ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 2009. No. 4(92)
ELEKTRONIKA IR ELEKTROTECHNIKA

TELECOMMUNICATIONS ENGINEERING
T180
TELEKOMUNIKACIJŲ INŽINERIJA

Genetic Programming Based Network Traffic-Profiling System

M. Ekmanis
Faculty of Electronics and Telecommunication, Riga Technical University,
Riga, Azenes str. 12-137, LV-1048, phone: +371 29119856, e-mail: Martins.Ekmanis@rtu.lv

Introduction

There are two generally accepted categories of intrusion detection techniques: misuse detection and anomaly detection. Misuse detection refers to techniques that characterize known methods to penetrate a system. Anomaly detection refers to techniques that define and characterize normal or acceptable behaviors of the system. Behaviors that deviate from the expected normal behavior are considered intrusions. The pattern/signature might be a static string or a set sequence of actions.

I propose to use the finite state machine (FSM) as a model of traffic source behavior descriptor. Model instances are automatically generated by genetic algorithm (GA) allowing to group similar sources and describe their role in the network. This model allows handling event sequence with internal loops. Most of GA and genetic programming (GP) solutions are not capable of handling loops and multiple host interaction in solution domain.

This work is focused on the TCP/IP network protocols. The paper is organized into the following modules:

Section 1 is introduction;

Section 2 describes traffic structure in the session level and underscores the most important attributes in it;

Section 3 provides background information on FSM and GA;

Section 4 discusses implementation of the algorithm;

Section 5 presets experimental results;

Finally, section 6 makes some concluding remarks.

Traffic structure

If we look on the actual network infrastructure and data flows, the only sensible data source for the analysis is netflow export. These vectors correspond to the session layer data, describing interhost communications. Usually information payload is not exported or analyzed to avoid specific analysis of huge amount information as well private information. The usage of compression and data encryption in the protocol layer makes the content analysis more difficult or even impossible.

Most of the traffic flows are mutually related. Some relations are defined in the protocol level such as FTP control channel what uses different port numbers than the data channel. Some protocols supplement each other like POP3 before SMTP as authorization of electronic mail gateway.

If we look on causally connected events (Fig.1.), we can distinct several types of possible relations (Table 1.) [1].

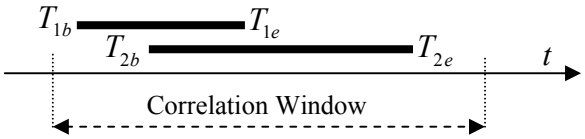

Fig. 1. Event correlation window

The event correlation window (Fig. 1) describes two important events – the beginning of the flow (T_b) and the end of flow (T_e). Any correlated events must incorporate in the determinate correlation window. The correlation window can be defined as time window or event count window [2].

Table 1. Temporal relations

Relation	Name
$T_{1b} \geq T_{2e}$	Follow
$T_{1b} \geq T_{2b}$ & $T_{2b} \geq T_{1b}$ & $T_{1e} \geq T_{2e}$	After
$T_{1b} \geq T_{2b}$ & $T_{1e} \leq T_{2e}$	During
$T_{1b} = T_{2b}$	Start
$T_{1b} \leq T_{2b}$ & $T_{1e} \geq T_{2b}$ & $T_{2e} \geq T_{1e}$	Before
$T_{1e} \leq T_{2b}$	Precede

Casually connected events have significant correlations [3]. For example, HTTP and DNS - usually the host clarifies the target address before establishment of a new connection (Fig. 2).

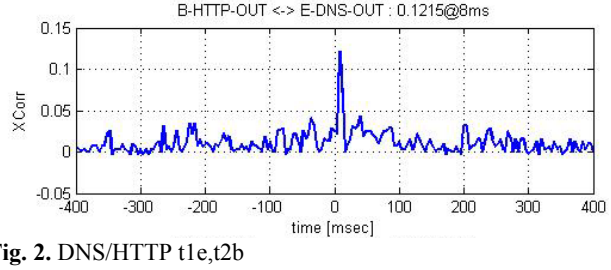

Fig. 2. DNS/HTTP t1e,t2b

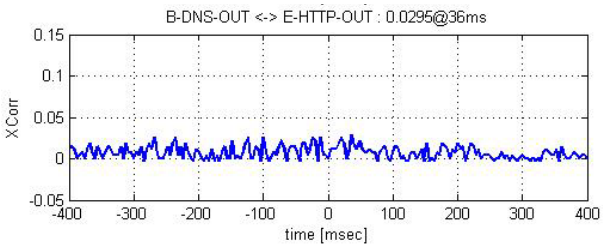

Fig. 3. DNS and HTTP request correlation

Unfortunately in heavy loaded systems the response time diffuse and the correlations are not visible. In this matter it is useful to explore other mechanisms to discover and model the causally connected events.

FSM and GA models

Several publications [4] show finite state machine (FSM) as a way to describe traffic behavior.

As an FSM input alphabet Σ is used data vectors F(h), anent to host h. S is finite, no-empty set of states where s_0 is initial state. $\delta_{x,y}(F)=\{true, false\}$ is the state transition functions from the state x to the state y.

Each traffic source (host) can operate with several parallel processes and each of them can perform different algorithm. Taking into account this moment I will extend FSM by allowing more than one state at the same time. Each input vector is verified against any active state $s=\{s_i^1, s_i^2, ..., s_i^n\}$, transferring it to the next state if state transition function $\delta(F)=true$. When the last state is activated, the algorithm is considered as finished and all involved flows are selected.

If the current input does not change any of existing instances, F is verified against initial state transaction functions $\delta_{0,y}(F)$ and in case of success, the new state y is added to the active state list.

Most of traffic sources have very complicated algorithms involving different external events and data. The size of search space is huge and can't handle by traditional search techniques. Artificial Intelligence (AI) techniques can be used for behavior classification and data reduction.

GA is the general search method, which uses analogies from natural selection and evolution. It is a search technique to find approximate solution and requires two things to be defined: a genetic representation of the solution domain and a fitness function to evaluate the solution domain. Other publications show successful usage of GA in IDS and IPS solutions [5–8].

GA algorithm can be divided into a number of sequential steps:

1. Create a random population of rules.
2. Evaluate each rule by assigning a fitness value according to a fitness function, which can measure the capability of the rule to solve the problem.
3. Generate the new population using reproduction with crossover, mutation, or other operators from a randomly chosen set of parents.
4. Repeat steps 2 onwards for the new population until a predefined termination criterion has been satisfied, or a fixed number of generations have been completed.
5. The solution to the problem is the final population.

Different authors use different fitness functions, but most of them have similar structure (1). Frequently is used the sum of attributes $a_1,...,a_n$ with different weights $w_1,...,w_n$, where attributes represent some aspects of solution (simplicity, efficiency, accuracy).

$$fit = w_1 a_1 + ... + w_n a_n . \quad (1)$$

The transformation 1/(1+a) are also frequently used to make the higher assessment for individuals with small attribute values.

Implementation

First of all captured flows are converted to stream of events (Fig. 4).

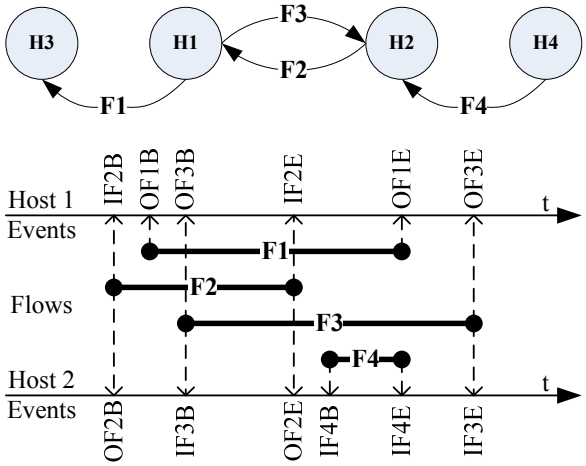

Fig. 4. Flows to connection events transformation

FSM is used as a representation of solution domain. Transition function matrix (2) is encoded in chromosome. Each chromosome is square matrix consisting of $n \times n$ elements, where n is FSM maximum state count.

$$M(s,F)=\begin{bmatrix} \delta_{0,0}(F) & ... & \delta_{0,n}(F) \\ ... & ... & ... \\ \delta_{n,0}(F) & ... & \delta_{n,n}(F) \end{bmatrix}. \quad (2)$$

Each row j represents one state s_j and its possible transitions $\delta_{ji}(F)$ to other states, where $i \in [1,n]$. The first and the last state have special meaning, namely the initial and the final state. As a result, the first column and the last row actually are not used, as there is no way out of finish state and no way back to initial state.

As an FSM input alphabet is used flow vectors (3).

$$F = \{a_{host}, a_{dir}, a_{prot}, a_{port}, a_{event}\}, \quad (3)$$

where a_{host} – the distant IP address; $a_{dir}=\{in, out\}$ – the connection direction anent to host under research; a_{prot} –

the protocol identification number; a_{port} – the service port number if applicable; $a_{event} = \{begin, end\}$ – the event type.

According to (3) the state transition function $\delta_{x,y}(F) = \{true, false\}$ is trigger – it implements generalization (any host, port above 1024, disabled transition).

The value of all populations (4) is used to make decision when to stop evolution.

$$fit = \frac{\sum c_{select}}{c_{total}}. \quad (4)$$

The fitness for each individual is calculated as (5):

$$M_{fit} = w_0 Tr + w_1 Ti + w_2 Lo + w_3 Pe, \quad (5)$$

where Tr (traffic) – the sum of all traffic involved in successfully finished M transactions; Ti (time) – the sum of time covered by all transaction instances of M; Lo (loops) – the total number of successful M transactions; Pe (penalty) – calculated according stability of the solution. Each transition function $\delta_{x,y}(F)$ with low usage rate characterize possibility of noise or unusual case so it receives "-1" point.

The goal of this GA solution is to cover as much as possible traffic with in limited time and resources. To reach that, I included diffusion element (6) in weights. If the begin or the end event is caused by other events, whole flows are considered as casually connected. If two or more chromosomes select the same flow vector, wins the one with higher fitness value in the previous generation. Fs is the number of selected flows, Fw is the number of winning flows.

$$w_{i=[0,3]} = Di \cdot w_i^{'}, \quad (6)$$

where $Di = \frac{Fw}{Fs}$.

The number of chromosomes of population is increased as old ones have found their maximum. If increasing count does not increase overall fitness (4), process is stopped.

Discovered structures are user readable and describe event sequences from the start state S_0 to finish state S_F (Fig. 5)

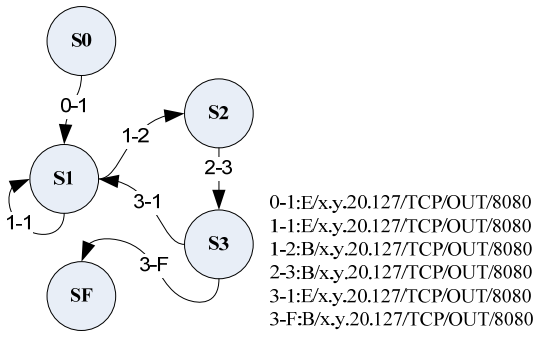

Fig. 5. Web proxy client – FSM model

Part off the traffic cannot be classified using this mechanism, since it consists of irregular structures, single connections, corrupted connections (noise) etc.

Experimental results

I have implemented algorithm in c#. The netflow data are exported from corporate network core router through netflow v5 protocol. In the next step samples was aggregated to bidirectional data flows. The beginning and the end events were sorted chronologic and applied to each host involved in this connection (Fig. 4). Genetic algorithm is based on multithreading for faster computation and hash tables for faster search.

I have used several training traces each of them containing of 1h traffic data (~3.7 million flows/8156 active hosts). Weights in experiment was set to $w_0 = 1$, $w_1 = 1$, $w_2 = 100$, $w_3 = 100$. Crossover, mutation, dropping operators was used proportional. The time window of 1sec was used as a correlation window.

Evolution process (Fig. 6) stops when 65 to 80% flows are assigned to unique groups (chromosomes).

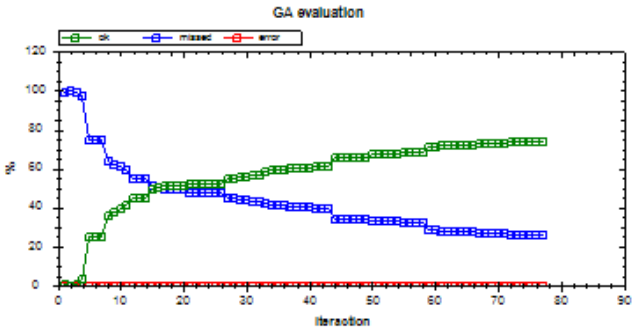

Fig. 6. GA evalution

Increasing population size overall result does not increase since a new chromosome was not able to found any free local maximum. Maximum state count changes readability of solutions. Three state models are trivial and adapt to any traffic very fast, but are not able to describe useful dependencies.

Ten or more sate models increases overtraining effect (Fig. 7) and computation capacity decreasing the total results. Optimal results were obtained with five or six state models.

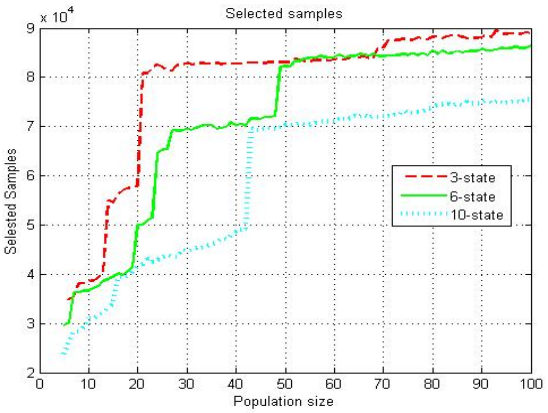

Fig. 7. Selection rate dependency from population size

The experimental results discovered several important anomalies in analyzed traffic traces: Remotely controlled group of zombie computers what are controlled using IRC channel (part of "botnet"); Virus infected hosts sending out

spam; Unauthorized proxy services; Active interaction of the group of users connected via the same servers (the most popular social networking server pool in Latvia).

Conclusions

In this paper, I have presented a GA-based approach for traffic profiling. FSM is used as model with internal loops to describe traffic source behavior. The proof of concept implementation shows potential capability to detect network anomalies and sources of unwanted traffic. This algorithm work in general terms, without requiring any a priori knowledge about a particular type of applications.

Host attribute allow discovering important peers in communication (servers, hubs, command and control hosts, trackers etc.). Some part of traffic is unstable and can't be classified. Still 65-80% of traffic has strong structure with loops and it is enough to identify services and host profiles.

Each traffic source belongs to one or more extracted groups providing automatic grouping functionality. This feature can be used as a behavior metric for future research in this field.

Acknowledgment

This work has been partly supported by the European Social Fund within the National Programme "Support for the carrying out doctoral study programm's and post-doctoral researches" project "Support for the development of doctoral studies at Riga Technical University.

References

1. **Aboelela E., Douligeris C.** Fuzzy Temporal Reasoning Model for Event Correlation in Network Management // Proceedings of the 24th Annual IEEE Conference on Local Computer Networks. – 1999. – P. 150 .
2. **Onut I. V., Ghorbani A. A.** Feature Classification Scheme for Network Intrusion Detection // International Journal of Network Security. – 2007. – Vol. 5. – No. 1. – P. 1–25.
3. **Hlavackova–Schindler K., Palus M., Vejmelka M., Bhattacharya J.** Causality detection based on information–theoretic approaches in time series analysis // Physics Reports. – 2007. – Vol. 441, Issue 1. – P. 1–46.
4. **Kannan J., Jung J., Paxson V., Koksal C.** Semi–Automated Discovery of Application Session Structure // Proceedings of ACM Internet Measurement Conference. – 2006. – P. 119–132.
5. **Lu W, Traore I**. Detecting new forms of network intrusion using genetic programming // The 2003 Congress on "Evolutionary Computation". – 2003. – Vol. 3. – P. 2165–2172
6. **Li W.** Using Genetic Algorithm for Network Intrusion Detection // Proceedings of the United States Department of Energy Cyber Security Group. – 2004
7. **Mukkamala S., Sung A. H., Abraham A.** Modeling Intrusion Detection Systems Using Linear Genetic Programming Approach // Proceedings of the 17th international conference on Innovations in applied artificial intelligence. – 2004. – P. 633–642.
8. **Zhang W., Wu Zhi–ming., Yang Gen–ke.** Genetic programming–based chaotic time series modeling // Journal of Zhejiang University Science. – 2007. – Vol. 5. – No. 11. – P. 1432–1439.

Received 2009 02 15

M. Ekmanis. Genetic Programming Based Network Traffic-Profiling System // Electronics and Electrical Engineering. – Kaunas: Technologija, 2009. – No. 4(92). – P. 49–52.

A new approach is proposed for unattended grouping of traffic sources. This is GA based search technique, which use finite state machine as a genetic representation of solution domain. Models approximate multi host communication in relation to host under research. They describe traffic source behavior handling event sequence with internal loops and provide similar traffic source grouping. The proof of concept implementation shows potential capability to detect network anomalies and sources of unwanted traffic. Ill. 7, bibl. 8 (in English; summaries in English, Russian and Lithuanian).

М. Екманис. Система идентификации сетевого трафика путем генетического программирования // Электроника и электротехника. – Каунас: Технология, 2009. – № 4(92). – С. 49–52.

Предложен новый отличительный метод группировки безнадзорных источников сетевого трафика. Этот поисковый метод основан на использовании генетического алгаритма. При решении был применен метод конечных автоматов. Модель аппроксимирует связи многих источников по отношению к исследуемому источнику. Модель описывает поведение источников сетевого трафика, в том числе содержащих внутренние циклы, и обеспечивает группировку схожих источников. Доказательством концепции служит потенциальная способность алгоритма идентифицировать нежелательные аномалии сетевого трафика и источников. Ил. 7, библ. 8 (на английском языке; рефераты на английском, русском и литовском яз.).

M. Ekmanis. Duomenų tinklų identifikavimo sistemos tyrimas taikant genetinį algoritmą // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2009. – Nr. 4(92). – P. 49–52.

Pasiūlytas naujas srautų šaltinių grupavimo metodas. Tai GA paieškos technologija, paremta baigtine sistemos būsena ir genetiniu algoritmu. Modeliai aproksimuojami daugialinijiniu priėmimo ryšiu. Aprašoma šaltinių ir duomenų srautų būsena, įskaitant tuos, kurių sudėtyje yra vidaus ciklų, pateikiami šaltinių grupavimo metodai. Koncepcijos teisingumą įrodo potencialus gebėjimas aptikti tinklo anomalijas, nepageidaujamas duomenų srautuose ir šaltiniuose. Il. 7, bibl. 8 (anglų kalba; santraukos anglų, rusų ir lietuvių k.).