Proceedings of the 5th WSEAS Int. Conf. on SIMULATION, MODELING AND OPTIMIZATION, Corfu, Greece, August 17-19, 2005 (pp596-603)

Detecting Hostile Accesses to a Web Site Using a Visualization Method Based on Probabilistic Clustering

NAOKO HIROSE EINOSHIN SUZUKI
Electrical and Computer Engineering
Yokohama National University
79-5 Tokiwadai, Hodogaya, Yokohama 240-8501
JAPAN
http://www.slab.dnj.ynu.ac.jp/

Abstract: - In this paper, we propose a visualization method based on probabilistic clustering in order to detect hostile accesses to a Web site. A system administrator is required to monitor a huge amount of access log data in order to detect novel types of hostile accesses. Our PrototypeLines is a visualization method based on probabilistic clustering with a single parameter that must be tuned and has been successful in medical domain. Thus we believe that PrototypeLines is more attractive than conventional hostile access detection methods based on machine learning since each of the latter methods typically has many parameters that must be tuned. We modify our PrototypeLines for hostile access detection and investigate its performance by experiments with real data. Experimental results show that our method is effective in detecting hostile accesses since it provides a display of a large amount of access sessions in a compact manner emphasizing hostile accesses with warm colors.

Key-Words: - PrototypeLines, Visualization, Web Log Analysis, Fraud Detection, EM algorithm, Hue Allocation.

1 Introduction

Hostile accesses to Web sites have caused serious damage to our community and have been increasing in number. For instance, Code Red Worm[1] and Nimda Worm[2], each of which represents a powerful virus, have spread all over the world rapidly. It is mandatory for a system administrator to gather security information and perform system updates frequently.

It is well-known that antivirus is effective for hostile accesses which are already known. Thus, detection of novel types of hostile accesses has attracted attention of researchers. Such detection can be situated among fraud detection research and application of fraud detection techniques which are based on a machine learning method is expected to be promising. Such methods include association rule [7], classification [5], statistical test [4], meta learning [2], outlier detection [10], and clustering [8]. However, such a method has a deficiency that its performance depends on many parameters that must be tuned.

Due to intuitive comprehensibility of its results, information visualization is highly effective in application domains where user-interaction is required, and is gaining importance in data mining [1,6]. Among such visualization methods, our PrototypeLines [9] is based on probabilistic clustering and has been successful in medical domain. The success deserves attention since PrototypeLines has only one parameter that must be tuned and provides an intuitive display of a large amount of data emphasizing medical test results that require special attention with warm colors. In this paper, we modify our PrototypeLines for hostile access detection to Web sites and investigate its performance by experiments with real data.

2 Detection of Hostile Accesses

Each time a user issues a request to a Web site, an access log entry is generated. We show below an example of a Web access log entry.

kali.slab.dnj.ynu.ac.jp - ningen [04/Apr/2001: 16:48:57 +0900] "GET /algodata HTTP/1.0" 401 476

This log entry, which represents a legal access, consists of seven parts. They are, from left to right, the

[1]http://www.cert.org/advisories/CA-2001-19.html
[2] http://www.cert.org/advisories/CA-2001-26.html

name of the remote host[3], the name of the user in the remote host[4], the name of the user in the identification, the day and the time of the access, the content of the request, the status code which was returned to the request, and the bytes which were sent. The content of the request consists of an HTTP command, the name of the requested file, and the version of HTTP protocol. This example shows that a user from ``kali.slab.dnj.ynu.ac.jp" tried to obtain files below ``/algodata" with an identification name ``ningen", and obtained a status code ``401" which represents a denial of identification, probably due to a misuse of a password.

Access log data exhibit several problems including chronological problem, readability problem, and diversity problem. They represent reasons for dynamic characteristics of hostile accesses, cause cognitive load to the system administrator, and necessitate complex measures to hostile accesses, respectively.

A session, which represents a sequence of requests from a user to a Web site, begins when s/he enters in a Web site and ends when s/he exits from it. We believe that handling sessions instead of log entries results in a natural analysis as well as a compact display of Web data. The problem dealt in this paper is to visualize a set of given sessions $S = \{s_1, s_2, \ldots, s_{|S|}\}$ in order to detect a set S' of hostile sessions from them $S' \subseteq S$.

We follow the conventional settings and assume that a session lasts no longer than 30 minutes. If a user issues requests to a Web site longer than 30 minutes, we group the requests by 30 minutes and consider each group as a session. A session s_i consists of $|L_i|$ log entries i.e. $s_i = \{l_{i1}, l_{i2}, \ldots, l_{i|L_i|}\}$. We model that a log entry l_{ij} consists of V attributes values $v_{ij} = \{v_{ij1}, v_{ij2}, \ldots, v_{ij|V|}\}$ and v_{ijk} takes one of W_k discrete labels $w_{k1}, w_{k2}, \ldots, w_{k|W_k|}$ where $W_k = \{w_{k1}, w_{k2}, \ldots, w_{k|W|}\}$.

3 PrototypeLines for Hostile Session Detection

[3] Here a remote host name can be replaced by its IP address

[4] A ``-" is shown when the remote host does not employ an identification service, a daemon which gives information of the user for identification. A typical host does not employ such services

3.1 Definition of Attributes

In order to perform an analysis oriented to the user side and the server side, we have classified the attributes in a log entry to three groups. They are the user group (the name of the remote host, the name of the user in the remote host, the name of the user in the identification, and the day and the time of the access), the user request group (the HTTP command, the request file, and the HTTP protocol), and the server response group (the status code which was returned to the request and the bytes which were sent). We believe that a typical fraudster conceals his identity thus decided not to employ the attributes in the user group. Thus our method visualizes the user request and the server response of each session.

For the user request group, the attribute HTTP command is considered as mandatory since it represents various requests of the user. A requested file provides valuable information for detecting hostile accesses which are related with several characteristic types of directories and files such as non-existing files. On the other hand, we have decided not to employ the HTTP protocol since we consider that it provides little information for our objective. Therefore, we employ the HTTP command and the requested file as the attributes which represent the request of the user.

For the attributes in the server group, we have decided to employ both the status code and the bytes sent. Both of the two attributes exhibit typical values for a hostile access.

3.2 Definition of Attribute Values

We assume that a label w_{kk} takes one of VD (very dangerous), D (dangerous), N (normal), S (safe), and VS (very safe). For the attributes except for the requested file, we have defined their labels as shown in Table 1 For the HTTP command, the attribute values 2, 1, 0 represent GET, HEAD, and others (POST, PUT, LINK, UNLINK, PATCH, DELETE, TRACE, OPTIONS, CONNECT, PROPFIND), respectively. While most of the values of the HTTP command are GET, HEAD is considered safe thus we have attributed safe labels to these commands. On the other hand, other HTTP commands are considered to be dangerous since they are extremely rare. For the requested file, we consider that the frequency of

requests approximately represents its reliability thus have attributed safer labels to larger numbers of requests. For the status code, 4XX (Client Error) and 2XX (Success) have been attributed VD and VS, respectively, while other codes have been attributed labels in terms of their meanings. For the bytes sent, we have attributed VD to 1 - 999 bytes since they are often related with errors while bytes larger than 1000 have been attributed S or VS since they are often related with successes.

On the other hand, the requested file attribute requires special care since it can be regarded as most important in detecting hostile accesses and it provides useful information from its character string. Thus we represent it using five subattributes shown in Table 2. The value of the requested file attribute is the label determined by the EM algorithm in the next section from its subattributes.

For the string length, the labels are decided as shown in the Table since a long file name is likely to be related with a hostile access. For the ratio of alphabets, a file name with a small ratio has been attributed a dangerous label since a request of a hostile access often contains many non-alphabetical symbols such as ``%" and ``&". For the access numbers to the file, we have attributed dangerous labels to small numbers since they represent rarely used files. For this attribute we consider a directory instead of a file since some of the files are rarely accessed. The number of child nodes concerns the hierarchical directory structure and the labels have been attributed in order to reflect the reliability of the file. The depth of the file also concerns the hierarchical directory structure and we have attributed the labels as shown in the Table since a hostile access is often related with the file just under the root directory.

3.3 Probabilistic Mixture Model

We express a session s_i with a probabilistic mixture model which consists of $|C|$ prototypes $c_1, c_2, \ldots, c_{|C|}$ where $C = \{c_1, c_2, \ldots, c_{|C|}\}$.

$$\Pr(s_i) = \sum_{m=1}^{|C|} \Pr(s_i \mid c_m; \theta) \Pr(c_m; \theta) \quad (1)$$

Here $\Pr(c_m; \theta)$ represents the probability that a prototype c_m occurs and $\Pr(s_i \mid c_m; \theta)$ is the conditional probability of s_i given c_m.

We use the EM algorithm [3] in which we employ the occurrence probability of c_m and the probability of $w_{k\kappa}$ given c_m as parameters $\theta_{c_m}, \theta_{w_{k\kappa}|c_m}$, respectively of c_m.

$$\theta_{c_m} = \Pr(c_m; \theta) \quad (2)$$

$$\theta_{w_{k\kappa}|c_m} = \Pr(w_{k\kappa} \mid c_m; \theta) \quad (3)$$

$$\theta = \{\theta_{c_m}, \theta_{w_{k\kappa}|c_m}\} \quad (4)$$

Below we show our EM algorithm.

1. Give initial values for $\Pr(c_m; \theta)$ and $\Pr(w_{k\kappa} \mid c_m; \theta)$.

2. Obtain $\Pr(c_m \mid s_i; \theta)$ by Bayes rule.

$$\Pr(c_m \mid s_i; \theta) = \frac{\Pr(s_i \mid c_m; \theta) \Pr(c_m; \theta)}{\Pr(s_i; \theta)} \quad (5)$$

where $\Pr(s_i) = \sum_{m=1}^{|C|} \Pr(s_i \mid c_m; \theta) \Pr(c_m; \theta)$

How to obtain $\Pr(s_i \mid c_m; \theta)$ depends on the probabilistic model and we use the following.

$$\Pr(s_i \mid c_m; \theta) = \prod_{j=1}^{|L_i|} \prod_{k=1}^{|V|} \Pr(v_{ijk} \mid c_m; \theta) \quad (6)$$

3. Update $\Pr(c_m; \theta)$ and $\Pr(s_i \mid c_m; \theta)$ based on the average of the posterior probabilities. For the method for the updates, we adopt the standard method as follows.

$$\hat{\theta}_{c_m} = \frac{\sum_{i=1}^{|S|} \Pr(c_m \mid s_i; \theta)}{|S|} \quad (7)$$

$$\hat{\theta}_{w_{k\kappa}|c_m} = \frac{\sum_{i=1}^{|S|} \Pr(c_m \mid s_i; \theta)\gamma}{\sum_{i=1}^{|S|} \Pr(c_m \mid s_i; \theta)} \quad (8)$$

$$\text{where } \gamma = \begin{cases} 0 & (w_{k\kappa} \neq v_{ijk}) \\ 1 & (w_{k\kappa} = v_{ijk}) \end{cases} \quad (9)$$

4. Iterate (2) and (3) until convergence.

3.4 Hue Allocation to Prototypes

In our method, a session is visualized using the prototypes obtained with the method in the previous section. Here obtaining the degree d_m of safety of a prototype c_m is not an easy problem since a prototype concerns multiple attributes. We have invented a heuristic method based on the ratio $\Pr(w_{k\kappa} \mid c_m; \theta)$ of the label $w_{k\kappa}$ which consists of a prototype.

$$d_m = \sum_{\kappa=1}^{|W_k|} \Pr(w_{k\kappa} \mid c_m; \theta)\kappa \quad (10)$$

Here recall that $w_{k\kappa}$ represents the κ-th label of an attribute k. We define w_{k1} = VD, w_{k2} = D, w_{k3} = N, w_{k4} = S, and w_{k5} = VS and represent by $\Pr(w_{k\kappa} \mid c_m; \theta)$ the conditional probability of $w_{k\kappa}$ in a prototype c_m. In (10), we multiply the number κ of the label $w_{k\kappa}$ for reflecting its safety.

In color science, warm colors such as red and orange are known to give dangerous impressions while cold colors such as blue and light blue can be considered as the opposite of warm colors. It should be noted that green is considered as the middle of warm colors and cold colors. In our method, we allocate warmer colors for prototypes with smaller d_m in order to represent the degree of danger of a session.

3.5 Visualization Display

In our PrototypeLines, we visualize obtained prototypes as shown in Figure 1. In the Figure, the upper half and lower half represent three prototypes of the user request group (REQUEST) and four prototypes of the server response group (RESPONSE) as we describe each session in terms of the two groups (cf section 3.1). A colored bar for each attribute represents a ratio graph for its labels where the correspondence of colors and labels are shown at the top of the Figure. For instance, in the leftmost prototype in the user request group, the probability of the label VS for the attribute HTTP is 100% while the probabilities of the labels VS, S, N, D for the attribute FILE are approximately 55%, 30%, 2%, 13%, respectively. A gray bar for each prototype represents the occurrence probability of the prototype. For instance, the prototype 1 occurs approximately 5%.

Figure 2 shows an example of display of sessions with our PrototypeLines dating from July 28 to August 7. Each row represents the result of the corresponding day and sessions are aligned in chronological order from left to right i.e. from 0:00 AM to 24:00 PM. In a specific day, the leftmost digits ``1" and ``2" represent the user request group (REQUEST) and the server response group (RESPONSE), respectively. A colored bar represents a visual display of a session, where the upper half and the lower half correspond to the user request group and the server response group, respectively. For a group of a session s_i, a color for a prototype c_m is attributed by the method in the previous section and the height of a colored bar is proportional to $\Pr(s_i \mid c_m; \theta)$. In this way, the user can easily recognize various kinds of information related with hostile accesses.

4 Experimental Evaluation

4.1 Obtained Probabilistic Clustering

In the experiments, we employed Web access log data to our Web site ``www.slab.dnj.ynu.ac.jp" from December 8, 2000 to October 7, 2002. In the data, the number of log entries is 205,590.

In order to decide the value for our single parameter that must be tuned i.e. the number $|C|$ of prototypes, we varied it from 3 to 9. In the process, the initial values for $\Pr(c_m; \theta)$ and $\Pr(w_{k\kappa} \mid c_m; \theta)$ are given randomly. As the EM algorithm represents a hill climbing method in terms of likelihood, we used 100 random restart and report the result that exhibits the highest likelihood.

Based on the difference of the prototypes and visual inspection, we determined the numbers of prototypes to 3 and 4 for REQUEST and RESPONSE,

respectively. The obtained prototypes are shown in Figure 1.

In the Figure, we see that the probability of the label VD for the attribute HTTP is 0, which means that a session with HTTP commands other than GET and HEAD are not reflected in the result. This is due to the nature of our approach to consider all log entries in a session in calculating the occurrence probability of a label. In this way, a label can be neglected if it is extremely rare.

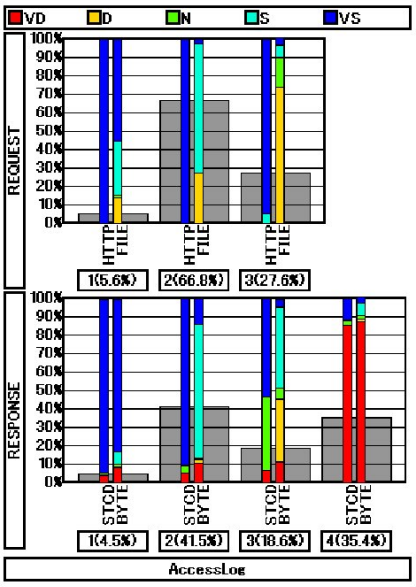

Figure 1. Obtained prototypes

Other than this problem, each prototype is considered to well represent the characteristic of the data and aligned by their degrees of danger from left to right. For the REQUEST group, the prototypes at the right-hand side are aligned by their degrees of danger of the attribute FILE. For the RESPONSE group, roughly speaking, prototype 1 represents an access which successfully reads a large file such as a csv file while prototype 2 represents an access which successfully reads a normal-sized file. Prototype 3 represents an access which fails to ascertain the update of a file with the HEAD command and prototype 4 is related with request errors.

4.2 Visualization by PrototypeLines

Figure 2 shows an example of display of sessions by our method. The period in the Figure concerns the beginning of proliferation of Code Red. We see that our PrototypeLines is effective since the display is almost red during August 5 to August 7, when we experienced an intensive attack with the virus.

We have also tried a simple alternative of our PrototypeLines to the same data and show the result in Figure 3. The result shows that the simpler version is outperformed by our method. For instance, the colors in the REQUEST group for the period of the intensive attack are red and green in Figures 2 and 3, respectively. We attribute the reason to the fact that our method considers the degree of danger more appropriately than its simple alternative.

4.3 Examples of Detected Accesses

Here we describe several examples of our discoveries from the visualization results. In our data, the virus Code Red Worm appeared in July 2001 for the first time and proliferated from August 2001. Figure 4 shows a visualization result of August 8, when we experienced an intensive attack. We see that our PrototypeLines has succeeded in visualizing the dangerous virus with warm colors.

In our data, the virus Nimda Worm appeared in June 2001 for the first time and continued to appear for a long time. Figure 5 shows a visualization result of October 30 and 31, when we experienced an intensive attack. We see that our PrototypeLines has again succeeded in visualizing the dangerous virus with warm colors.

Likewise, our PrototypeLines visualized accesses with search robots and accesses for using our server as a relay point of SPAM mails in red. The first result represents a failure since hostile accesses can be buried among such accesses. Our method mistook such accesses as hostile since a search robot tried to read ``/robots.txt", which resulted in an error since the file does not exist in our server. It should be noted that a search robot produces an error when it tries to request a file which has been deleted since its last visit. We believe that both of the problems can be fixed with exception handling methods. The second result represents a success and can be considered to be useful in issuing an alert to such an access, which exists in our data in various forms from August 2001.

It should be noted that our method has been successful in emphasizing hostile accesses with warm colors though it does not employ access patterns and rules which are specific to known viruses. We

consider that the fact shows effectiveness of our PrototypeLines for detecting novel types of hostile accesses.

5 Conclusion

In this paper, we proposed a visualization method based on probabilistic clustering for detecting hostile accesses to a web site. Web access log data are huge, dynamic, difficult to read, and rich in diversity. We believe that our visualization method serves as an effective tool for a system administrator for monitoring log data and detecting novel types of attacks. We have modified our PrototypeLines, which we proposed in [9], to visualize Web sessions in order to fulfill the objective.

Experimental results show that our PrototypeLines is promising and outperforms its alternative. Our future work includes investigation of representation of a session with its log entries and application of our approach to other kinds of fraud detection problems.

Acknowledgements

This work was partially supported by the grant-in-aid for scientific research on fundamental research (B) 16300042 and priority area ``Active Mining'' from the Japanese Ministry of Education, Culture, Sports, Science and Technology.

References:

[1] S. K. Card, J. D. Makinlay, and B. Shneiderman (eds.), *Readings in Information Visualization,* Morgan Kaufmann, San Francisco, 1999.

[2] P. K. Chan and S. J. Stolfo, Toward Scalable Learning with Non-Uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection, *Proc. Fourth Int'l Conf. on Knowledge Discovery and Data Mining (KDD),* 1998, pp. 164-168.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin, Maximum Likelihood from Incomplete Data via the EM Algorithm, *Journal of the Royal Statistical Society B,* Vol.39, No.1, 1977, pp. 1-38.

[4] W. DuMouchel and M. Schonlau, A Fast Computer Intrusion Detection Algorithm Based on Hypothesis Testing of Command Transition Probabilities, *Proc. Fourth Int'l Conf. on Knowledge Discovery and Data Mining (KDD),* 1998, pp. 189-193.

[5] T. Fawcett and F. Provost, Adaptive Fraud Detection, *Data Mining and Knowledge Discovery,* Vol.1, No.3, 1997, pp. 291-316.

[6] U. Fayyad, G. G. Grinstein and A. Wierse (eds.), *Information Visualization in Data Mining and Knowledge Discovery,* Morgan Kaufmann, San Francisco, 2002.

[7] W. Lee, S. J. Stolfo, and K. W. Mok, Mining Audit Data to Build Intrusion Detection Models, *Proc. Fourth Int'l Conf. on Knowledge Discovery and Data Mining (KDD),* 1998, pp.66-72.

[8] M. Narahashi and E. Suzuki, Detecting Hostile Accesses through Incremental Subspace Clustering, *Proc. 2003 IEEE/WIC International Conference on Web Intelligence (WI),* 2003, pp.337-343.

[9] E. Suzuki, T. Watanabe, H. Yokoi, and K. Takabayashi, Detecting Interesting Exceptions from Medical Test Data with Visual Summarization, *Proc. Third IEEE International Conference on Data Mining (ICDM),* 2003, pp. 315-322.

[10] K. Yamanishi et al., On-line Unsupervised Outlier Detection using Finite Mixtures with Discounting Learning Algorithms, *Proc. Sixth ACM Int'l Conf. on Knowledge Discovery and Data Mining (KDD),* 2000, pp.320-324.

Table 1. Attribute labels except for the requested file, where VD, D, N, S, VS, and nv represent very dangerous, dangerous, normal, safe, very safe, and ``no value".

Attribute (Abbreviation)	Label				
	VD	D	N	S	VS
HTTP command (HTTP)	nv, 0			1	2
Requested file (FILE)	1	2 - 100	101 - 1000	1001 – 2499	2500 -
Status code (STCD)	4XX	5XX	3XX	1XX	2XX
Bytes sent (BYTE)	1 - 999	nv	0	1000 – 10000	10001 –

Table 2. Subattributes and labels for their values for the requested file.

Attribute (Abbreviation)	Label				
	VD	D	N	S	VS
String length (LENG)	300 -	50 - 299	15 - 49	7 - 14	0 - 6
Ratio of alphabets (CHAR)	0 - 49	50 - 59	60 - 69	70 - 89	90 - 100
Numbers to the directory (ACCS)	0	1 - 9	10 - 99	100 - 999	1000 -
Number of child nodes (CHLD)	0	1 - 5	6 - 10	11 - 100	101 -
Depth of the file (LEVL)	1	2 - 4	5 - 7	8 - 10	11 -

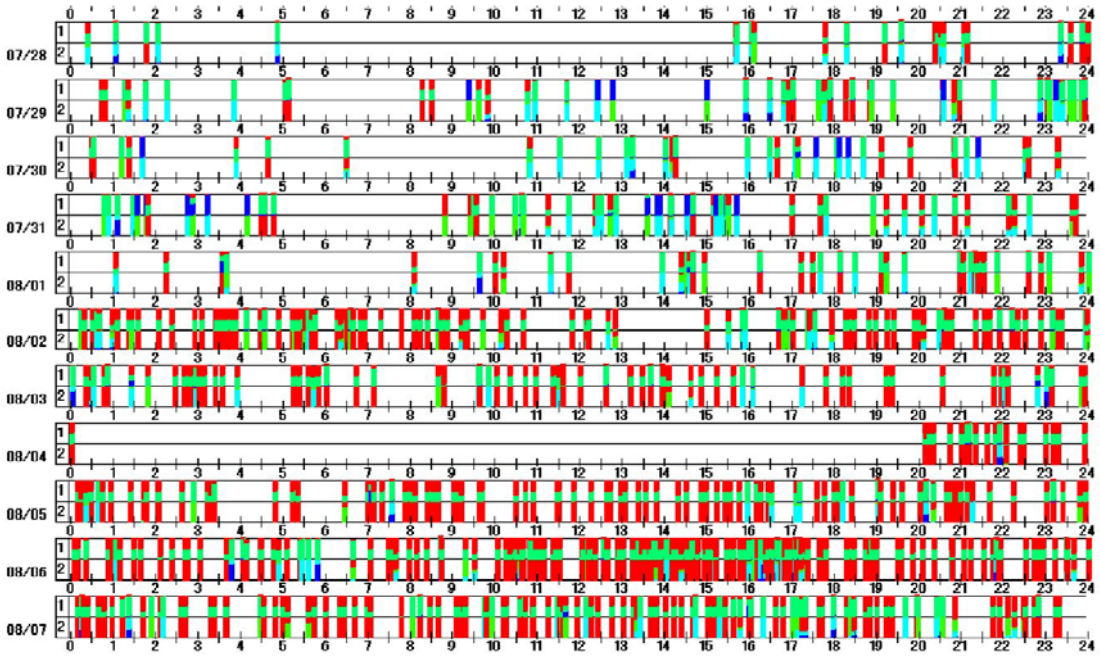

Figure 2. Example of display with PrototypeLines

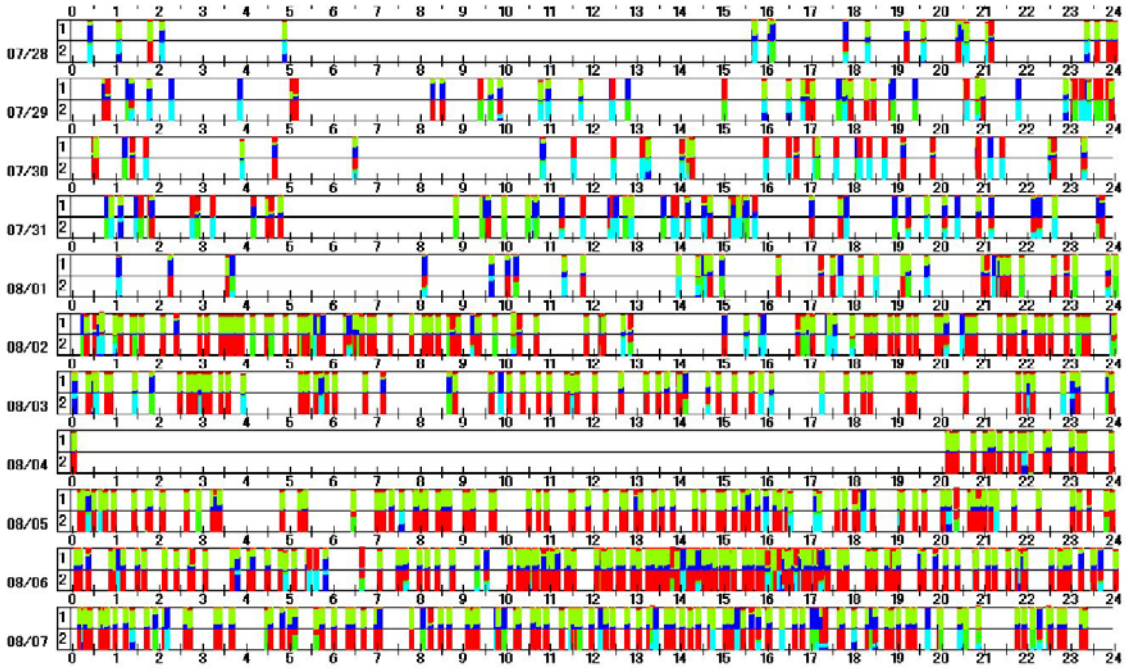

Figure 3. Example of display with a simpler version of PrototypeLines

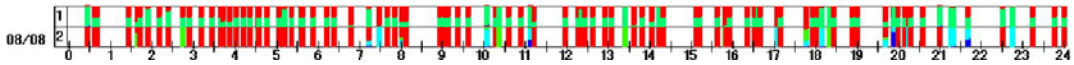

Figure 4. Example of display with PrototypeLines for a period which contains many accesses with Code Red Worm

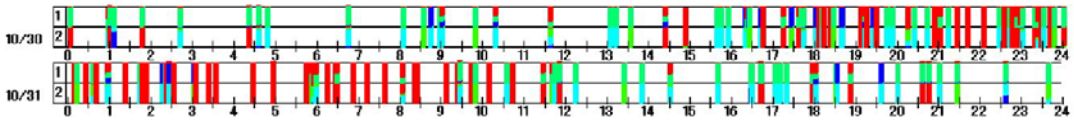

Figure 5. Example of display with PrototypeLines for a period which contains many accesses with Nimda Worm