Cloud Network Security Monitoring and Response System

Murat Mukhtarov
Information Security Faculty
National Research Nuclear
University MEPhI
Moscow, Russia
Muhtarov.mr@gmail.com

Natalia Miloslavskaya
Information Security Faculty
National Research Nuclear
University MEPhI
Moscow, Russia
NGMiloslavskaya@mephi.ru

Alexander Tolstoy
Information Security Faculty
National Research Nuclear
University MEPhI
Moscow, Russia
AITolstoj@mephi.ru

***Abstract* — The public clouds network monitoring and response system, based on flow measurements, open source tools and CSMS (Cloud Security Monitoring System) module, is to be introduced in this paper. The main goal of the research is to develop an algorithm and to implement a system, which automatically detects and makes a response to network anomalies, occurring inside a Cloud infrastructure. In this research is proposed approach of anomaly detection inside the Cloud infrastructure which is based on a profiling method of IPFIX (IP Flow Information Export) protocol data and idea of negative selection principle is used for generating signatures of network anomalies, which are named detectors. The automatic response module makes a decision about network anomalies origin, based on several iterative checks and creates a record on the firewall rules table. The network traffic profiling process automatically generates the firewall rules set for all traffic classes, obtained during the learning process. Main results of the research are development of the algorithms and the way of the monitoring network attacks inside the Cloud. Implementation of the algorithms is python-based script and currently stays under hard-testing phase.**

Keywords - Cloud computing; Cloud infrastructure; Virtual Infrastructure; Application Hosting; Network Security.

I. INTRODUCTION

Cloud computing is a novel way to provide customers with Information Technology services, but with virtualization technologies in the background. Cloud computing uses networked infrastructure; software and computing power to provide resources to customers in an on-demand environment. With cloud computing, information is stored remotely in a centralized server farm and is accessed by the hardware or software thin clients that can include desktop computers, notebooks, handhelds and other devices. Typically, Clouds utilize a set of virtualized computers that enable users to start and stop servers or use compute cycles only when needed (also referred to as utility computing) [1]. In terms of information security, the cloud computing threat model consists of three fundamental issues: availability, integrity and confidentiality violations. Availability is terminated via Denial of Service -attacks. The likelihood and easiness of these attacks will increase as the volume of information exchanged between a user and a cloud provider increases. Integrity issues arise due to the fact that users must be sure that the information they retrieve is the same as that they store. This is a difficult task for one reason: information changes over time as do users themselves. But, also, it is important to separate users' information from the production information (for example configuration files, system files integrity, and so on). Finally, confidentiality issues may take place, for example over (accidental) disclosure of information to third parties or because of aggregation. Most computer compromises result in information leakage, so this is also an important issue [2].

In this research, we focus on availability as a main issue and the other issues that arise from it, so they are subsidiary risks for us. One possible way for Cloud networks to be monitored is to use a network telemetry principal with such protocols as Cisco Netflow [3] or IPFIX (Internet Protocol Information Export) [4]. Design of the open source virtualization technologies provides an opportunity to use Netflow/IPFIX probes on a hypervisor without performance reduction. IPFIX protocol has some advantages while being compared with the Netflow; it is not proprietary, it is open-standard and has improvements [5] that can be used in open source systems such as Linux or BSD (Berkley Source Distribution) -derivate systems. IPFIX is a lightweight network monitoring protocol for the connection control and volume-based traffic estimation [6]. Here we propose an approach to profile IPFIX data in such environment as a Cloud infrastructure and also suggest ways to make an automatic response to the detected anomalies inside a network. The way described in the paper is applicable to the Cloud solutions that provide their customers with such services as Infrastructure as a Service, Platform as a Service. In other words a Cloud Infrastructure consists of large amount of virtual machines running inside virtual infrastructure based on physical servers and network equipment.

II. STATE OF ART AND RELATED WORKS

The main focus of the paper is a network security monitoring approach in a Cloud infrastructure. We discuss some network security threats and issues that may occur in the virtual infrastructure clouds. All of them use shared hardware, network [1] and hypervisor's resources [2].

Security threats related to hosting application in a Cloud Infrastructure are covered by Molnar and Schechter [7]. The researches compare traditional and cloud hosting focused on information security threats.

The Virtual Local Area Network (VLAN) separation technique on a Cloud Infrastructure is mentioned by Berger’s et al. [8]. They suggest a way of increasing virtual infrastructure security by using a strong security policy inside a cloud infrastructure – Trusted virtual data center (TVDC). Their idea is based on the research of Bussani, et al., Trusted Virtual Domains (TVD): Secure Foundations for Business and Information Technology Services [9]. The main idea of TVDC is a strong isolation and integrity guarantee in virtualized, cloud computing environments [8]. To achieve this isolation researchers use network separation techniques based on IEEE 802.1q [10], memory control techniques and “colorizing” each data flow inside a cloud. Another approach to a Cloud infrastructure monitoring called “Private Clouds MONitoring Systems” (PCMONS) was created by Chaves, et al. [11]. Their main goal was to develop a modular and extensible monitoring system for the private Clouds. PCMONS is implemented as a module for the open source monitoring system Nagios and is compatible with the open source IaaS platform Eucalyptus [11]. But, it has several disadvantages: as PCMONS is a Nagios module, it inherited Nagios performance and scalability issues that eliminate applicability to the huge Cloud infrastructure; also it is compatible only with one solution. The described system monitoring approach is focused on network security monitoring and response actions inside a Cloud. The main advantages of the CSMS approach are compatibility with the majority of operating systems and network equipment due to IPFIX protocol, ability of an automatic response to a network attack and ability of identifying unknown network anomalies in some cases.

III. PROFILING NETWORK TRAFFIC DATA

To monitor the network traffic anomalies, that in fact are the result of DDoS-attacks or abuse traffic, we have to find a way that will be applicable to the implementation inside a network of a Cloud Infrastructure.

We worked out several requirements to this approach:

1) To be informative enough to analyze network traffic volumes by traffic types;

2) To be lightweight;

3) To be easy to spread through a Cloud infrastructure network and

4) To not impact production network performance.

The best way that satisfies all these requirements is to use flow-based measurement protocols like Netflow or IPFIX [6]. Here, we use IPFIX, because it is an open standard protocol.

To profile IPFIX data, we use a maximum entropy estimation approach, introduced in [12] and [13]. We have to modify and improve an algorithm of profile estimation to make it applicable to IPFIX data analysis (Fig. 1). For designing an algorithm we have to classify a given pattern of network traffic. Network traffic classification process is needed because traffic patterns usually consist of large amount of the different traffic packets and storing profile of raw traffic data will require large amount of disk space. Therefore, large volumes of data will require more processor time for processing. So, we propose to use preprocessing classification algorithm, which allows us to work with volume-based estimation of network traffic data, which is divided by classes. Result of preprocessing is a significant reduction of the size of data which should be processed by monitoring system. Amount of traffic classes should be selected by user. Also, an expert should exclude “anomalies” if they are present in a given pattern.

This algorithm checks in a cycle each traffic class with maximum entropy approach and estimates weights of each traffic class in a model. The algorithm’s result is the network traffic profile in which only the most significant traffic classes in a given pattern are stored.

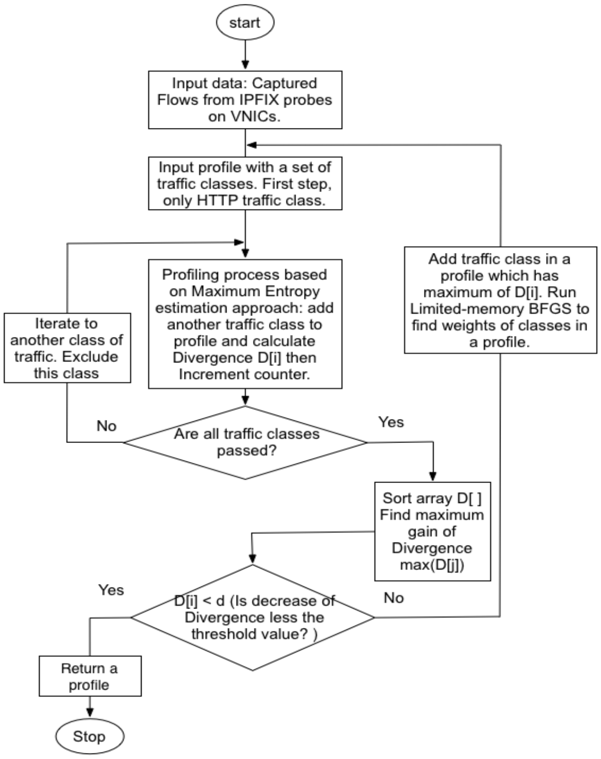
start
Input data: Captured Flows from IPFIX probes on VNICs.
Input profile with a set of traffic classes. First step, only HTTP traffic class.
Iterate to another class of traffic. Exclude this class
Profiling process based on Maximum Entropy estimation approach: add another traffic class to profile and calculate Divergence D[i] then Increment counter.
Add traffic class in a profile which has maximum of D[i]. Run Limited-memory BFGS to find weights of classes in a profile.
No
Are all traffic classes passed?
Yes
Sort array D[ ] Find maximum gain of Divergence max(D[j])
Yes
D[i] < d (Is decrease of Divergence less the threshold value?)
No
Return a profile
Stop

Figure 1. Block diagram of IPFIX data profiling.

The algorithm stops when the next traffic class does not improve the profile enough, in other words the decrease of divergence should be less then threshold value.

To adapt the algorithm to a Cloud Infrastructure network, we propose to make some special traffic classes that are inherent to the network of a Cloud infrastructure. We placed HTTP, HTTPS, DNS, SMTP, POP3, IMAP, POP3S, IMAPS, RDP, VNC, SQL ports in the separate classes. Also, we modified greedy algorithm to make it easier for implementation. We propose to exclude network traffic class from sampling process, after cycle pre-check with a given pattern. This improvement allows avoiding additional checks of the network traffic patterns due to IPFIX protocol input data format.

IV. ANOMALY DETECTORS GENERATION ALGORITHM

Another approach proposed in this paper is a special way of generating a network anomaly detector. The idea of this approach lays in a negative selection algorithm, introduced by Forrest et al. [14]. According to the negative selection, a network traffic profile, which is returned by the IPFIX data-profiling algorithm (normal behavior profile), could be modified in the manner proposed below. To create a set of potential anomalies detectors, we increased the volumes of the traffic classes in a normal behavior profile with the random values in the range of Lower_Tr and Upper_Tr variables. Also there are several settings for the detector generating process: the amount of detectors needed, the amount of affected positions in a profile and a threshold value of divergence. The block diagram of the algorithm is shown on the Fig. 2.

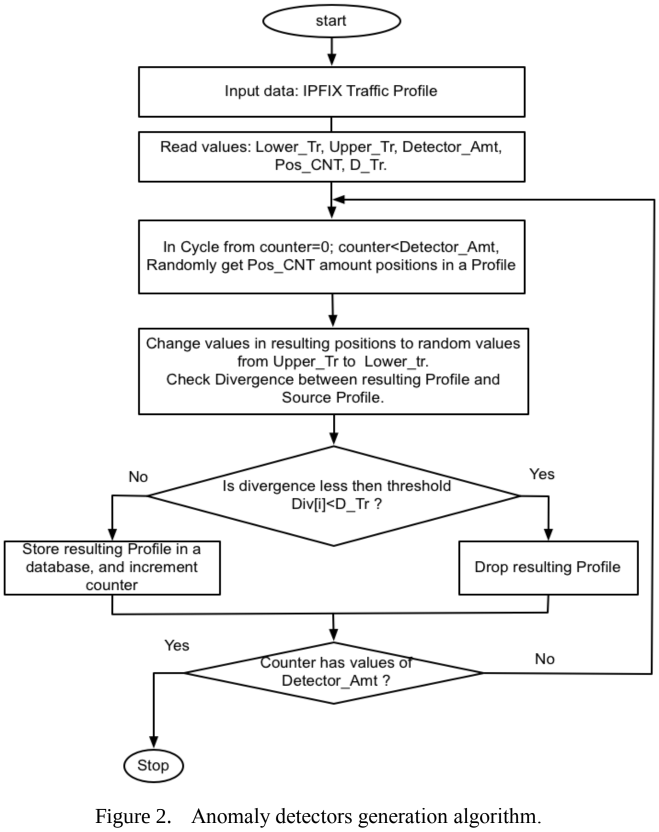
start
Input data: IPFIX Traffic Profile
Read values: Lower_Tr, Upper_Tr, Detector_Amt, Pos_CNT, D_Tr.
In Cycle from counter=0; counter<Detector_Amt, Randomly get Pos_CNT amount positions in a Profile
Change values in resulting positions to random values from Upper_Tr to Lower_tr. Check Divergence between resulting Profile and Source Profile.
No
Is divergence less then threshold Div[i]<D_Tr ?
Yes
Store resulting Profile in a database, and increment counter
Drop resulting Profile
Yes
Counter has values of Detector_Amt ?
No
Stop

Figure 2. Anomaly detectors generation algorithm.

The algorithm randomly changes values of positions of the network traffic classes inside a profile according to the values of the Lower_Tr and Upper_Tr. The stopping criteria is an achievement of the required number of anomaly detectors. Detectors that are similar to the normal behavior profile should be dropped. All other detectors should be stored inside the database.

V. ANOMALY DETECTION AND RESPONSE INSIDE CLOUD INFRASTRUCTURE

Anomaly detection is based on the set of detectors, recorded in the database. Fig. 3 shows a detector life cycle, called "maturing" while comparing it with an immune system.

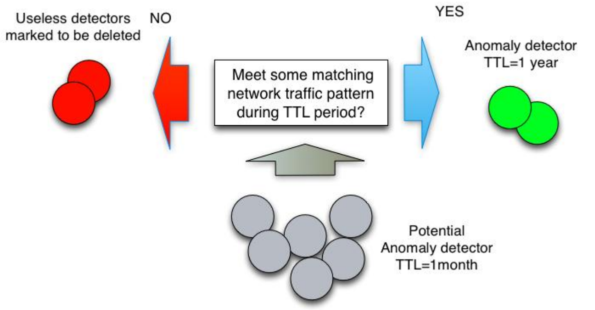
Useless detectors marked to be deleted
NO
YES
Anomaly detector TTL=1 year
Meet some matching network traffic pattern during TTL period?
Potential Anomaly detector TTL=1month

Figure 3. Anomaly detectors maturing process.

The probes are the exporting flows of IPFIX data to the collectors. A collector consists of two parts: the first one normalizes incoming data from the probes, and the second one performs a comparison of a captured traffic pattern with an anomaly detector. Each anomaly detector has "time-to-live" (TTL) attribute. Normally we set its values as one month. If a detector never matches any of the captured traffic patterns within one month it will be marked for deletion and it would be dropped at the end of the next month. One month period seems reasonable to reduce impact on the monitoring system performance and limit number of detector, but depends on user settings. Deletion of the detector does not mean that network anomaly which should be covered with deleted detector would not be handled properly. Generating of the anomaly detectors is a pseudorandom process which allows the possibility of the collisions. So such kind of anomaly possible could be found with detectors from another generation with some probability.

Another case is when a detector matched some of the network traffic pattern. This detector changes its TTL ("time-to-live") attribute to one year and spreads it across all probes. Hence, we could clean our detectors database from the patterns that we will never observe in the network traffic and collect patterns that are really useful for anomaly detection.

Fig. 4 introduces our algorithm of anomaly detection and response actions. IPFIX information has several attributes referred to the IP packet header data. When a network anomaly is detected, a Cloud monitoring system could tell us what kind of traffic causes an anomaly. In this case, we could find out a source IP address of anomaly traffic and block it inside a firewall.

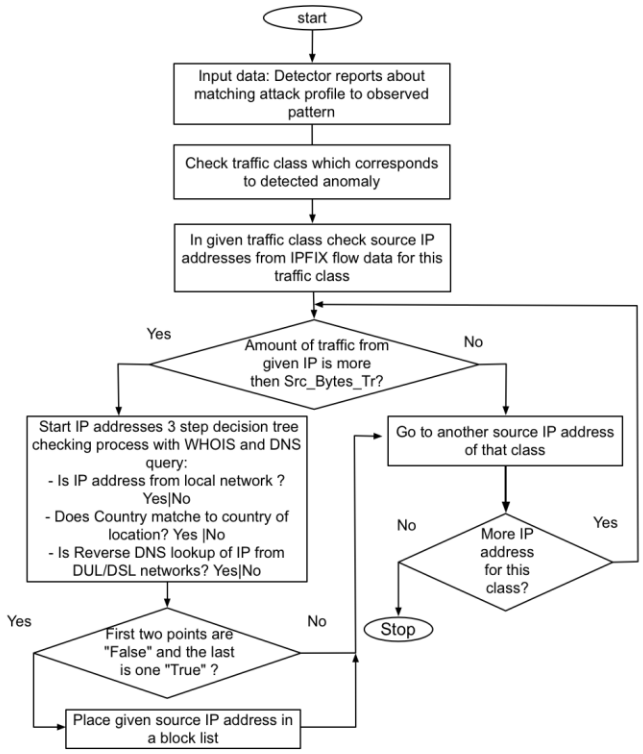
start
Input data: Detector reports about matching attack profile to observed pattern
Check traffic class which corresponds to detected anomaly
In given traffic class check source IP addresses from IPFIX flow data for this traffic class
Amount of traffic from given IP is more then Src_Bytes_Tr?
Yes
No
Start IP addresses 3 step decision tree checking process with WHOIS and DNS query:
- Is IP address from local network ? Yes|No
- Does Country matche to country of location? Yes |No
- Is Reverse DNS lookup of IP from DUL/DSL networks? Yes|No
Go to another source IP address of that class
More IP address for this class?
No
Yes
Stop
First two points are "False" and the last is one "True" ?
Yes
No
Place given source IP address in a block list

Figure 4. Block diagram of monitoring and response process.

To increase accuracy of the blocking system and to preserve a normal traffic we use "Whois" Database queries to learn an origin of IP address, location and reverse DNS queries to estimate the purpose of IP address usage. We perform several checks: "*Is the IP address from our network?", "Is the IP address from our Country?", "Is the IP address from Digital Subscriber Line (DSL)/Dial Up Line (DUL) network?".* So, if the IP address is outside of a Cloud network and region, we propose that it probably could be the reason for an anomaly and we block it for an hour. We perform the same action if the IP address is from DSL/DUL networks.

VI. INTEGRATING CSMS IN EXISTING CLOUD INFRASTRUCTURE

To show integration process of CSMS module and IPFIX sensors inside typical cloud datacenters, in this section we demonstrate deployment example. For example, we have already deployed cloud infrastructure based on open source private Cloud Eucalyptus as shown on Fig 5.

Eucalyptus Cloud Controller usually runs on a Linux-based computer with two network interface cards (NIC). Cloud Controller is a front end of the Cloud Infrastructure and it divides network on two parts: public local area network (on Fig 5. Public switch) and private local area network (on Fig 5. Private switch). We suggest to deploy CSMS module on Cloud Controller as it is central part of the Cloud Infrastructure and it is connected both private and public networks. Also, we suggest deploying firewall equipment, which is connected to the Public switch and able to block outside IP addresses in case of receiving command from CSMS.

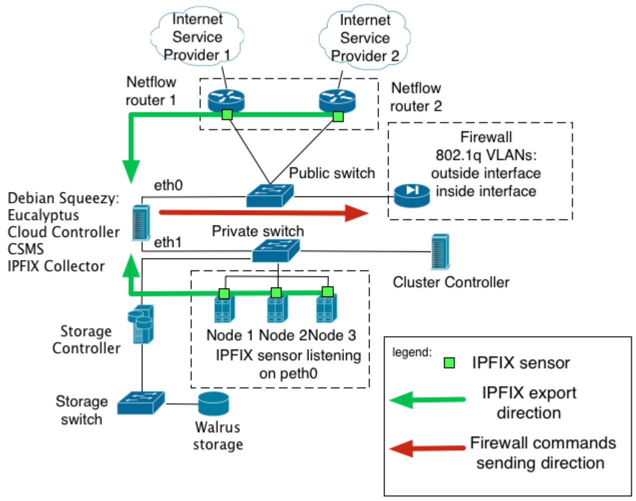
Internet Service Provider 1
Internet Service Provider 2
Netflow router 1
Netflow router 2
Firewall 802.1q VLANs: outside interface inside interface
Public switch
eth0
Debian Squeezy: Eucalyptus Cloud Controller CSMS IPFIX Collector
Private switch
eth1
Cluster Controller
Storage Controller
Node 1 Node 2Node 3 IPFIX sensor listening on peth0
Storage switch
Walrus storage
legend:
🟩 IPFIX sensor
IPFIX export direction
Firewall commands sending direction

Figure 5. CSMS deployment inside Eucalyptus Cloud infrastructure.

IPFIX sensors deployed in the Cloud Infrastructure Nodes (component of the Cloud where runs Virtual Machines of the End User) and send information to the IPFIX collector, which is also deployed on the Cloud Controller. In addition, IPFIX data is exporting from the border routers. CSMS module analyzes incoming data from the several sources (Nodes and Routers) and performs anomaly recognition actions – compare anomaly detectors patterns against observing network traffic data. In the case when anomaly discovered, CSMS performing IP address check process to be sure that traffic not from own or trusted networks and then sends command to Firewall equipment in order to block malicious IP address. Advantage of this approach lies in possibility to deploy IPFIX sensors in every operating system which supports traffic capturing. It means that no matter which kind of Cloud or Virtualization technology going to be used, the most important is the ability to export IPFIX data from network equipment or from virtual network interfaces of the Cloud nodes.

VII. CONCLUSION AND FUTURE WORKS

Flow-based measurement protocols such as IPFIX are an appropriate source of network traffic information, which allows us to analyze traffic with statistical frameworks and approaches. In the paper we use the maximum entropy estimation approach to obtain the normal behavior network traffic profile based on IPFIX data. This way of monitoring network security is more productive and easy to implement in existing Clouds due to design and implementation of open source-based virtualizing software. The suggested approach of anomaly detection based on negative selection algorithm seems to be an appropriate way of monitoring in distributed environments such as a Cloud infrastructure network. It is ready to detect DDoS-attacks and other abuse traffic attacks, having an availability issue for Cloud computing as a main concern. Automatic response ability of the CSMS with the “Whois” and reverse DNS information, based on source IP address filtering, is a useful way to preserve customers from false-positive errors.

The future developments of this research are testing and implementing of proposed algorithms and approaches to find a suitable way of integrating them inside the existing open source Cloud infrastructures. Also an applicability of the described proposal to the network attacks should be analyzed.

REFERENCES

[1] Securing the Cloud: A review of Cloud Computing, Security Implications and Best Practicies http://www.savvis.com/en-us/info_center/documents/savvis_vmw_whitepaper_0809.pdf. VMware Inc.(2009) (last access date 13/03/2012).

[2] Schoo P., Fusenig V., Souza V. at el. Chanlanges of Cloud Networking Security. 2nd International ICST Conference on Mobile Networks and Management, September 2010 Santandar Spain (October 2010). HP Laboratories, HPL-2010-137 (2010).

[3] Claise B. RFC 3954 Cisco Systems Netflow Services Export Version 9 (2004).

[4] Claise B. RFC 5101 Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information (2008).

[5] Boschi E. and Trammell B. Bidirectional Flow Measurement, IPFIX, and Security Analysis. pp. 8-10 (2006).

[6] Mukhtarov M., Miloslavskaya N., Tolstoy A. Netowrk security Threats and Cloud Services Monitoring. ICNS 2011 May 22, 2011 Venice/Mestre Italy. pp. 141-145 (2011).

[7] Molnar D., Schechter S. Self Hosting vs. Cloud Hosting: Accounting for the security impact of hosting in the cloud. WEIS 2010, pp.149-164 (2010)

[8] Berger S., Caceres R., Goldman K. and others Security for the cloud infrastructure: Trusted virtual data center implementation. IBM J. RES & DEV. Vol. 53, No. 5, paper 6, pp.1-12 (2009).

[9] Bussani A., Griffin J. L., Jasen B., Julisch K., Karjoth G., Maruyama H., Nakamura M., et al., ‘‘Trusted Virtual Domains: Secure Foundations for Business and IT Services,’’ Research Report RC23792, IBM Corporation (November 2005) (2005).

[10] IEEE Standard 802.1Q for Local and Metropolitan Area Networks—“Virtual Bridged Local Area Networks”. IEEE 2005 see: http://standards.ieee.org/getieee802/download/802.1Q-2005.pdf

[11] Chaves, S.A.; Uriarte, R.B.; Westphall, C.B. "Toward an Architecture for Monitroing Private Clouds," IEEE Communications Magazine Vol. 49, Issue 12, pp. 130-137 (2011)

[12] Gu Y., McCallum A. and Towsley, D. Detecting anomalies in network traffic using maximum entropy. Tech. rep., Department of Computer Science, UMASS, Amherst, pp. 345-350 (2005).

[13] PietraS.D., Pietra V.D. and Lafferty J. Inducing features of random fields. IEEE Transactions on Pattern Analysis andMachine Intelligence 19, 4, pp. 380–393 (1997).

[14] Stephanie Forrest, Alan S. Perelson, L. Allen, and R. Cherukuri. Self -nonself discrimination in a computer. In Proceedings of the 1994 IEEESymposium on Research in Security and Privacy. IEEE Computer Society Press, pp. 360-365 (1994).