

# USim: A User Behavior Simulation Framework for Training and Testing IDSes in GUI Based Systems \*

A. Garg, V. Sankaranarayanan, S. Upadhyaya  
Dept. of Computer Science and Engineering  
University at Buffalo, SUNY  
Buffalo, NY 14260 USA

Email: {ashish, vs28, shambhu}@cse.buffalo.edu

K. Kwiat  
Air Force Research Laboratory  
525 Brooks Rd.  
Rome, NY 13441 USA  
Email: kwiatk@rl.af.mil

## Abstract

Anomaly detection systems largely depend on user profile data to be able to detect deviation from normal activity. Most of this profile data is based on commands executed by users on a system and may not represent user's complete behavior which is essential for effectively detecting the anomalies in the system. Collection of user behavior data is a slow and time consuming process. In this paper, we propose a new approach to automate the generation of user data by parameterizing user behavior in terms of user intention (malicious/normal), user skill level, set of applications installed on a machine, mouse movement and keyboard activity. The user behavior parameters are used to generate templates, which can be further customized. The framework is called USim which achieves rapid generation of user behavior data based on these templates on GUI based systems. The data thus generated can be utilized for rapidly training and testing intrusion detection systems (IDSes) and improving their detection precision. This framework can also benefit research where user behavior data is utilized to improve usability and quality of software products. One such application is in the Human-Computer Interaction (HCI) domain, where the proposed technique can provide better testing capabilities.

## 1 Introduction

Anomaly detection can be performed at various levels such as network packets, system calls (program execution time) and user command level. All the anomaly detection techniques are based on the premise that the malicious activity will be a deviation from the normal user behavior. To be able to make a distinction between normal and mali-

cious behavior, these detection systems collect and utilize data from user sessions to build user profiles. This data is initially used to train the detection system about what is *normal* and later for detecting the *malicious* activity. Traditionally the user profile data has been based only on the commands a user executes on a system.

Schonlau et al. [1] collected Unix command line data of 50 users for testing and comparing various intrusion detection methods. This dataset is useful for representing user profiles for users in a Unix like system where most of the user activity is recorded as the command sequence. It should be noted that this type of dataset would not be able to correctly represent the behavioral profiles of users working on modern graphical user interface (GUI) based systems such as Microsoft Windows and some variants of Linux operating system. This is due to the fact that command line data is not capable of capturing the users actions such as mouse activity, keyboard typing speed and mannerisms, which are essential for correctly determining user's behavior on these systems.

A careful observation of the existing anomaly detection techniques reveals that better detection mechanisms can be developed with the use of pre-deployment training datasets. As mentioned above, existing datasets provide user profile data for the initial stages of the IDS training, however they do not provide details about user behavior, which is essential to effective training and performance evaluation of detection mechanisms. The process of data generation and collection using the real system takes a long time, ranging from few weeks to years. One such case is reported by Lane et al. [2], where the data collection process took few years. Also, the type of data used for detecting malicious activity at different levels is different due to their dissimilar source of generation. This disparity makes it difficult to generate/synthesize such data automatically. There is a strong need for a tool which can automate the task of fast data generation for IDS training

---

\*Research supported in part by DARPA, under contract: F30602-00-10507

and evaluation. Modeling the user behavior and generating realistic data can provide solution to this problem. It should be noted that although simulation tools in other domains such as computer networking [3, 4], manufacturing processes (discrete event simulation) [5, 6] etc. are common and readily available, there is a scarcity of similar tools in the computer security domain.

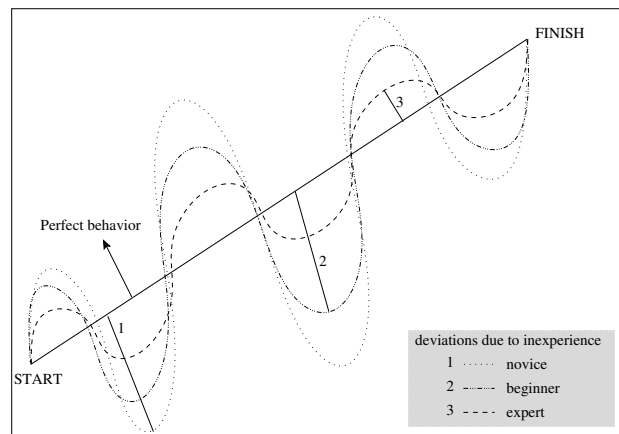


Figure 1: Behavioral Path for Various Types of Users

Figure 1 shows an example of probable behavioral paths taken by various types of users such as novice, beginner and expert. The paths shown represent the amount of efforts to achieve a task by these users drawn based on their activities on the system. These activities include but are not limited to mouse movements, keyboard typing speed, number of commands executed and duration taken to perform a job function. As can be seen from the figure, the behavior of users differs greatly depending on their skill levels. An expert user will be more towards the *perfect behavior*<sup>1</sup> statistics because of his knowledge and experience using the system. A novice user on the other hand will show a great variation in the behavior due to lack of experience and knowledge about the applications on system. Numbers 1, 2 and 3 in the figure represent the deviation from the perfect behavior due to the level of inexperience of these users with the system. For example, an expert's behavior will show shortest time to complete the job along with fewest steps (in terms of commands, mouse clicks etc.). It should be noted that although this deviation can come close to the perfect behavior line, it cannot touch it ideally. It is also interesting to note here that for performing the same job function various users will utilize the system differently thus generating a totally different behavioral path. This leads to the fact that care-

<sup>1</sup>perfect behavior is defined as the minimum effort level or system activity involved in achieving a task

fully drawn behavioral paths of various users can provide details about their behavioral profiles and thus enable a more accurate anomaly detection as compared to utilizing the command or audit data alone.

In this paper, we propose and develop a new framework called USim for generating user behavior data based on parameters provided through customizable templates. In the remainder of this section, we discuss the goals and expected impact of this framework.

## 1.1 Summary of Contributions

The main contributions of the paper are in the simulation technology of intrusion detection systems.

- Ability to rapidly model and simulate behavior of a variety of users such as administrator, developer or a secretary in an organization. This user behavior data is very useful for training and evaluating IDSeS. Different users have different skills and they will use system applications based on their needs. Thus their use of resources on a system will vary greatly based on user behavior.
- The user simulation tool generates realistic data by utilizing complete set of characteristics for simulated user. These characteristics include but are not limited to temporal or time dependent activities, such as 9am - 5pm working hours for a user to role based activities such as administrator vs. developer activities etc.
- The proposed framework can simulate users realistically and is highly tunable in terms of types of users it can simulate. It can be utilized to simulate a normal user, a malicious user, an exploratory user, a naive user or an expert user. This feature will provide more depth to the training and testing of IDSeS.

## 1.2 Paper Organization

The remainder of the paper is organized as follows: We give a literature review of the related work in the area of user simulation and intrusion detection in Section 2. Section 3 describes the design of the USim framework. Section 4 discusses the USim approach for modeling user behavior and generating customizable templates. Section 5 provides illustration of mouse movement as one of the user behavior features utilized by USim framework. In Section 6 we show the feasibility of our research by discussing applications for USim generated dataset for training and testing cyber security systems. We conclude the paper and provide future directions of research in Section 7.

## 2 Related Work

The idea of modeling user behavior is not new and has been studied in many areas including cognitive science, human computer interaction (HCI) systems, speech recognition and training systems [7]. We will mainly discuss the user behavior modeling from computer security perspective, which is the main focus of this paper. This involves being able to interpret user's behavior on a computer to identify any changes in behavior leading to security concerns. User behavior models try to catch the higher level interactions by a user on a computer. These interactions can be in the form of commands a user runs on the system, his habits in using a particular feature of system (keystroke combination or typing command completely), his nervousness during the usage of computer etc. These interactions are also based on the skill level of a particular user. A novice will use the computer differently as compared to an expert user and thus generates a different set of behavioral characteristics. While using the commands as a behavior characteristic, the inter-command dependencies are an interesting way to look at the intention of the user. These inter-command dependencies of user behavior have been modeled using Markov chains by Menasce et al. [8], Calzarossa et al. [9, 10], and Chen et al. [11]. Although inter-command dependencies can provide intentions of the user, they cannot provide complete user behavior characteristics required to make an informed decision about malicious activities on a system.

Our work finds great relevance in areas that utilize user behavior data to make certain decisions about users. Specifically the domain of application for this work would be anomaly detection techniques. So first we mention some well-known works in this area and the improvements in the intrusion detection techniques over the past few years. Then we describe some of the publicly available datasets, which have been utilized for anomaly detection in the previous efforts. Finally, we discuss our work in the context of user behavior modeling and data generation using simulation tools available both in the academic and commercial domains.

We would like to refer to our previous work [12] where we developed a similar tool to generate user command dataset. RACOON, as this tool is called, is able to extract features from an existing dataset and use those features to generate volumes of similar data to be used for training and testing anomaly detection systems. RACOON is a very effective tool for systems where user commands are the most prominent way of communicating with system and these commands themselves can be effectively utilized for constructing user profiles. This tool however

will not be able to generate complete profiles of users on a GUI based systems. USim fills in the gap by providing ability to generate GUI based behavior datasets containing information about user's mouse movements, keyboard activity etc.

### 2.1 Anomaly Detection Based on User Data

User action based anomaly detection systems have been proposed and developed by many research groups. Lane et al. [13, 14] assert that user actions are causal in nature and sequence matching can provide appropriate solutions for detecting anomalies. They show that a Hidden Markov Model can provide a good approximation of user command behavior [15]. In another work, Schonlau et al. [16] utilized various statistical techniques ("Uniqueness", "Bayes one-step Markov", "Hybrid multi-step Markov", "Compression", "IPAM" and "Sequence Match") and for evaluating their effectiveness in masquerade detection. Naive Bayes Classifier was used by Maxion et al. [17] on a truncated user command dataset. They provide results to prove that their technique improves detection significantly giving very low false positives. In a later work [18] they claim that enriched command dataset resulted in a better detection accuracy. A recent effort using data mining [19] for detecting masqueraders is also worth mentioning here. We leverage some ideas from these efforts and show how a user behavior profile can provide better detection capabilities.

### 2.2 Data Simulation Tools

In computer security domain the first tool to give a realistic dataset pertaining to the domain was LARIAT from MIT/Lincoln Labs [20]. LARIAT uses highly customizable scenarios and plays them back at the test network. Debar et al. proposed and developed an experimental workbench [21] for intrusion detection systems. The proposed workbench simulates user interaction with various network services. Another significant effort in the area of data simulation is by Lundin et al. [22]. It should be noted that although, all these efforts are related to data simulation and generation in the computer security domain, there is no tool in our knowledge that addresses the user behavior issues, which are very important when considering masquerade detection. USim addresses that concern very well and provides a realistic dataset for user behavior.

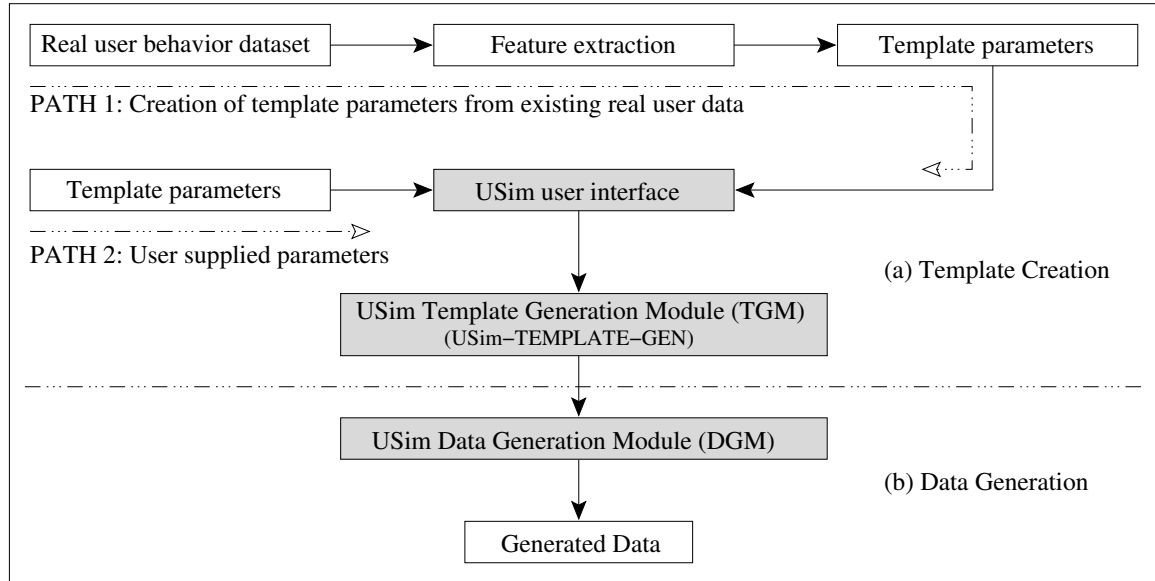


Figure 2: Data Generation Using USim

### 3 Design of USim

USim is a specialized tool designed for generating user behavior data. USim generates this data based on provided customizable templates. There are two ways a template can be created in USim. Firstly, the proposed framework can take an existing real user behavior data and extract parameters from this data for creating templates. Secondly, USim can also be fed precise values of the parameters of a simulated user for creating templates. Figure 2 shows these two paths of the USim framework and data generation methodology. This methodology can be used to rapidly generate large volumes of user behavior data. This data is needed for thorough training and testing of intrusion detection systems.

When USim is supplied with the real user behavior dataset (see PATH 1 in Figure 2), it extracts the simulation parameters from this dataset. These parameters are passed to the USim template generation module (TGM). These parameters could consist of, but are not limited to, user behavior (normal/malicious), user session scope (type of applications being used), user role (admin/developer) etc. Each parameter is specified with a probability and modeled as appropriate for the required user behavior. TGM module generates USim understandable template and sends them to the data generation module of USim (DGM). USim uses XML for representing templates due to its wide popularity and cross-platform compatibility. Once DGM receives the templates, it generates appropriate data based on the provided parameters.

The second branch of USim (see PATH 2 in Figure 2), deals with the parameters directly passed for creating templates. These parameters are expected to be as precise as possible and represent a type of users (like a group of developers or graphics designer). The created template can be tuned later on depending on type of user being simulated. In the remainder of this section and the next section, we discuss the USim framework in detail and describe the various terms and techniques used.

#### 3.1 Real User Behavior Data

Data collected during a user’s activities on a system in real-time is called *real user behavior data*. This dataset describes a user’s behavior according to his/her job assignment. The real user behavior data is utilized by the USim framework for extracting features and creating templates for generating similar data. The generated data in this case should closely resemble the real data.

#### 3.2 Dataset Structure

Structure of the data generated using USim framework is shown in Figure 3. As can be seen from the figure, the dataset contains details about user behavior such as executed commands, mouse and keyboard activities, currently running background processes etc. The mannerism of user behavior provides information about deviation from normal behavior and is a useful feature for improving detection accuracy.

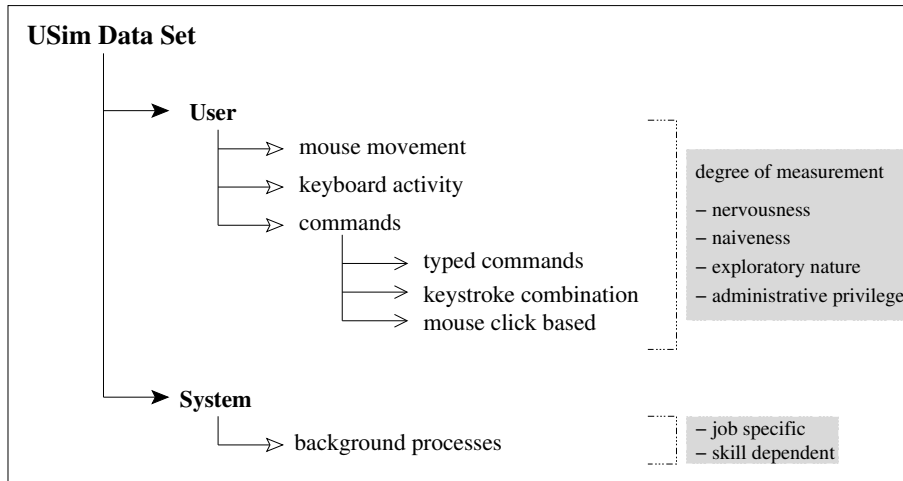


Figure 3: USim Dataset Structure

### 3.3 Simulated User Behavior

Another approach taken by USim is generating user behavior data based on specifications provided to the framework. The specifications are converted to templates, which are customizable and can be tuned for a specific purpose. These templates contain information such as user's job function, type of user, mannerisms of keystrokes and mouse movements etc. The data thus generated is said to be for a *simulated user*.

## 4 Modeling User Behavior

Anomaly detection mechanisms rely heavily on the understanding of user's behavior for detecting malicious activity effectively. User behavior is a complex process and to be able to efficiently model it, we identified some key features from the Human factors domain such as nervousness while using the computer, keyboard and mouse activity, typing speed etc. [23]. These features help explain USim framework and put our work in perspective.

- **Command:** The most basic entity to achieve a certain goal on a computer is called *command*. A command is the most prominent feature in the user behavior dataset. It describes the goal and hence user intention. Example of a command is `ls` and `vim` Unix.
- **Meta-command:** The category of a command is called *Meta-command*. Commands related to the same goal are collectively categorized as one meta-command. For example `vi` and `emacs` belong to `editor` meta-command.

- **Session scope:** The set of meta-commands to achieve a set of goals pertaining to users role is called his/her session-scope. For example, a sample session-scope of a user, whose role is *developer*, would be meta-commands: `editor`, `compiler`, `linker`, `debugger`. It should be noted that the above three features are referred from our previous work in [12].
- **User role:** A user's role represents the scope of his/her responsibility. User's role also dictates his/her session-scope.
- **Key-time:** Usual time difference between key strokes for a particular user's *normal* profile is called *Key-time*. It should be noted that this is a very useful feature and in limited logging scenarios may be used for separating masqueraders from normal users.
- **Mouse-graph:** The movement of mouse based on user's level of nervousness. A high mouse-graph may either mean high nervousness of an insider or high anxiety level for a masquerader.
- **Posture and body language:** The way a user is sitting while working also plays an important role in determining his level of comfort while working. Again a masquerader won't be too much concerned about his posture or body language but a normal user would first try to seat himself properly and then start working. This particular feature although very useful in determining a malicious user physically, is out of scope of this type of framework and hence will not be considered in our model. It is provided here for the sake of completeness of the model.

## 4.1 User Behavior

User behavior in cyber domain can be described as the way a user performs his activities on a computer system. This includes the applications and resources utilized, the temporal relationships between the various activities performed on the system (time between various commands, key strokes etc.), the movement of mouse (a nervous/malicious user might move the mouse more rapidly than a normal user). USim aims to simulate this user behavior (also called the user behavior *profile*) and provide realistic datasets representing this behavior. The advantages of such datasets are many fold. They can help in determining the efficacy of the systems algorithms, the response of the system to such users and the performance of the system.

User behavior can be defined with the help of two classes of parameters. Firstly, the system settings such as mouse movement speed setup, left handed or right handed mouse, individual application being used and screen resolution. Secondly, the environmental/technical factors like knowledge of the peripheral device being used, user's comfort level with the system and expertise with the particular application being used. A good example of behavioral differences in two people is a secretary in an organization versus a graphics designer. While the secretary would be utilizing the keyboard more for typing the documents and memos, the graphics designer would be involved in heavy mouse movement activities due to the nature of his work. The overall approach of USim framework is to provide customizable templates to capture this user behavior and help in generating datasets to represent the behavior.

## 4.2 Template Generation

We consider a particular template that satisfies the requirements of most of the anomaly detection algorithms available today. Consider a desktop system with a user working on it to fulfill a given job requirement. We consider a Microsoft Windows environment for illustration purposes, although the template can easily be modified to suit any operating system. Most of the anomaly detection schemes require a series of commands that have been executed by the user. At least, till date, the data that has been fed into such algorithms have been that way, i.e., a list of commands. Such lists, while adequately representative of a UNIX system of the past decade, are no longer meaningfully representative of the systems today. Graphical user interfaces (GUIs) have become the norm for most of the computer systems today. Hence user characteristics are better represented by datasets that are more

expansive and take into account the user's operation on a GUI system rather than just the command list. Simulating such behavior has unique challenges and is an issue that has never been addressed before. A typical Microsoft system has user run processes and system run background processes. The load on the system, which includes both these processes, also vary according to the users command execution. While we say command execution, we include the applications run by the user such as MS Word and Internet Explorer in addition to the usual command prompt tools/utilities. Furthermore, two additional user characteristics are included, which are the mouse movement and the keyboard typing speed. These characteristics have largely been ignored by the simulation tools so far, which is surprising since they are the closest we can get to the user characteristics apart from the commands they execute. We describe in further detail the template for generating dataset encompassing such characteristics:

- The first manner in which we can generate simulated data is from a set of real data. Real data for various reasons, privacy being the foremost, are hard to come by for a system. However, in cases when they are available, USim can take two inputs viz. the real data and a template that effectively describes the dataset and an expected distribution of the dataset features. With sufficient amount of real data, we do not require the expected distribution of the dataset features. This methodology of generation is similar to our prior work in [12].
- Secondly, we describe a customizable template that can be used to generate user data taking into account the GUI characteristics of the system. As mentioned before, mouse movements and keyboard typing characteristics are the two features that come very close to representing the user (almost in a biometric manner, similar to signatures and thumb-prints). This template can be used for training and testing any intrusion detection system.

The template also specifies the roles of the user and can provide precise behavior profiles accordingly. The template for a graphics designer will have more graphics applications as compared to an administrator. Hence USim can generate the mouse movements and command streams accordingly (adhering to profile, exploratory user or malicious user). In addition, the system background processes and the load on the system also change correspondingly. A complicated graphics rendering engine might take a huge amount of load. A system utility may also generate a huge load. However when a system utility is executed,

background processes with the system credentials (as opposed to the admins credentials) will be fired up. The template specifies these criteria also and USim generates the command stream appropriately. This section thus illustrates how flexible template generation is through the USim architecture.

### 4.3 Mathematical Representation of Behavior

We define the user behavior mathematically based on the features and parameters extracted in the previous subsections. USim framework can be directly fed the parameters required for creating templates. These templates can be used to generate user behavior data. The following parameters are used by the Template Generation Module (TGM) of USim framework as shown previously in Figure 2:

- User intention:  $M/G$ , where  $M$  = malicious and  $G$  = good
- User type (skill level):  $U_{t_1}, U_{t_2}, \dots, U_{t_n}$
- Left/Right handed user:  $L_u/R_u$
- Operating System:  $O_1, O_2, \dots, O_p$
- Application set on machine:  $A_1, A_2, \dots, A_m$
- System set:  $S = \{O_k, A_m\}$
- Simulated role:  $R_1, R_2, \dots, R_l$
- Duration of session:  $T_d$

where the user role  $R_i$  can be defined as:

$$R_i = f(O_j, A_k)$$

Based on the above parameters, we can define the behavior  $B$  of user  $U_i$  as follows:

$$B(U_i) = f(U_{t_i}, M/G, L_u/R_u, A_j, O_k, R_l, T_d)$$

where the symbols have their respective meanings as above.

Similar to the parameters defined above for template generation, USim can also utilize the existing user behavior dataset for extracting features. These features can be plugged into a template to generate similar data. We used a logging tool to generate the real user data for extracting features. The features thus extracted are:

- Stream of commands issued by the user.
- Keystroke sequence with relevant timing information.

- Background processes and related information like CPU usage and user information.
- Number of Left/Right clicks/duration of activity, where duration can be anywhere from 10 seconds to few minutes or even hours.
- Number of Left clicks following Right clicks.
  - inside the area of right click menu
  - outside the area of right click menu
- Speed of mouse movement (pixels/ms).
- Number of extreme clicks (left bottom or right top), responsible for closing an application or starting a new one.
- Intentionality of movement: defined as number of movements culminating in a left/right click and their speed.
- Mouse scrolling speed.
- Idle mouse clicks: resulting in no useful activity on system. It should be noted that this is a crucial feature, as this will greatly vary from person to person.
- Transition probabilities: probability of a command on mouse click after another similar action has been performed.
- Command instantiation probability: probability of a mouse click or command execution followed by a mouse movement.
- Coordination probability (or factor): probability of coordination of mouse movement along with a click or command execution.

We now describe an algorithm called USim-TEMPLATE-GEN (see Algorithm 1), which can generate templates suitable for USim framework for generating user behavior datasets. The inputs to this algorithm are *System set*  $S$ , describing the system parameters such as operating system and applications running on the system, *User role*  $R$ , consisting of jobs assigned to simulated user, *Behavior set*  $B$ , a collection of user behavior characteristics, and number of items in  $R$  ( $rcount$ ) and  $B$  ( $bcount$ ).

---

**Algorithm 1** USim Data Generation Algorithm

---

USim-TEMPLATE-GEN $\{S, R, B, rcount, bcount\}$ 

```
1  $X[rcount] \leftarrow$  zero matrix % initialize the matrix
2  $Y[bcount] \leftarrow$  zero matrix % initialize the matrix
3 while ( $R$  is not empty) % till there is data in  $R$ 
4 do For each user role  $r_i \in S$ , add it to  $X$ 
5   For each behavior  $b_j \in B$ , add  $average(b_j)$  to  $Y$ 
6 Convert frequencies in  $Y$  to probabilities
7  $Z[rcount, bcount] \leftarrow$  zero matrix % initialize the matrix
8 for  $X[bcount]$ 
9   for each  $Y[bcount]$ 
10  $Z[rcount, bcount] \leftarrow X[rcount], Y[bcount]$  % copy data from  $X$  &  $Y$  to  $Z$ 
11 while ( $Z[rcount, bcount]$  is not empty)
12 do For each behavior item  $b_i$ ,
13   Calculate behavior probabilities related to each job
14   Calculate transition probabilities for each activity
15   Store probabilities, user role and system set to  $B$ 
16 Convert  $B$  to USim template
```

---

License Stuff			
Initialized at	8/24/2005 2:55:25 PM		
Computer Name	shail		
System User Name	SHAIL\ Ashish Garg		
User belongs to Administrative Group			
IP Address 0	192.168.236.128		
632604921288593750	8/24/2005 2:55:28 PM	Mouse Coordinates	774,738
632604921289375000	8/24/2005 2:55:28 PM	Left Click	
632604921795625000	8/24/2005 2:56:19 PM	KeyDown	P
632606530093340000	8/26/2005 11:36:49 AM	New Process	acrobat.exe

Figure 4: A Sample Dataset Generated by Logging Tool

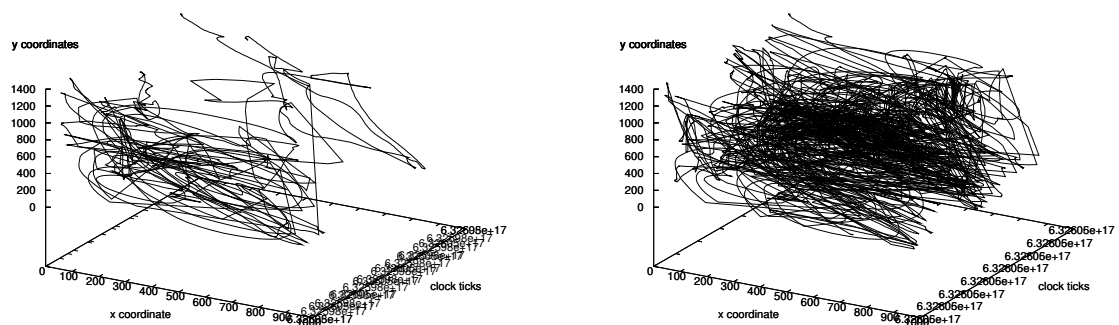


Figure 5: Retraced Graph of Mouse Movement for Two Users

## 5 Illustration of a Feature: Mouse Movement

We collected data generated by user activities on 2 different computers. These users were of different skill level 8

and they performed different jobs during the login period. This data contains the commands/applications executed, mouse movement, keyboard activity and the system back-

ground processes running during the login session. We developed a logging tool for Windows XP platform to collect the data. We used Microsoft .NET framework and C# language for development. A sample log file from this tool is shown in Figure 4. We sanitized the dataset generated by the logger and retraced the user mouse movements on the system. Two such sample graphs are shown in Figure 5. The graphs in the figure represent the mouse movement of a single user during two sessions with unequal time durations. This type of graphs can be drawn for multiple sessions of a single user or for a single session of multiple users. As can be seen by these graphs, the difference in behavior of users can drastically change with the job function and the duration of session. These graphs also provide an insight into how nervous a user is (rapid movements of mouse). Comparing the two graphs from the same users during two different sessions can lead to clues about the variation or abnormality in his behavior and may indicate an intrusion or masquerade activity. The research issues in determining the similarities in the mouse graphs are time duration of the user session and time of the day (fatigue may cause changes in behavior).

We demonstrated these mouse graphs to try and predict the similarities in the behavior of a single user across multiples sessions and also to predict the differences between different users. A more detailed analysis is being done to provide evidence that this feature will be an important criteria for detecting anomalies.

## 6 Applications of USim Generated Dataset

USim finds great relevance in the systems relying on the user behavior data. In this section, we present two applications as a proof-of-concept for USim generated dataset. The first application is a document management system (DMS). A document management systems is used to manage and protect the digital documents in an organization (see Figure 6).

A Document Management System (DMS) provides for secure management and collaborative editing of digital documents. Most of the activities on the digital documents are performed on GUI based systems. These activities involve mouse movements providing degree of nervousness, text reading speed (based on scroll speed), opening or closing applications and copy or paste text, keyboard activities such as commands, typing the documents, copy or paste text and opening or closing applications.

Training the IDS and testing it against any malicious

activity on a DMS is a difficult task and involves the knowledge of user's behavior while using the system. The changes in behavior can be utilized to detect the anomalies in the system. Also, the behavioral characteristics in these systems are useful to predict the misuse of resources. In short, to be able to train and test these systems, there is a strong need for a USim like framework which can accommodate the requirements of DMS by providing valid datasets for training and testing. Above figure describes how USim generated dataset can be used effectively in such a system. Another application of the USim generated dataset is in the Human-Computer Interaction (HCI) domain. A definition of HCI [24] is:

“Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.”

As can be seen from the definition, HCI desires to know user's behavior in a particular environment. Once this behavior is known, HCI can provide guidance as to what part of the product/services can be improved. Web page usability studies rely heavily on user behavior data to determine if the user is able to find certain links on a page easily or in determining the best position for certain buttons. USim provides such behavior data for training and testing, which can be effectively utilized for improving web pages.

## 7 Conclusion and Future Work

In this paper we presented a new approach of generating user behavior profiles in the form of datasets. The generated datasets will not only include the traditional user command data but will also contain the behavior characteristics of users such as mouse movements while a system is being used and keyboard characteristics such as typing speed. USim can use the existing real user data as an input to extract features necessary for creating templates. These templates can be used to rapidly generate volumes of similar data. The proposed framework also supports creation of templates by supplying precise parameters for describing the user behavior. The templates created by USim framework are highly tunable and require minor changes in the parameters to provide similar datasets for covering a range of user behaviors. The generated datasets can be used for rapidly training the IDSes and thoroughly testing their detection capabilities. Currently USim is a framework to create templates for gener-

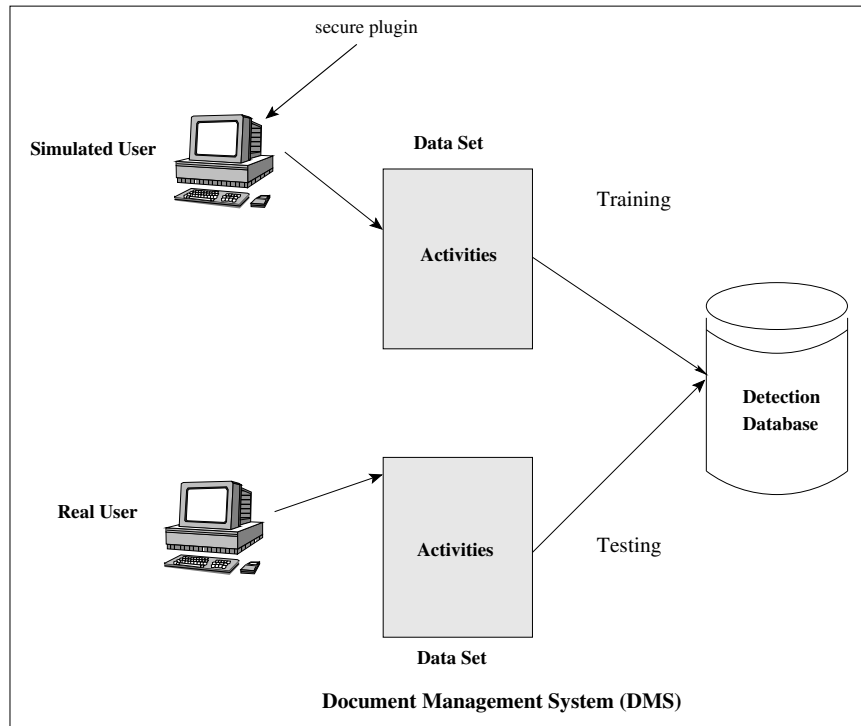


Figure 6: A Sample Application of USim Generated Dataset

ating user behavior dataset. We are working on creating the templates for use with data generation module.

Once the datasets are generated by USim framework, we plan to validate these datasets by utilizing various statistical means. For the first branch of USim, where the real user behavior data is provided as input to the framework, a comparison between the real user data with the USim generated data will provide details about the *closeness* of these datasets and thus the efficiency of USim. For the second branch of USim, where the parameters for template creation are directly provided to the framework, the validation of USim generated dataset is trickier. It can be done by testing a detection system with a similar real user behavior data and USim generated data and then comparing the performance of the detection system. We would like to stress that validation of the USim generated datasets is necessary and will provide a measure of the usefulness of our tool.

We would also like to test our tool and the generated dataset with more IDSes to determine the effectiveness of our method. In future we plan on extending our tool to incorporate simultaneous generation of multiple user behavior data to be able to simulate the profiles of people collaborating to attack larger systems, i.e., coordinated attack scenarios.

Our work also finds great application in the Human-Computer Interaction (HCI) related research. HCI deals with the usability of user interfaces and how human behave in a particular environment when using computers. USim tries to model user behavior and generated datasets based on behavior. This data could also be used in experiments to simulate various users and draw conclusions based on the statistics. These conclusions can be used by HCI researchers to improve the usability of products and services.

## References

- [1] M. Schonlau. Masquerading User Data, 1998. <http://www.schonlau.net/intrusion.html>.
- [2] T. Lane. Purdue UNIX User Data. [http://www.cs.unm.edu/~terran/research/data/Purdue\\_UNIX\\_user\\_data.tar.gz](http://www.cs.unm.edu/~terran/research/data/Purdue_UNIX_user_data.tar.gz), 1999.
- [3] X. Chang. Network Simulations with OPNET. In *Proceedings of the 31st conference on Winter simulation*, pages 307–314. ACM Press, 1999.
- [4] Information Sciences Institute, USC School of Engineering. The Network Simulator: ns2,

- <http://www.isi.edu/nsnam/ns/>. Accessed on Jan 05, 2003.
- [5] R. Chawla and A. Banerjee. A Virtual Environment for Simulating Manufacturing Operations in 3D. In *2001 Winter Simulation Conference*, 2001.
- [6] L. Silva, A. L. Ramos, and P. M. Vilarinho. Using Simulation for Manufacturing Process Reengineering - A Practical Case Study. In *2000 Winter Simulation Conference*, 2001.
- [7] W. Eckert, E. Levin, and R. Pieraccini. User Modeling For Spoken Dialogue System Evaluation. In *Proceedings of Automatic Speech Recognition and Understanding Workshop*, 1997.
- [8] D. Menasce, R. Fonseca V. Almeida, and M. Mendes. Resource Management Policies for e-commerce servers. In *2nd Workshop on INternet Server Performance (WISP 99)*, 1999.
- [9] M. Calzarossa, R. marie, and K. Trivedi. *System performance with User Behavior Graphs*. 1990.
- [10] M. Calzarossa, G. Haring, and G. Serazzi. *Workload Modeling for Computer Networks*. Springer-Verlag, Muenchen, 1988.
- [11] C. Chen. Structuring and Visualizing the WWW by Generalized Similarity Analysis. In *Proceedings of Hypertext '97*, 1997.
- [12] R. Chinchani, A. Muthukrishnan, M. Chandrasekaran, and S. Upadhyaya. RACOON: Rapidly Generating User Command Data for Anomaly Detection from Customizable Templates. In *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC '04)*, Tucson, Arizona, December 6-10 2004.
- [13] T. Lane and C. E. Brodley. An Application of Machine Learning to Anomaly Detection. In *Proceedings of Twentieth National Information Systems Security Conference*, volume 1, pages 366–380, Gaithersburgh, MD, 1997. The National Institute of Standards and Technology and the National Computer Security Center.
- [14] T. Lane and C. Brodley. Sequence Matching and Learning in Anomaly Detection for Computer Security. In *Proceedings of AAAI-97 Workshop on AI Approaches to Fraud Detection and Risk Management*, pages 43–49, 1997.
- [15] T. Lane. Hidden Markov Models for Human-computer interface modeling. In *Proceedings of IJCAI-99 Workshop on Learning About Users*, pages 35–44, 1999.
- [16] M. Schonlau, W. DuMouchel, W.-H. Ju, A. F. Karr, and M. Theus and Y. Vardi. Computer Intrusion: Detecting Masquerades. In *Statistical Science*, volume 16, pages 58–74, 2001.
- [17] R. A. Maxion and T. N. Townsend. Masquerade Detection Using Truncated Command Lines. In *Proceedings of International Conference on Dependable Systems and Networks (DSN '02)*, pages 219–228, 2002.
- [18] R. A. Maxion. Masquerade Detection Using Enriched Command Lines. In *Proceedings of International Conference on Dependable Systems and Networks (DSN '03)*, San Francisco, CA, June 2003.
- [19] K. Wang and S. J. Stolfo. One Class Training for Masquerade Detection. In *ICDM Workshop on Data Mining for Computer Security (DMSEC 03)*, 2003.
- [20] L. M. Rossey, R. K. Cunningham, D. J. Fried, J. C. Rebek, R. P. Lippmann, J. W. Haines, and M. A. Zissman. LARIAT: Lincoln Adaptable Real-time Information Assurance Testbed. In *2002 IEEE Aerospace Conference*, March 2002.
- [21] H. Debar, M. Dacier, A. Wespi, and S. Lampart. An Experimentation Workbench For Intrusion Detection Systems. Technical report, IBM Research, Zurich Research Laboratory, 1998.
- [22] E. L. Barse, H. Kvarnstrom, and E. Jonsson. Synthesizing Test Data for Fraud Detection Systems. In *Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC 2003)*, 2003.
- [23] S. J. Bates. User Behavior in An Interactive Computer System. *IBM Systems Journal*, 13:1–18, 1974.
- [24] Hewett, Baecker, Card, Carey, Gasen, Mantel, Perlman, Strong, and Verplank. ACM SIGCHI Curricula for Human-Computer Interaction. <http://sigchi.org/cdg/index.html>.
- [25] S. J. Stolfo, S. Hershkop, K. Wang, O. Nimeskern, and C. Hu. Behavior Profiling of Email. In *1st NSF/NIJ Symposium on Intelligence and Security Informatics (ISI 2003)*, Tucson, Arizona, USA, June 2-3 2003.