of designing their tree specific to the RATCHET implementation through another survey instrument. This exercise spanned approximately three weeks of the course.

From this exercise, several changes were made to the RATCHET system mainly revolving around the interface design. The ability to change the focus of the tree and how to indicate the node on the screen that commands or instructions apply to was enhanced. It was noticed that students did not immediately build or create the attack tree in RATCHET. Rather they built the attack tree first on a whiteboard and then transposed it to RATCHET.

Students ran into problems with creating trees, adding new nodes, and voting. While these issues did not stop the exercise, they did limit the amount of work that the students could complete. Based on the feedback modifications were made to RATCHET.

In fall 2014, Professor Markowsky taught a cybersecurity course at the University of Maine that included a section on attack trees. Students were required to create attack trees using three different methods: drawing attack trees manually, drawing trees using Seamonster [12] and drawing attack trees using RATCHET. Informal feedback was solicited. As might be expected, the manual approach was the easiest approach to start with. It was clear that there were advantages to generating to generating attack trees on the computer since they generally looked better and were easier to share. Seamonster was easy to use but did not support crowdsourcing. Some people thought RATCHET was more complicated to use than Seamonster, but just about everyone agreed that RATCHET produced the best-looking trees. Many students did not like the fact that they were unable to delete nodes once they were created.

In the original design of RATCHET, it was decided that users could not delete nodes and that voting would be the way nodes were promoted or demoted. We did not want a user to be able to delete a tree or node that many other people had worked on. It was clear that in creating attack trees, there would be occasional false steps that users wanted to remove from the tree. When they were unable to do so they became frustrated. We plan to explore ways to enable users to delete nodes in a controlled fashion through consensus building and other situations. This might take the form of enabling deletion just of nodes created by the user within a single editing session or allowing users to delete nodes that they created but which have not been used by anyone working on the attack tree.

In spring 2016 courses at both RIT and SUNY Albany will test RATCHET to see how well it works as a software package and how well cybersecurity expertise translates into quality of attack trees and nodes. The testing will use the previous testing as a starting point and seek to develop a richer evaluation framework.

VI. Conclusions and Future Work

Compared to other established fields of study such as engineering and chemistry, computing security is still in its infancy. As at the beginning of those other fields, methods, language, and measurements had to be discovered, developed, and generally accepted. The ability to record, exchange, and compare the security state of an environment or situation is necessary for the field to progress beyond an art. We believe that attack trees can be used to document, describe, and measure complex, multivariable, and situationally sensitive environments such as the ones found in computing security.

We hope that the impact of attack tree software on the field on computing security will be similar to the impact of electronic spreadsheet programs such as Lotus 1-2-3 or VisiCalc [14]. Just as spreadsheets permit business people to do easily “What if?” scenarios, attack trees can provide cybersecurity people the opportunity to experiment with “What if?” scenarios. Examples of the sort of questions people could ask are: “What if I replaced one firewall product with another?”, “What if the Exchange mail server is replaced by the Exim mail server?”, “What if remote management is enabled on a particular networking device?” In general, these and other scenarios could be explored quickly and inexpensively.

RATCHET has already been used to a moderate extent, and many valuable lessons have been learned through classroom testing from the earlier work [4]. In this work, we have done additional testing and taken the first steps to validate the results outside the classroom. The research and development efforts so far have demonstrated that attack trees can be constructed through the work of a community. The ability of attack trees to record and communicate the attack surface of an operating system, a service, and a situation has been demonstrated. It has also been demonstrated that attack trees can be shared and assembled to build large and complete attack trees. We look forward to continuing the development of RATCHET, and we welcome feedback from the IT community.

References

[1] Bruce Schneier, *Secrets and lies: digital security in a networked world: with new information about post-9/11 security*. Indianapolis, Ind.: Wiley, 2004, https://www.schneier.com/books/secrets_and_lies/.

[2] Clifton A. Ericson II, “Fault Tree Analysis - A History” *Proceedings of the 17th International System Safety Conference*, 1999. https://www.relken.com/sites/default/files/Seminal%20Documents/ericson-fta-history.pdf.

[3] Michael Stamatelatos, William Vesely, Joanne Dugan, Joseph Fragola, Joseph Minarick, and Jan Railsback, *Fault Tree Handbook with Aerospace Applications*, NASA, August 2002, http://www.hq.nasa.gov/office/codeq/doctree/fthb.pdf.

[4] Matthew Tentilucci, Nick Roberts, Shreshth Kandari, Daryl Johnson, Dan Bogaard, Bill Stackpole, and Georg Markowsky, “Crowdsourcing Computer Security Attack Trees”, *Proceedings of the 10th Annual Symposium on Information Assurance* (ASIA15), Albany, NY, June 2-3, 2015, pp. 19-23. http://www.albany.edu/iasymposium/proceedings/2015/10-Tentilucci&Roberts&Kandari&Johnson&Bogard&Stackpole&Markowsky_Ratchet.pdf.

[5] Phillip J. Brooke and Richard F. Paige, “Fault trees for security system design and analysis,” *Computers and Security*, vol. 22, no. 3, pp. 256-264, 2003.

[6] Jide B. Odubiyi and Casey W. OBrien, “Information security attack tree modeling,” *Practical and Experimental Approaches to Information Security Education: Proceedings of the Seventh Workshop on Education in Computer Security (WECS7)*, Naval Postgraduate School, Monterey, CA, January 4-6, 2006, pp. 29-37, http://www.cisr.us/events/downloads/WECS7/wecs7_proceedings.pdf.

[7] Jennifer L. Bayuk, *Stepping Through the InfoSec Program*, 1st edition. Rolling Meadows, IL: Isaca, 2007.

[8] Tao Zhang, Mingzeng Hu, Xiaochun Yun, and Yongzheng Zhang, “Computer vulnerability evaluation using fault tree analysis,” *Information Security Practice and Experience*, Springer, 2005, pp. 302-313.

[9] Kenneth S. Edge, George C. Dalton, Richard A. Raines, and Robert F. Mills, “Using attack and protection trees to analyze threats and defenses to homeland security,” *Military Communications Conference*, 2006. MILCOM 2006. IEEE, 2006, pp. 1-7.

[10] Arpan Roy, Dong S. Kim, and Kishor S. Trivedi, “Cyber security analysis using attack countermeasure trees,” *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research*, 2010, article no. 28, pp. 28-1-28.4.

[11] Michael D. Bauer, *Linux Server Security*, OReilly Media, Inc., 2005.

[12] Seamonster - Security Modeling Software, Sourceforge Project, http://sourceforge.net/projects/seamonster/.

[13] Justin Scott Giboney, Jeffrey Gainer Proudfoot, Snajay Goel, and Joseph S. Valacich, “The Security Expertise Assessment Measure (SEAM): Developing a Scale for Hacker Expertise,” to appear in *Computers and Security*.

[14] Peter Cunningham and Friedrich Fröschl, Electronic Business Revolution: Opportunities and Challenges in the 21st Century. Springer Science & Business Media, 1999.