Towards Making SELinux Smart

Leveraging SELinux to Protect End Nodes in a Federated Environment

L. Markowsky
School of Computing and Information Science, University of Maine, Orono, ME USA

Abstract – *This paper describes an intelligent, active, real-time, risk adaptable access control (RAdAC) system designed to extend the benefits of the National Security Agency's Security-Enhanced Linux (NSA's SELinux) by using SELinux not only as a secure base, but also as a source of input features to a Support Vector Machine (SVM) that will classify events/attacks in several categories. By enhancing SELinux with intelligence, it is hoped that the design will lead to real-time, non-signature based defensive systems capable of detecting and taking action against hostile users in the earliest stages of an attack.*

Keywords: support vector machine, machine learning, risk adaptable access control, RAdAC, SELinux

1 Introduction

The transformative vision of the Department of Defense's decentralized Global Information Grid (DoD's GIG) and the nation's dependence on Supervisory Control and Data Acquisition (SCADA) systems present challenging security issues. Effective security in these and many other federated environments is best implemented in layers, employing intelligent security mechanisms both centrally and on the end nodes.

The proliferation of cyberattacks will eventually overwhelm signature and rule-based approaches [1], and many critical applications and files must be permitted to continue to run or exist even when under attack. Many current solutions, however, rely on signature-based detection, kernel modifications, prevention of selected system functions while critical applications are running, the deletion or encryption of sensitive material while selected system functions are permitted, or computationally expensive data mining for anomalies [2][3]. Each of these approaches fails to meet at least one of the following desirable goals: detection of zero-day attacks, continuous operation of critical systems while under attack, widespread applicability of the technique, and real-time protection.

New approaches using machine learning and a focused set of input features [4] promise to revolutionize defensive systems. Support Vector Machines (SVMs) are among the best (and many believe are indeed the best) 'off-the-shelf' supervised learning algorithms [5].

This paper describes a prototype of an intelligent, active, real-time, risk adaptable access control (RAdAC) system designed to extend the benefits of SELinux by using SELinux not only as a secure base, but also as a source of input features to an SVM that will classify events/attacks in several categories. The system is designed to be integrated into an end node in any environment, including end nodes in federated environments such as DoD's GIG and SCADA systems. By enhancing SELinux with intelligence, it is hoped that the design will lead to real-time, non-signature based defensive systems capable of detecting and taking action against hostile users in the earliest stages of an attack.

Specifically, the prototype of the defensive system is designed to be:

- Integrable into Nearly Any Computerized Device – The defensive system is designed to be integrated into nearly any Linux-based end node (any Linux system running a 2.6 kernel and using a filesystem with extended attributes), including hand-held devices, servers, workstations, notebooks, and dedicated single purpose devices;
- Zero-Impact on Protected Applications and Files – The defensive system requires no modifications whatsoever to the software and files to be protected;
- Configurable for Critical Systems – The defensive systems can be tailored to create a focused defensive system for critical files and applications and for known personnel;
- Risk Adaptable – The defensive system is an RAdAC system in which an administratively-controlled "Current Operational Need" and the attacks and events detected by the system itself together designate the current risk level;
- Modular – The defensive system is modular in order to facilitate future extensions;

- Real-Time – By leveraging SELinux, the defensive system is designed to be lightweight enough to run in real time; and

- Compatible with National Security Goals – The defensive system is designed to parallel the National Security Agency (NSA) Information Assurance Directorate's vision for securing content in DoD's GIG.

2 Prototype – A smart, active, SELinux-based RAdAC defensive system

2.1 Modular defensive system design

The modular defensive system (Figure 1) features machine learning to overcome the limitations of signature and rule-based defenses and input from SELinux to enable the system to run in real time.

Module 1 uses SELinux denials generated by local and remote system requests to produce feature vectors suitable as input to an SVM. Module 2 then uses a previously trained SVM to classify attacks/events in several discrete categories in real time. Module 3, a graded response system, provides feedback to Module 2 and selects a response appropriate for the detected event, the history of events on that system, and the current operational need.

Testing and analysis will include study of the input feature set selection, the graded response system, and the tradeoffs between error rates and performance (false negative/positive rates vs. throughput and load on the system).

2.2 Extending Module 1

Module 1 may be extended to protect critical applications and files, to detect keyloggers, and to detect attackers with physical access (Figure 2). To protect critical applications and files, application-specific and file-specific raw input would be used in addition to SELinux denials in order to generate input feature vectors for the SVM. For example, to configure the input feature extractor to protect a web server, messages from Apache2, ModSecurity, and messages specific to the protected web pages would be used as raw input to the feature parser. Keystroke dynamics [6] [7] may be used to implement detection of keyloggers and attackers with physical access. While these extensions may enhance security, the input feature extractor will be tied to particular applications, files, and users, making this design suitable only for critical systems.

2.3 Design goals

The design of the defensive system (Figure 1) adheres to DoD's Three Tenets of Cyber Security. First, SELinux's mandatory access control mechanism (MAC) limits "access points to only those necessary to accomplish the mission [thereby making] critical access points and associated security less accessible to [the] adversary." Second, dynamically relabeling the SELinux context of a critical application "moves it out of band" when under attack. Third, the graded response system "denies [the] threat capability [by imposing] appropriate penalties when [an] attack is detected" [8].

Also, the design of the defensive system supports users of end nodes in federated systems by protecting "edge users who must operate across multiple domains and communications paths, on less hardened networks, to reach other tactical mission players, and to access protected core information systems and data warehouses" [9]. The defensive system achieves this goal by using a graded response module that neither suspends critical applications nor deletes critical and files – except in the most extreme circumstances – enabling end node systems to prevent "an attack from becoming successful while allowing the executing software and associated data being protected to remain operational and trustworthy" [10][11].

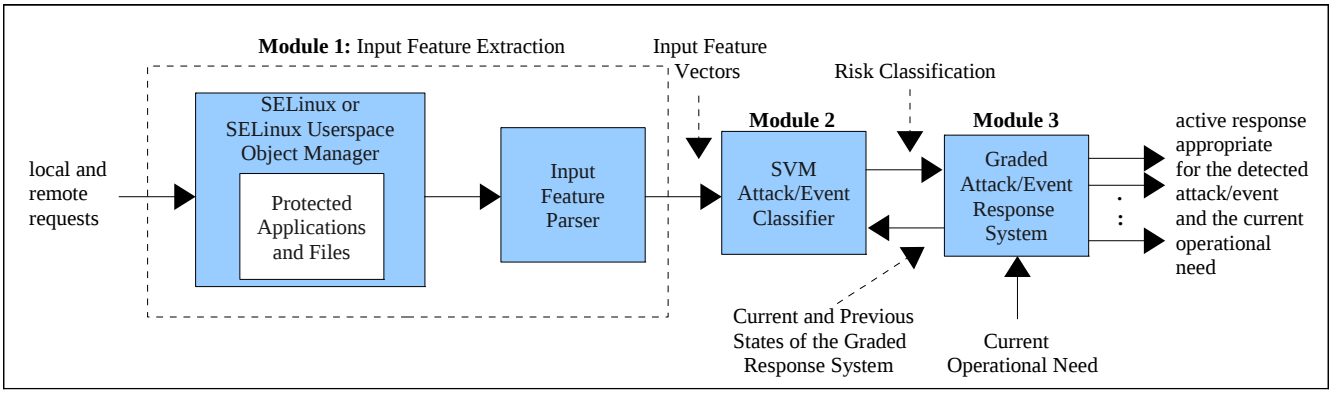

Figure 1. An Intelligent, SELinux-Based, RAdAC Defensive System

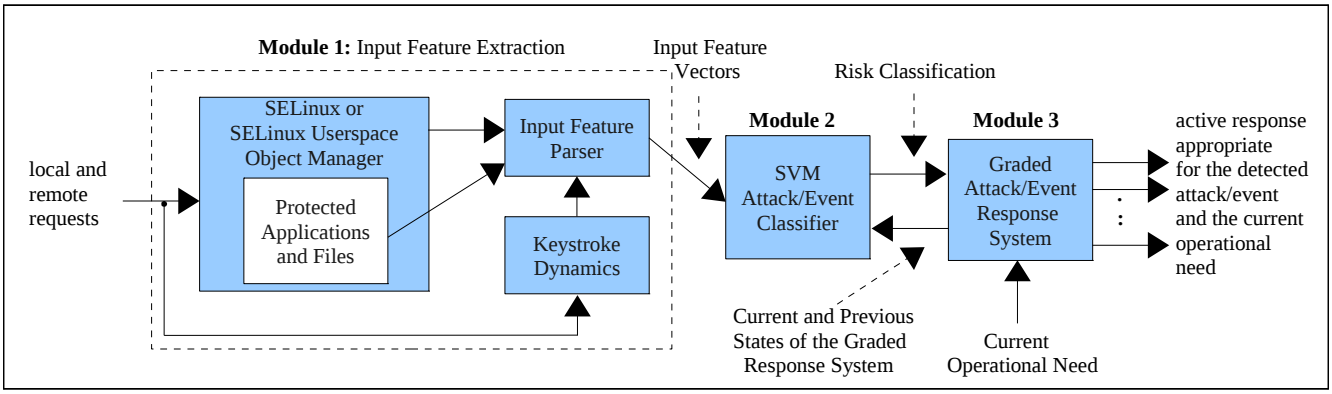

Figure 2. An Application-Specific, Personnel-Specific, Intelligent, SELinux-Based, RAdAC Defensive System

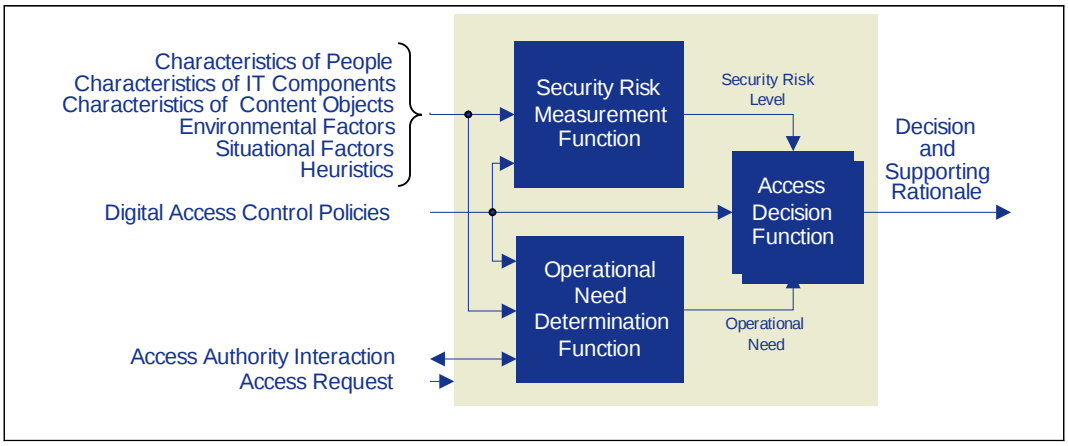

Figure 3. NSA's Vision for Access Control in DoD's GIG [12]

Finally, the design of the extended defensive system (Figure 2) parallels the NSA Information Assurance Directorate's vision for securing content in DoD's GIG (Figure 3). Modules 2 and 3 are analogous to the Security Risk Measurement Function and the Access Decision Function of Figure 3, respectively; Keystroke Dynamics, SELinux, and the Target Application messages in Module 1 are all analogous to the Characteristics of People (or other entities) [13]. The modular design facilitates future extensions that might incorporate Situational Factors or automate the current Operational Need.

3 User interface

The user interface is database-driven website designed to be friendly but restricted to authorized administrators. The main menu consists of: System, SVM, Packet Captures, Datasets, Analysis, Documentation, and Database Administration.

3.1 System submenu

The System submenu provides forms that enable the administrator to start or stop selected modules of the defensive system (Figures 4 and 5). To facilitate testing and analysis, the system permits Module 1 alone, Modules 1 and 2, or the entire defensive system to be run.

The System submenu consists of:

- Reset the SELinuxSVM Defensive System
- Start the SELinuxSVM Defensive System
- Stop the SELinuxSVM Defensive System

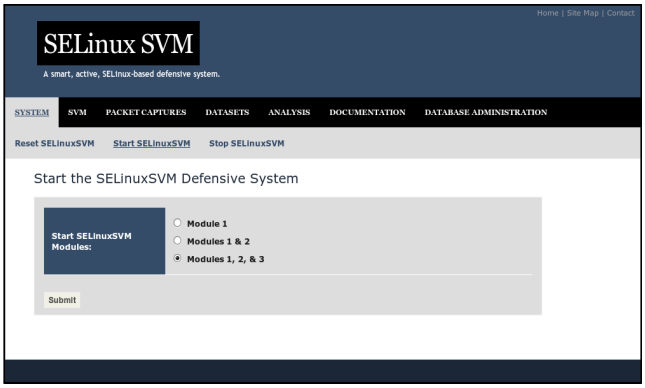

Figure 4. Starting the SELinuxSVM Defensive System

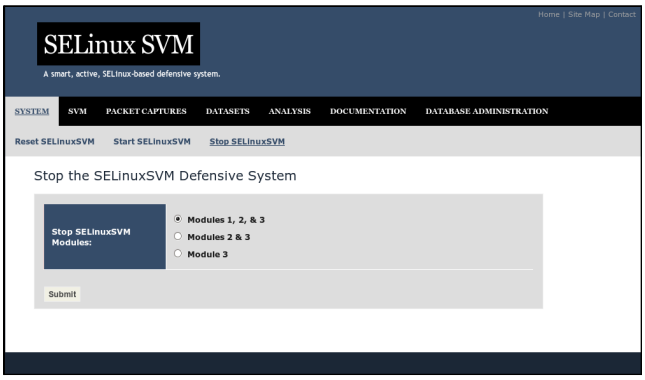

Figure 5. Stopping the SELinuxSVM Defensive System

3.2 SVM submenu

The SVM submenu provides forms that enable the administrator to select optimal SVM training parameters and dataset features as well as forms to train and test the SVM (Figure 6).

The SVM submenu consists of:

- Select Dataset Features Used to Train the SVM
- Select Parameters (Grid Search for Optimal C and gamma)
- Train the SVM
- Classify Data Points

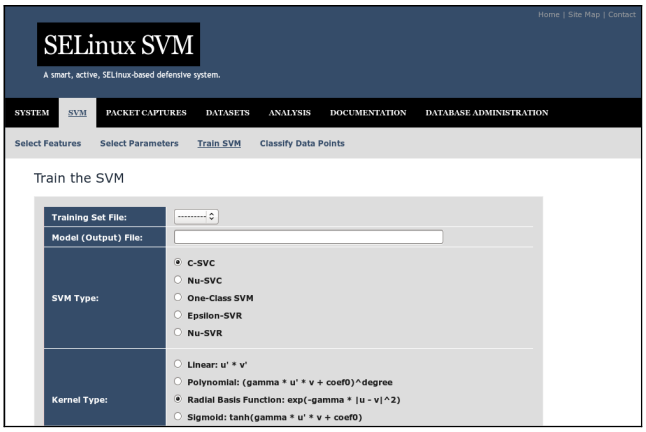

Figure 6. Training the Support Vector Machine

3.3 Packet captures submenu

The Packet Captures submenu provides forms to enable the administrator to view, replay, and filter packet capture files (Figure 7). These forms feature packet captures collected during the 2009 and 2010 Northeast Collegiate Cyber Defense Competitions (NECCDC), which are discussed in Section 7.

The Packet Captures submenu consists of:

- Filter an NECCDC 2009 Packet Capture File
- Filter an NECCDC 2010 Packet Capture File
- Filter a PREDICT Packet Capture File
- Replay a Packet Capture File
- View a Packet Capture File

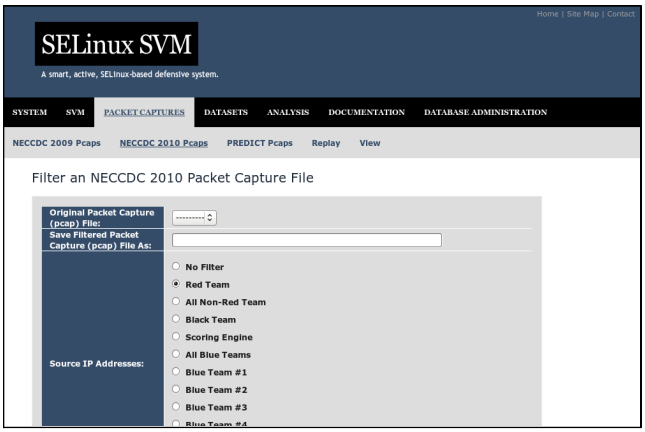

Figure 7. Filtering an Existing Packet Capture File

3.4 Datasets submenu

The Datasets submenu provides forms to enable the administrator to generate a dataset from a packet capture file, scale a dataset (Figure 8), and relabel and edit datasets. Generating datasets from packet capture files is discussed in Section 7.

The Datasets submenu consists of:

- Generate a Dataset from a Pcap File
- Scale a Dataset
- Relabel a Dataset
- Edit a Dataset

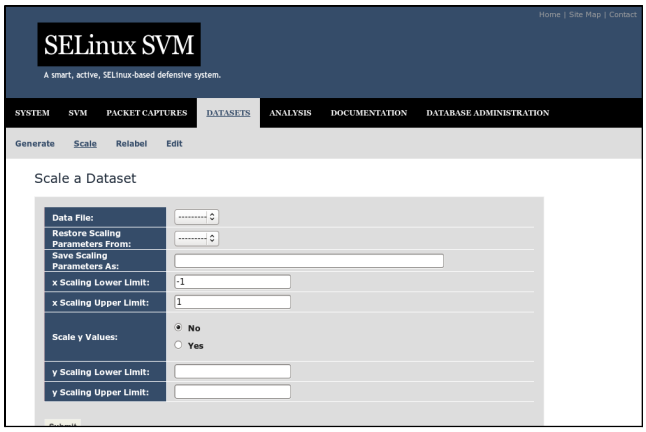

Figure 8. Scaling a Dataset

3.5 Analysis submenu

The Analysis submenu provides three performance metrics, which are discussed in Section 7, and two methods for the user to view results. “View Results” and “Plot Results”, respectively, are tools to visualize and plot two-dimensional slices of the SVM together with training or testing datasets, regardless of the number of the input features.

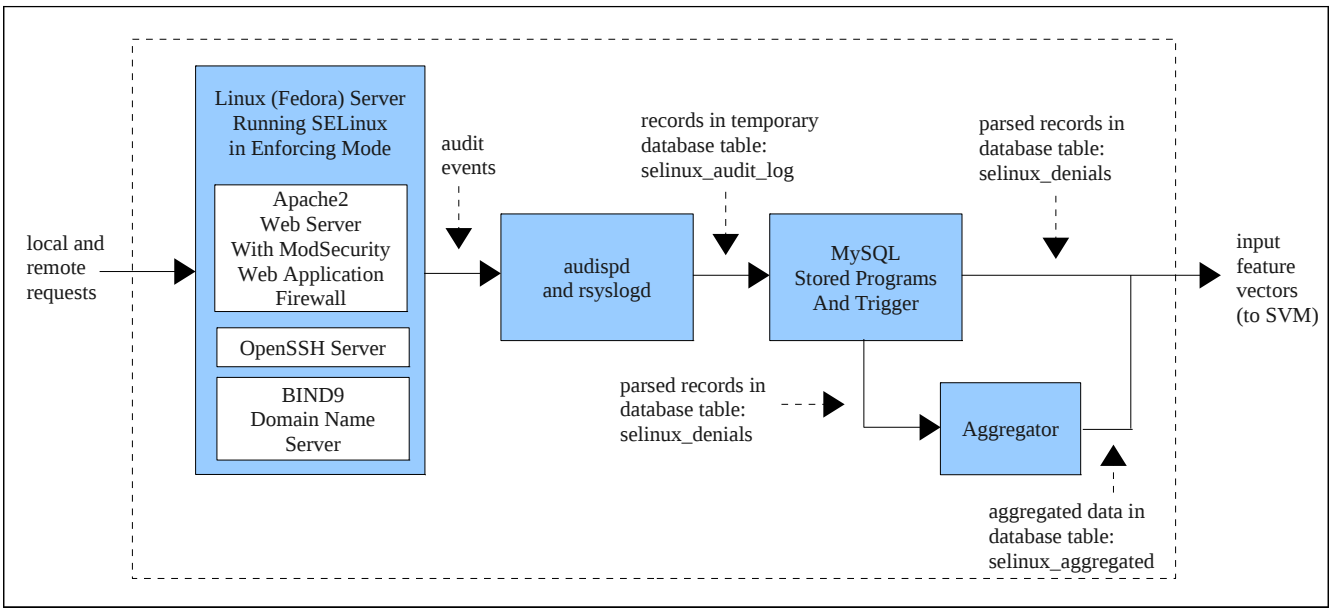

Figure 9. Module 1: Input Feature Extraction

The Analysis submenu consists of:

- Performance Metric 1: V-Fold Cross Validation Accuracy
- Performance Metric 2: SVM Training Time
- Performance Metric 3: SVM Prediction Time
- View Results (in Two Dimensions)
- Plot Results (in Two Dimensions)

4 Module 1 – Input feature extraction

Module 1, the Input Feature Extractor, automatically generates input feature vectors suitable for an SVM from local and remote requests (Figure 9). Audispd (an audit event multiplexer) and rsysogd (an extended message logging utility) are configured to enter copies of SELinux denials in a temporary MySQL database table called selinux_audit_log (Figure 10). When an entry is made in selinux_audit_log, a stored MySQL trigger parses the message to create a more useful table entry in selinux_denials (Figure 11). Offloading the parsing from the system logging mechanism to MySQL is designed to avoid a bottleneck, since parsing using rsyslogd involves time-intensive regular expression pattern matching, which is likely to be slower than MySQL stored programs. Similarly, aggregated data is collected by MySQL stored programs and entered in the selinux_aggregated table.

5 Module 2 – SVM attack/event classifier

The SVM attack/event classifier uses input from Module 1 and feedback from Module 3 in order to classify events in several discrete categories:

- Origin – an authenticated user on a tty, a user on the LAN, or a remote request;
- Number of Sources – single source vs. distributed attack;
- Target – the defensive system itself, SELinux, the operating system, the protected critical process, the protected executable, or protected files associated with the critical system;

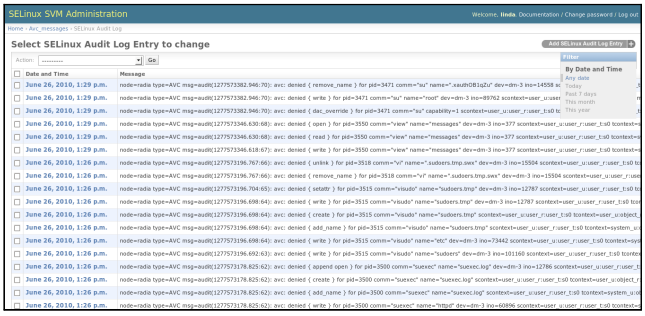

Figure 10. The SELinux Audit Log Database Table

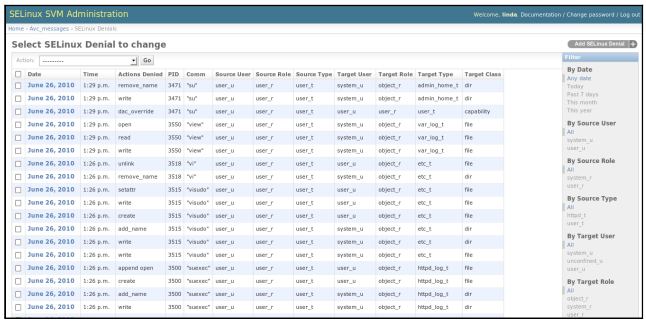

Figure 11. The SELinux Denial Database Table

- Time Span – single burst, an hourly or daily recurring event; and

- Type/Severity – single read attempt, a copy attempt over the Internet, a malicious write attempt, an unauthorized SELinux relabeling attempt, or an unauthorized attempt to transition into the SELinux sysadm_r role.

The SVM and kernel types are determined during training. Default values are C-SVC (classifier) and the radial basis function (RBF) or Gaussian kernel: $\exp(-\gamma * \|u - v\|^2)$. All SVM-related functions are implemented using libsvm [14].

6 Module 3 – Graded attack/event response system

The graded attack/event response system selects a defensive action appropriate for the classification of the attack/event as determined by the SVM, the current and previous states of the graded response system, and the current operational need. The response system selects actions appropriate for the severity of the event:

- Minor Events: In response to minor events, actions taken include alerting the administrator, filtering and saving logs, and taking a snapshot of the process tree.

- More Severe Events: In response to more severe events, actions taken include killing the offending process and processes directly related to the offending process, adding IPTables firewall rules, moving attacked files to a secure location, and relabeling the SELinux security context and Linux's discretionary access control (DAC) of the applications and files under attack.

- Extreme Events: Only in extreme circumstances (such as evidence of an attacker with physical access to the machine attempting to transition into the sysadm_r role) will critical files be deleted or critical processes terminated.

Two active responses of Module 3 specifically related to SELinux are:

- Reconfiguring Linux's DAC to dynamically manage the flow of input to the defensive system, thereby controlling the system's throughput and load; and

- Relabeling the SELinux security context of the files and processes under attack.

Relaxing the DAC causes SELinux's MAC mechanism to be consulted more frequently, increasing the load on the operating system but also catching attempted attacks at an earlier stage. If the load on the system is too great, then the DAC labels are strengthened, allowing the defensive system to continue to operate in real time. If, on the other hand, a process or file is so critical that any unauthorized attempt to read/write/execute that file would indicate an attack, then the DAC is set to the most permissive label (777) so that SELinux will be consulted on every read/write/execute request of that file, detecting the attacker at an earlier stage.

Relabeling the SELinux security context of critical files and processes under attack creates a dynamically changing protection boundary on the end node. In effect, critical files and processes are moved out of band in order to frustrate the attacker while simultaneously keeping the files and processes trustworthy and operational.

The defensive system aggressively protects itself by including the operating system, SELinux, and the system itself in the classes of targets of detected attacks. Any attempt to undermine the defensive system is be considered to be an "extreme" event.

7 Testing and analysis

7.1 Packet capture files

The packet capture files provide raw input that can be filtered and replayed to generate training and testing datasets. Packet capture files collected during the 2009 and 2010 NECCDC are currently available via the defensive system user interface. These defensive cybersecurity competitions pitted "blue" teams, each of which protected a group of servers and workstations from a "red" team charged with attacking them. The "blue" teams were prohibited from engaging in offense. A "black" team and scoring engine generated friendly traffic and monitored the services required of the blue teams' servers.

Since the IP addresses of the "blue", "red", and "black" teams are known, it is possible to filter friendly and hostile traffic using wireshark or tshark filters. In addition, the target IP addresses of the filtered packets can be rewritten to redirect the packets to a test host running the defensive system prototype. The filtered packet capture files can then be replayed to produce training and testing datasets.

7.2 Training and testing datasets

Training and testing datasets are used to train and test the SVM and graded response system. Dataset features should be scaled to prevent one feature from dominating and skewing the resulting SVM. The user interface also permits the administrator to relabel a dataset to indicate friendly traffic or hostile traffic in several classifications.

7.3 Time and performance metrics

Three performance metrics measure the accuracy and time performance of the defensive system.

The V-Fold Cross Validation Accuracy metric prevents overfitting, that is, prevents producing an SVM that is too specific for a particular dataset. Since the purpose of an SVM is to predict the classification of unknown data points, an overfitted SVM is undesirable. V-fold cross validation is a relatively simple concept:

> In v-fold cross-validation, we first divide the training set into v subsets of equal size. Sequentially one subset is tested using the classifier trained on the remaining (v – 1) subsets. Thus each instance of the whole training set is predicted once so the cross-validation accuracy is the percentage of data which are correctly classified [15].

The SVM Training Time measures the time to calculate the SVM from a dataset, and the SVM Prediction Time measures the time to make a single prediction using an existing SVM. These metrics are included to measure the defensive system's ability to run in real time, since one of the goals of the project is to attempt to design a non-signature based defensive system that can run in real time by leveraging existing security mechanisms and by dynamically adjusting the load.

8 Future work

The prototype is currently being developed. First, a complete implementation of Module 1 and preliminary implementations of Modules 2 and 3 will be completed and the NECCDC 2009 and 2010 packet captures will be used to generate testing datasets. Following a complete analysis of Module 1 using the preliminary prototype, Modules 2 and 3 will be fully implemented, the PREDICT packet captures will be added to the files used to generate testing datasets, and the entire defensive system will be tested and analyzed.

9 Acknowledgments

The author thanks Dr. James Fastook, Dr. Phil Dickens, Dr. Bruce Segee, and Dr. George Markowsky of the University of Maine and Dr. Danny Kopec of the CUNY (City University of New York) Graduate Center and Brooklyn College for their invaluable advice and support.

10 References

[1] Y. Song, M. Locasto, A. Stavrou, A. Keromytis, and S. Stolfo, “On the infeasibility of modeling polymorphic shellcode.” Presented at the 14th ACM Conference on Computer and Communications Security (CCS 2007), October 2007.

[2] P. Kabiri and A. Ghorbani, “Research on intrusion detection and response: A survey.” International Journal of Network Security, Vol. 1, No. 2, pp. 84-102, September 2005.

[3] F. H. Smith, “Defense against root.” Presented at the 2007 International Conference on Security & Management, June 2007.

[4] B. Thuraisingham, L. Khan, M. Kantarcioglu, and K. Hamlen, “Assured information sharing for security and intelligence applications.” Presented at the 2009 Cyber Security and Information Intelligence Research Workshop (CSIIRW’09), April 2009, pp.14-19.

[5] A. Ng, “Support vector machines.” Autumn 2008, p.1. (http://www.stanford.edu/class/cs229/notes/cs229-notes3.pdf)

[6] F. Bergadano, D. Gunetti, and C. Picardi, “User authentication through keystroke dynamics.” ACM Transactions on Information and System Security, Vol. 5, No. 4, pp. 367-397, November 2002.

[7] E. Lau, X. Liu, C. Xiao, and X. Yu, “Enhanced user authentication through keystroke biometrics.” Computer and Network Security Final Project Report, MIT, December 9, 2004.

[8] Software Protection Initiative (SPI). “The three tenets of cyber security.” (http://spi.dod.mil/tenets.htm)

[9] Office of the Secretary of the Defense (OSD). Small Business Innovation Research (SBIR) FY2009.2 Program Description, April 20, 2009, p.5.

[10] D. Alberts and R. Hayes, Power to the Edge: Command...Control...in the Information Age, 3rd Printing, April 2005.

[11] Office of the Secretary of the Defense (OSD). Op. Cit., p.29.

[12] R. McGraw, “Securing content in the Department of Defense’s Global Information Grid.” Presented at the Secure Knowledge Management Workshop, State University of New York, Buffalo, September 2004.

[13] Ibid.

[14] C. Chang and C. Lin, “LIBSVM: A library for support vector machines,” 2001. (http://www.csie.ntu.edu.tw/~cjlin/libsvm)

[15] C. Hsu, C. Chang, and C. Lin, “A practical guide to support vector classification,” May 19, 2009, p. 5. (http://www.csie.ntu.edu.tw/~cjlin)