International Journal of
Sensor Networks and Data Communications

Research Article Open Access

Monitoring Internet Access along with Usage of Bandwidth Using Intrusion Detection System

Rajagopal D* and Thilakavalli K

Department of Computer Applications, K.S. Rangasamy College of Arts and Science, Tiruchengode, Namakkal, Dt-637 215, India

Abstract

New Approach to observe web Access beside Usage of information measure victimization Intrusion Detection System could be a comprehensive web use observation and news utility for company networks. It takes advantage of the very fact that the majority companies give web access through proxy servers, like MS ISA Server, MS Forefront TMG, WinGate, WinRoute, MS Proxy, WinProxy, EServ, Squid, Proxy Plus, and others. Whenever the user accesses several websites, transfer files or pictures, these actions were logged. The system processes these log files to supply system directors a good vary of report-building choices. It might build reports for individual users, showing the list of internet sites visited, beside elaborate classification of web activity (downloading, reading text, viewing footage, observation movies, paying attention to music, and working). This technique might produce comprehensive reports with analysis of overall information measure consumption, building easy-to-comprehend visual charts that show the areas wherever wasteful information measure consumption has eliminated. This new approach is employed to observation the web information measure employed by the user. victimization this technique will simply decide that user fill the information measure most heavily, when, and what specifically they transfer, what proportion time they pay on-line, and what knowledge transfer traffic they produce.

Keywords: Bandwidth; Intrusion detection system; Proxy server; Network traffic; Network security

Introduction

Network managers and directors should get on guard against all types of unauthorized network use [1]. Intrusion Detection System observation network traffic for activity that falls inside the definition of prohibited activity for the network [2]. When found, the Intrusion can alert directors and permit them to require corrective action, interference access to vulnerable ports, denying access to specific science addresses, or move down services wont to enable attacks, this fast-alert capability makes an Intrusion Detection System the front-line weapon within the network directors war against hackers. The planned Intrusion Detection System put in on the server that serves native hosts and users over web. There are four actors within the system monitor, user, network and computer user. User sends request to the server over the web or native space Network and Intrusion Detection System can analyze the packets received by the server. This Intrusion Detection System detects each internal and external intrusion. If it detects any intrusion then it alerts computer user (Figure 1).

The planned approach permits centralized monitor of Users web access prevents personal usage of company information measure, reduces the web expenses, very easy-to-use. It will begin observation user's couple of minutes once, once the installation complete, works with all trendy proxy servers, permits the generation of a good range of reports and diagrams, that show the potency of proxy server usage, and it's a task computer hardware to automates the creation and delivery of reports to authorize personnel.

Advantages of the approach

- Allows centralized observation of Users web access
- Prevents personal usage of company information measure and reduces the web Expenses
- Extremely easy-to-use; will begin observation users couple of minutes once, once the installation is complete
- Works with all trendy proxy servers and permits the generation of a nice range of reports and diagrams, that show the potency of Proxy Server usage
- Task computer hardware to automates the creation and delivery of reports to authorize personnel.

Intrusion Detection System

Intrusion Detection refers to the method of observation the system for unauthorized access incidents, which might be the violation of the protection policy, system use policy, or the other security standards [2-5]. On the opposite hand, An Intrusion Prevention System (IPS) prevents unauthorized access incidents from being prosperous. To safeguard the system from any attacks, Intrusion Detection and Prevention System (IDPS), that give an utterly machine-controlled observation service, deployed on the systems [3]. Most of the IDPS systems log the incident on every occasion an attack on the system is that observe and notifies the administration of the system so all necessary actions will taken to avoid such incidents once more within the future. The directors of the system also can put together the IDPS to observe the violations of the tip user policies and alternative unauthorized activities [3].

Intrusion Detection System Types

Network intrusion detection system

It is a freelance platform, which identifies intrusions by examining network traffic and monitors multiple hosts [2]. Network Intrusion

***Corresponding author:** Rajagopal D, Department of Computer Applications, K.S. Rangasamy College of Arts and Science, Tiruchengode, Namakkal Dt-637 215, India, Tel: +91 4288 274741; E-mail: sakthiraj2782007@gmail.com

Received October 16, 2014; **Accepted** January 07, 2015; **Published** January 09, 2015

Citation: Rajagopal D, Thilakavalli K (2015) Monitoring Internet Access along with Usage of Bandwidth Using Intrusion Detection System. Sensor Netw Data Commun 4: 119. doi:10.4172/2090-4886.1000119

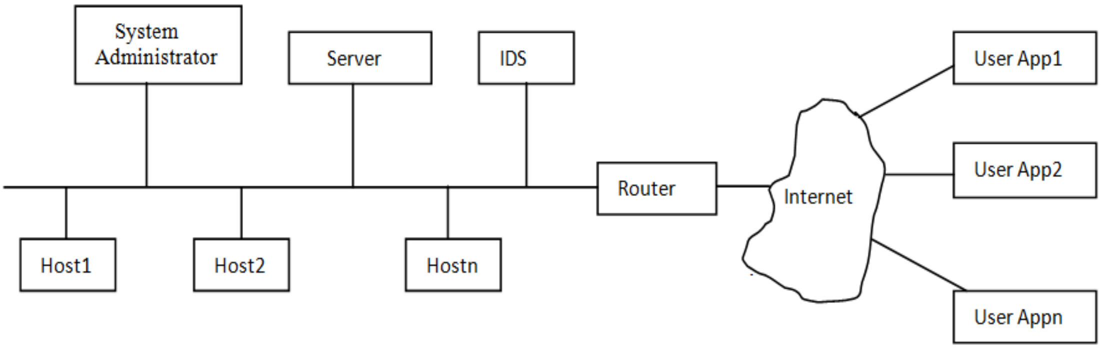


Figure 1: IDS working place in the network.

Detection System gain access to network traffic by connecting to a network hub, network switch designed for port mirroring, or network faucet [6,7]. A NIDS is place on a network to investigate traffic in search of unwanted or malicious events. Network traffic designed on varied layers; every layer delivers knowledge from one purpose to a different. The OSI model and transmission management protocol (TCP)/IP model show however, every layer stacks up. Inside the TCP/IP model, rock bottom link layer controls however, knowledge flows on the wire, like dominant voltages and the physical addresses of hardware, like Mandatory access Control (MAC) addresses (Figure 2).

The web layer controls address routing and contain the science stack. The transport layer controls knowledge flow and checks knowledge integrity. It includes the communications protocol and user datagram protocol (UDP). Lastly, the foremost sophisticated however most acquainted level is that the application layer, that contains the traffic employed by programs. Application layer traffic includes the online (hypertext transfer protocol [HTTP]), file transfer protocol (FTP), email, etc. Most NIDSs observe unwanted traffic at every layer; however concentrate totally on the applying layer.

Two main element sorts comprise a NIDS: appliance and software package solely. A NIDS appliance could be a piece of dedicated hardware: it is solely operated to be IDS. The Operating System (OS), software, and the network interface cards (NIC) are enclose within the appliance. The second element kind, software package solely, contains the entire IDS software package and generally the OS; but the user provides the hardware. Software-only NIDSs are usually more cost-effective than appliance-based NIDS because of they are doing not give the hardware; but, a lot of configuration is need, and hardware compatibility problems could arise.

Component types: Two main component types comprise a NIDS: appliance and software only. A NIDS appliance is a piece of dedicated hardware: its only function is to be an IDS. The operating system (OS), software, and the network interface cards (NIC) are included in the appliance. The second component type, software only, contains all the IDS software and sometimes the OS; however, the user provides the hardware. Software-only NIDSs are often less expensive than appliance-based NIDS because they do not provide the hardware; however, more configuration is required, and hardware compatibility issues may arise.

With an IDS, the "system" component is vital to efficiency. Often a NIDS is not comprised of one device but of several physically separated components. Even in a less complicated NIDS, all components may be present but may be contained in one device. The NIDS is usually made of components identified, but more specifically, the physical components usually include the sensor, management sever, database server, and console.

Sensor: The sensor or agent is the NIDS component that sees network traffic and can make decisions regarding whether the traffic is malicious. Multiple sensors are usually placed at specific points around a network, and the location of the sensors is important. Connections to the network could be at firewalls, switches, routers, or other places at which the network divides.

Management server: The management server will make decisions based on what the sensor reports. It can also correlate information from several sensors and make decisions based on specific traffic in different locations on the network.

Database server: Database servers are the storage components of the NIDS. From these servers, events from sensors and correlated data from management servers can be logged. Databases are used because of their large storage space and performance qualities.

Console: As the user interface of the NIDS, the console is the portion of the NIDS at which the administrator can log into and configure the NIDS or to monitor its status. The console can be installed as either a local program on the administrator's computer or a secure Web application portal. Traffic between the components must be secure and should travel between each component unchanged and unviewed. Intercepted traffic could allow a hacker to change the way in which a network views an intrusion.

Inline: An Inline NIDS sensor is placed between two network devices, such as a router and a firewall [8]. This means that all traffic between the two devices must travel through the sensor, guaranteeing that the sensor can analyze the traffic. An inline sensor of an IDS can be used to disallow traffic through the sensor that has been deemed malicious. Inline sensors are often placed between the secure side of the firewall and the remainder of the internal network so that it has less traffic to analyze (Figure 3a and 3b).

As the above diagram on the left, the computer running snort is connected to the firewall, the firewall would be configured with a "mirror" or "spanning" port that would essentially copy all of the incoming and outgoing traffic to a particular interface for the snort software to monitor. This way, any suspicious traffic passing the border of the network would be subject to examination.

As the above diagram on the right, the traffic is passing directly through the snort machine, using two Ethernet interfaces. This is an excellent solution for environments where a mirror port is unavailable,

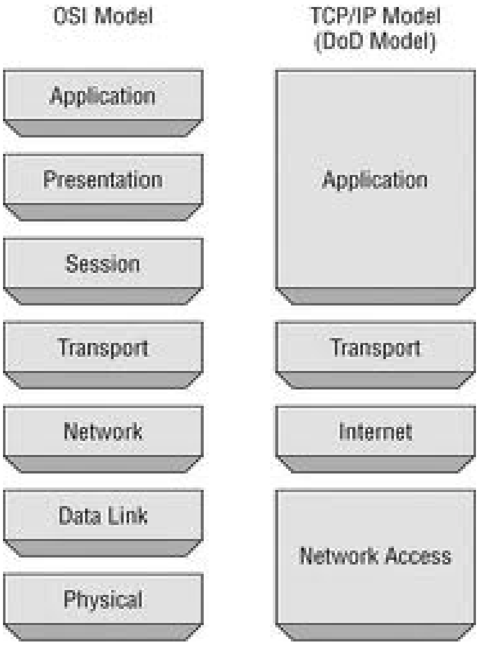


Figure 2: OSI and TCP/IP Models.

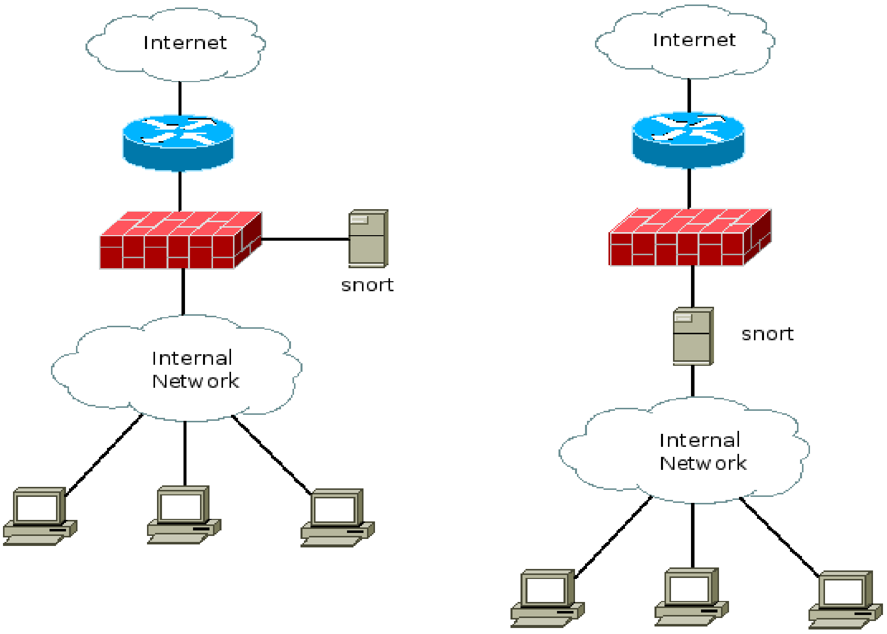


Figure 3: (a) Snort is connected to the firewall (b) Traffic is passing directly through the snort.

such as a branch office using low-end networking equipment that can't provide the additional interface. (It is important to note that a NIDS should be carefully placed within the network topology for maximum effectiveness. If two of the client machines in these diagrams are passing suspicious traffic between them, the snort machine will not notice; it only sees traffic destined for the Internet. It is always possible, of course, to run multiple NIDS systems and tie all of the alerts into one console for processing so as to eliminate these blind spots.)

Because of its large install base, rules for detecting new threats are constantly being produced and published for free usage on sites like Emerging Threats. If the administrator want to be alerted when a host on the network is connecting a known botnet controller [9], for example, the up-to-the-minute rules for this can be downloaded from ET. The same goes for signatures of new worms and viruses, command-and-control traffic, and more. So a NIDS is an excellent tool for detecting when a host on the network has been compromised or is otherwise producing suspicious traffic.

Passive: A passive sensor analyzes traffic that has been copied from the network versus traffic that passes through it. The copied traffic can come from numerous places.

Spanning port: Switches often allow all traffic on the switch to be copied to one port, called a spanning port. During times of low network load, this is an easy way to view all traffic on a switch; however, as the load increases, the switch may not be able to copy all traffic. Also, if the switch deems the traffic malformed, it may not copy the traffic at all the malformed traffic that may be the type the NIDS sensor must analyze.

Network tap: A network tap copies traffic at the physical layer. Network taps are commonly used in fiber-optic cables in which the network tap is inline and copies the signal without lowering the amount of light to an unusable level. Because network taps connect directly to the media, problems with a network tap can disable an entire connection.

Advantages of network based ids

- Monitor network for port scans.
- Monitor network for malicious activity on known ports such as http port 80.
- Identify varied varieties of spoofing attacks.
- Does not impact network performance.
- Increased tamper resistant.
- Operating systems independent.

Drawbacks of network based ids

- Packets lost on flooded networks.
- Reassemble packets incorrectly
- No understanding of O/S specific application protocols like SMB.
- No understanding of obsolete network protocols.
- Does not handle encrypted data.

Host-based intrusion detection system

It consists of an agent on a host that identifies intrusions by analyzing system calls, application logs, file-system modifications (binaries, secret files, capability databases, Access management lists etc.) and alternative host activities and state [7]. Whereas a NIDS watches the traffic on a network phase, HIDS watches the activities of a selected host. A common open-source HIDS system is OSSEC, named as a contraction of Open Source Security [10].

Much like a NIDS, the position of HIDS software package must arrange carefully. The user doesn't need to receive an alert on every occasion a file is reaching on a digital computer. The system has fastidiously to put together and the monitored behaviors cropped to on eliminate false alarms and make sure the true security problems are noticed and alerted properly [11].

Advantages of host based ids

- Monitor events native to a host, and might observe prosperous or failure of attacks that cannot be seen by a network-based IDS.
- Operate in the setting during which network traffic encrypted.
- Unaffected by switched networks and is independent of topology.
- Monitor system specific activities like file access, user access, etc.
- Provide thorough data gathered via logs and audit; for instance, Kernel logs know who the user is.
- No extra hardware is required to implement Host based IDS solution.
- When Host-based IDSs operate on OS audit trails, they will facilitate observe attacks that involve software package integrity breaches.

Drawbacks of host based ids

- Host based IDS are tougher to manage, as data should be designed and managed for each host singly.
- Host based IDS's network blind and cannot detect a network scans or other such surveillance that targets entire network.
- If the host is compromised, collected log by Host based IDS can be subverted.
- Disabled by bound denial-of-service attacks.
- Uses operating system audit trails as an information source. The number of information is large and might need extra native storage on the system.
- Inflict performance deficiency on monitored host.

Stack-based intrusion detection system

This type of system consists of an evolution to the HIDS systems. The packets examined as they are going through the TCP/IP stack and, therefore, it is not necessary for them to work with the network interface in promiscuous mode. This reality makes its implementation to be dependent on the Operating System [12].

This can be latest IDS technology and varies dramatically from vendor to vendor. Stack Based IDS works by integration closely with the TCP/IP stack, allowing packets to be watch as they traverse their way up the OSI Layers. Observation the packets during this means permits the IDS to drag the packets from the stack. To be complete Stack- Based ID ought to watch each incoming and outgoing network traffic on a system. By monitoring network packets destined just for a simple host, the principle is to create the IDS have sufficiently low overhead so each system on the network will run Stack-Based IDS.

Intrusion Detection Techniques

Statistical anomaly-based ids

A statistical anomaly-based IDS determines traditional network activity like what type of bandwidth is mostly used, what protocols are used, what ports and devices usually connect to each other- and alert the administrator or user once traffic is detected that is abnormal(not normal) [13]. The anomaly-based detection model detects the attacks based on the profiles. The profiles contain the patters or the traditional behavior during which the system is used. The profiles supported specific users, networks, or the applications. They are making by monitoring the system use over a period, known as the evaluation period. This model compares the present activities with the profiles to get the abnormal activity in progress, which regularly is an attack. Since the system use and also the network use do not seem to be not static and always contain some variation over time, the profile should additionally modify consequently. Therefore, once the creation of the profiles in the evaluation period, an IDPS changes the profiles over time. The samples of the profiles mentioned below. A user profile contains an email activity of 5%. Once the IDPS victimization anomaly-based detection model senses that the e-mail activity on the system is over 5%, it will consider about it an attack. Over the past few weeks, on average a user performs that open, read, and write operations on the file system for 2% of the time. Once the IDPS detects a growth within the file access operations, it reports an attack incident. The advantage of the anomaly-based detection model is that it is able to detect even unknown attacks by comparison the present abnormal events with something that is considering traditional. Further, this model also can be more efficient than the signature-based model given that there are a large range of signatures to compare inside the signature-based detection model. On the other hand, the attack incidents that anomaly-based model produces don't seem to be terribly specific and it takes some efforts by the administrator to pin - purpose the basis of the attack. Additionally this model subject to a "slow attack". In this type of attack, the attacker first finds out the threshold between the normal and abnormal activities. The attacker then would slowly attack the system making sure that the activities during the attack do not reach the threshold which results into the anomaly-based detection model not detect the attack.

Advantages of anomaly based ids

- Identify any potential attack.
- Identify attacks that have not seen before, or close variants to antecedently well-known attacks.

Drawbacks of anomaly based ids

- Normal will amendment over time, introducing the requirement for periodic on-line preparation of the behavior profile, result either in inaccessibility of the intrusion detection system or in extra false alarms.
- Current implementations give high false alarms.
- Requires experience to work out what triggered an alarm.

Signature-based ids

Signature based IDS monitor's packets within the Network and compares with pre-configured and pre-determined attack patterns called signatures [14]. The difficulty is that there will be lag between the new threat discovered and Signature being applied in IDS for detecting the threat. Throughout this lag time, IDS are going to be unable to spot the threat.

The signature refers to the pattern during which a antecedently well-known attack was performed. The signature-based detection methodology is that the method of comparison the present events with the signatures. The signature-based detection model produces terribly specific attack event reports as oppose to the anomaly-based detection model [5]. The disadvantage of a signature-based detection model is its inability to detect new unknown attacks since the system does not have any signature entry within the system for the new attacks.

Benefits of signature based ids

- Provides terribly low false alarms as compare to Heuristic based IDS.
- Provides detail contextual analysis providing steps for preventive or corrective actions.

Drawbacks of signature based ids

- It is tough to assemble data concerning well-known attacks and keeping up-to-date with new vulnerabilities.
- Signatures and corrective recommendations are generalized; so it makes it tougher to grasp them.
- Knowledge concerning attacks is extremely centered, keen about the operating system, version, platform, and application.
- Signature/Pattern based IDS are more popular and commercially used than Heuristic/Anomaly detection based IDS. Major vendors such as ISS offer network based and host based signature detection.

Stateful protocol inspection

Stateful protocol inspection is similar to anomaly based detection, but it can also analyze traffic at the network and transport layer and vender-specific traffic at the application layer, which anomaly-based detection cannot do.

Differences between Detection Techniques

Misuse detection vs. Anomaly detection

A misuse detection system, also known as a Signature-Based Intrusion Detection System identifies intrusions by watching for patterns of traffic or application data presumed to be malicious. These types of systems are presumed to be able to detect only 'known' attacks. However, depending on their rule set, signature-based IDSs can sometimes detect new attacks which share characteristics with old attacks.

An Anomaly-Based Intrusion Detection System identifies intrusions by notifying operators of traffic or application content presumed to be different from 'normal' activity on the network or host [15,16]. Anomaly-based IDSs typically achieve this with self-learning. In anomaly detection, the system administrator defines the baseline, or normal, state of the network's traffic load, breakdown, protocol, and typical packet size. The anomaly detector monitors network segments to compare their state to the normal baseline and look for anomalies [17].

Network-based vs. Host-based systems

In a network-based system, or NIDS, the sensors are located at choke points in the network to be monitored, often in the network borders. The sensor captures all network traffic flows and analyzes the content of individual packets for malicious traffic. In a host-based system, the sensor usually consists of a software agent which monitors all activity of the host on which it is installed. Hybrids of these two types of system also exist. A Network Intrusion Detection System is an independent platform which identifies intrusions by examining network traffic and monitors multiple hosts. Network Intrusion Detection Systems gain access to network traffic by connecting to a hub, network switch configured for port mirroring, or network tap. An example of a NIDS is Snort. A Host-based Intrusion Detection System consists of an agent on a host which identifies intrusions by analyzing system calls, application logs, file-system modifications (binaries, password files, capability/acl databases) and other host activities and state. A Hybrid Intrusion Detection System combines both approaches. Host agent data is combined with network information to form a comprehensive view of the network. An example of a Hybrid IDS is Prelude [18].

Passive system vs. reactive system

In a passive system, the IDS sensor detects a potential security breach, logs the information and signals an alert on the console. In a reactive system, the IDS respond to the suspicious activity by logging off a user or by reprogramming the firewall to block network traffic from the suspected malicious source, either autonomously or at the command of an operator. Though they both relate to network security, IDS differs from a firewall in that a firewall looks out for intrusions in order to stop them from happening. The firewall limits the access between networks in order to prevent intrusion and does not signal an attack from inside the network. IDS evaluate a suspected intrusion once it has taken place and signal an alarm. IDS also watch for attacks that originate from within a system.

Network Bandwidth

Network bandwidth refers to the amount of knowledge being transmitted across a network at any given purpose in time. Network bandwidth will decrease if devices that change networked communications fail. Network bandwidth might be forced by each hardware and software package limitations. Optimizing the out there Network information measure could be a primary responsibility of network administrators.

Current Techniques in Network Security

A number of techniques have been invented in the past few years to help a system administrator in strengthening the security of a single host or the whole computer network.

Audit trails

"A chronological record of system activities that is sufficient to enable the reconstruction, reviewing, and examination of the sequence of environments and activities surrounding or leading to an operation, a procedure, or an event in a transaction from its inception to final results." defined by the National Computer Security Center

Audit trail can be used in determining whether an unexpected or unauthorized behavior has occurred in a system.

Fire wall

A recent trend in network security enhancement involves the use of firewall, which is a collection of filters and gateways that shield trusted networks within a locally managed security perimeter from the external untrusted networks [13].

Screening router

Screening router is a router, which in addition to forwarding packets likes a normal router, also examines data in the packets, and applies some predefined access control policies on the packets to determine whether they can be forwarded to the next hop or should be discarded.

Application gateway

Application gateways provide one or more of the following functionality: relay, proxy and server filter. Relay gateway, passes the data between the two sides of a firewall system. In some special environments, like a company using "local" IP addresses (i.e. visible only within the company) for internal network, a relay gateway should also provide the function for translating these addresses before they are sent out.

Proxy is of most importance to a firewall system, for most of access control policies are enforced through application proxies. Usually, a proxy gateway is application specific. When a client program inside the firewall requires a connection with an outside server, an application proxy on the firewall will handle the request first.

Server filter works in the opposite direction as an application proxy. It handles the incoming connection requests from external network to the internal servers. Similar to inetd under most UNIX systems, a server filter acts as a proxy for multiple internal application servers. When receiving a connection request, the server filter dispatches it to the corresponding application server.

As an application gateway examines more data in a network packet than a screening router does, it provides more power in network intrusion detection and prevention. On the downside, it requires more system resources and more processing time.

Design and Methodology

There is a need for testing directly two different types of NIDS, in other words anomaly and signature detection. There is also a need for online evaluation using realistic simulated traffic generation, as opposed to offline evaluation using network traffic traces.

The initial objective of this experiment was to set up a testbed for two different types of NIDS and generate simulated background traffic as well as range of exploits. Such an experiment proved too generic since the choice of exploits ready to use was relatively small compared to the amount of existing exploits. Instead, the experiment was split in two: a first experiment on the learning window variation of an anomaly IDS, and a second experiment testing two different types of IDS in a specific, well-defined scenario. States that a valid computer security experiments should consist of only one varying component. The following sections define an overview of the testbeds used.

Network architecture

The basic network architecture is composed of a router and switch. Since the background traffic is split into two IP address ranges according to whether it is client or server, two **V**irtual **L**ocal **A**rea **N**etwork (VLAN) are needed to mimic internal and external traffic. The router is used in this configuration in order to route traffic between both VLAN s. Another essential piece of configuration is setting up the **S**witched Port Analyser (SPAN) port on the switch in order to send all traffic crossing the switch to the IDS station for analysis.

Training window experiment

This experiment aims at demonstrating any effects that a variation of training window length could have on an anomaly-based IDS .This station is linked to a switch and monitors all network traffic crossing this network device. The traffic generator is used to produce benign background traffic for anomaly system profile creation. Ideally, this station should produce this type of traffic with a traffic generation simulation tool. The exploit generator is used after the profile generation phase has been completed.

Finally, the router is used in this testbed in order to route or discard the generated traffic and make all the connections appear real to the IDS. To sum up, the anomaly-based IDS will be subjected to different learning periods. For each period, the profile created will be stored for the next experimental phase, being the attack detection. After this profile generation phase, the IDS will be subjected to a mix of benign background traffic and malicious traffic (Figure 4).

Scenario

This experimental scenario is realized in order to focus this research on a specific type of threat rather than only available threats. For this scenario, the following background is to be considered. A renowned bank branch computer networks system includes an FTP server hosting highly sensitive data, such as bank account details for example.

Currently, the corporation uses the following tools as part of its security system: a firewall at the boundary with the untrusted network, antivirus on local machines and built-in IDS on the gateway router analyzing traffic going in and out of the trusted network, such as on a Cisco router. The security at the boundary of the corporate network is optimum, but the security staff is worried about threats present on the inside of their network. Insiders threats are multiple, although here the main concern is data theft from the FTP server, only protected by a username and password combination.

In order to protect the branch from such a threat, the security staff would like to know which type of NIDS would be best suited in this case (Figure 5).

There are some considerations to take into account with regards to this scenario. Sensitive data would probably not be stored on a simple FTP server in a real case environment, and the access to such a server would probably be more securely controlled. The simplistic approach used in this scenario is chosen due to time considerations and testing focus: a FTP is faster to breach than a more secure server,

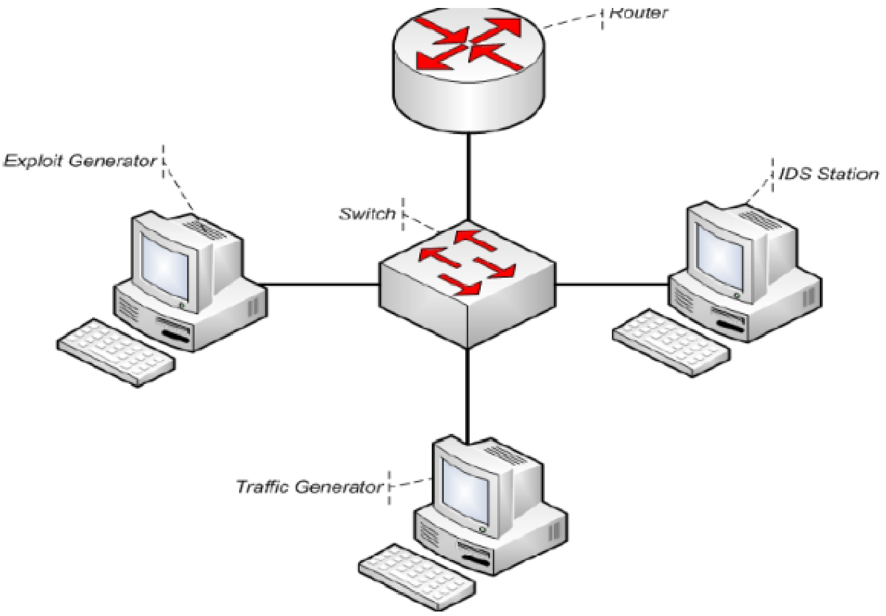


Figure 4: Training window experiment design.

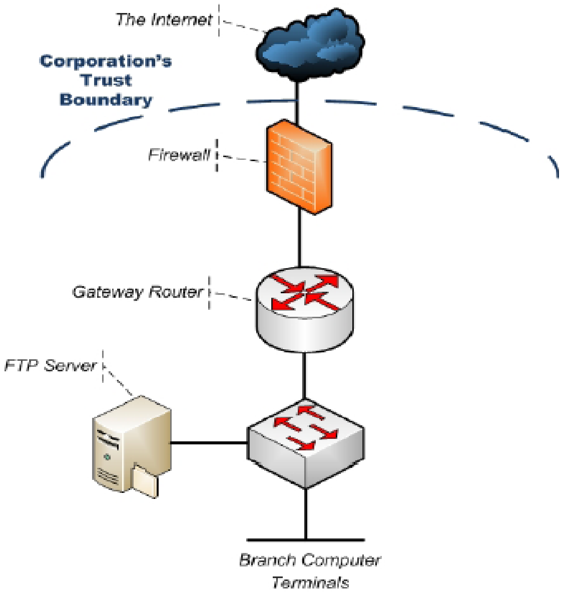


Figure 5: Scenario bank computer system.

and this experiment is focused on NIDS rather than server security. This scenario can show which IDS is best for such an environment. Adding additional security measures could not do any harm but make the whole system more secure (Figure 6).

The testbed is very similar to the one used in the learning window experiment, with the difference of an extra machine running an FTP server. The scenario threat is data theft. Data theft is usually composed of a collection of exploits following the steps highlighted. In this case, the data theft consists of:

- Live IP addresses scan
- Portscan on live addresses
- Brute force attack on FTP username/password
- Data theft

The exploit generation station will carry out every step of a data theft threat.

Distributed intrusion detection system

The Distributed Intrusion Detection System (DIDS) is an intrusion detection mechanism which was developed jointly by the University of California at Davis, Lawrence Livermore Laboratory, Haystack Laboratory and the U.S. Air Force. DIDS combines attributes of a network monitoring system with the system-level capabilities of an audit record-based combined anomaly/misuse detector. DIDS incorporates a monitor on each host, a monitor on the local area network (LAN), and a DIDS director [19].

Each host monitor consists of a host event generator and a host agent. The host event generator reviews the audit data from the host for indications of events which may be part of an attack. The DIDS host event generators also utilize user and group profiles to identify anomalous behaviors in the audit record. The information identified by the host event generator is reported to the DIDS director by the host agent. The LAN monitor is the network equivalent of the host monitor. It includes the LAN event generator and the LAN agent.

The DIDS director forms the heart of the intrusion detection mechanism. It is composed of three components, the communications manager, an expert system and a user interface. The communications manager is receives input from each of the host monitors and from the LAN monitor and forwards the information to the expert system for analysis. The communications manager is also capable of forwarding requests for additional information from the expert system to the host monitors and the LAN monitor.

The DIDS expert system is a rule-based system which is responsible to analyzing the information received from the monitors and reporting it to the security official. The final component of the DIDS system, the user interface, allows a security official to interactively review the status of the system, receive reports from the expert system, and request additional security-related information from the system. One of the essential elements of the DIDS system is the use of a Network-user Identification (NID). This is a process of establishing an identifier for each individual when they are initially logged into the network. This is especially important because many attackers use multiple accounts to attack a network or use the interconnectivity of computer networks to attempt to disguise their identity.

Once a user has logged into the network and been assigned a NID all subsequent activity conducted by that user is attributable through the NID. While the NID offers the potential to track an intruder through a variety of hosts and possible identities, there are ways to defeat the mechanism. By logging out of the monitored domain and then reentering under a different user id, an attacker can prevent DIDS from relating the two sessions. In addition, the DIDS system probably cannot attribute two related sessions to the same user if the user passes through an unmonitored domain. These difficulties aside, an initial DIDS prototype has successfully demonstrated the ability to track users through a monitored domain. Because of the complexity of the

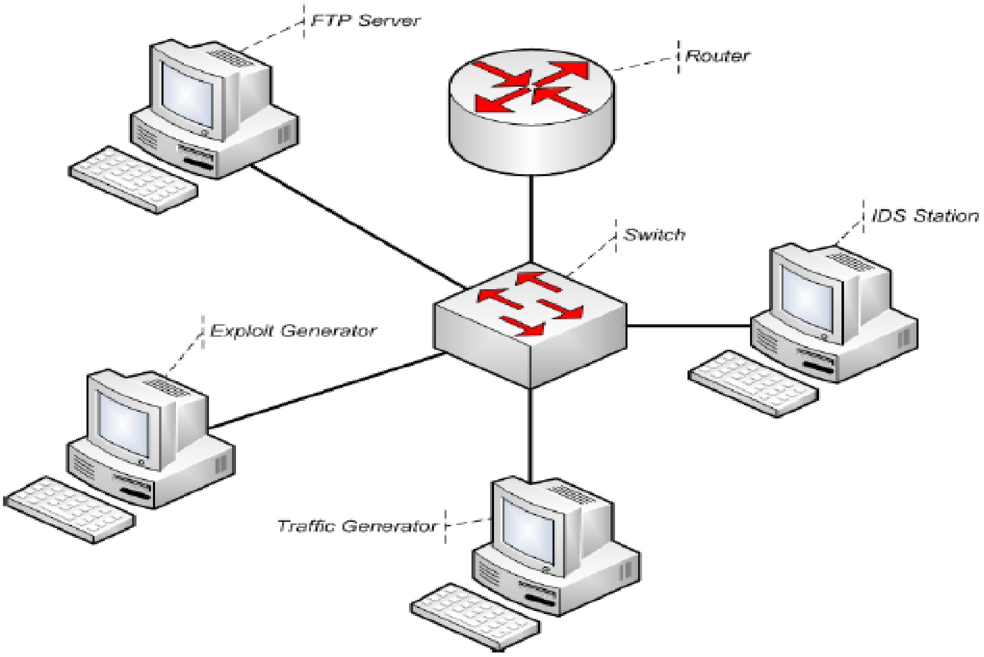


Figure 6: Scenario experiment design.

system and its use of audit data, DIDS retains the negative effect on the performance of the system which plagues most traditional intrusion detection systems. While this could be a significant disadvantage of the DIDS system, the innovative design of the system effectively addresses the difficulty in identifying intrusions in a networked environment.

State transition analysis tool (stat/ustat)

The State Transition Analysis Tool (STAT) and USTAT, the variation of STAT which was designed specifically for the UNIX operating system environment, are rule-based penetration detection approaches which characterize the process of an attack on a computer system as a series of transitions from an initial state to a compromised state. The technique defines specific events, called signature actions, which occur between each of the intermediate transitions. The omission of any of the signature actions results in a failed attack on the system.

If the current pattern of activity matches an established intrusion scenario, STAT/USTAT has the ability to predict the future activities of an attacker. The ability to predict behavior offers the advantage of allowing the Security Administrator to be more confident that an actual attack is occurring prior to utilizing any countermeasures. As more of the established scenario's activities are matched, the confidence level that an attack is occurring increases. In addition, because this technique does not rely on possibly unrelated events to indicate a potential attack, the incidence of false alarms reported by the system should be significantly reduced.

Another advantage of this approach is that because STAT/USTAT selects specific audit data for confirmation of potential intrusion patterns, only a portion of the audit data is actually reviewed. This reduces the reliance of the system on the entire set of audit data, thereby reducing the required storage space and memory requirements necessary for processing an entire audit trail. While STAT/USTAT offers significant advantages in its approach to intrusion detection, the technique is unable to detect other attack-type behavior such as denial of service attacks and masquerading. Because these indications of an attack cannot be ignored by an effective intrusion detection system, a mechanism which employs STAT/USTAT would also require a complementary rule-based anomaly/misuse detection system.

Tripwire

In November 1992, the COAST laboratory at Purdue University introduced Tripwire. Tripwire is an integrity checking program which permits a system administrator to monitor system files for addition, deletion, or modification.

Tripwire operates in one of four modes. In the database initialization mode, the program generate a database which contains all of the relevant information on the system files, including signatures. Because the baseline database is being generated based on the files which currently exist in the system, it is critical that the existing database is free of logic bombs, viruses, Trojan horses, or other attack programs. The integrity checking mode results in the creation of a new database from information contained in the configuration. The information in the new database is compared with the results contained in the original database. Any discrepancies are processed through a filter which determines which file attributes can be changed without adversely effecting the system. The remaining identified changes are then reported to the system administrator.

The final two operating modes are used to ensure that the information in the database is consistent. The database update mode calculates new signatures for those files which have been legitimately changed. In the interactive database update mode the program generates a list of those files which have been modified and updates those which are identified by the system administrator as legitimate.

Tripwire is a good tool for monitoring the status of system files. Tripwire makes no pretense of insuring the complete security of the computer system. It functions to notify system administrators of a very important indication of an intrusion. This information, combined with other security-related tools, should provide a more secure operating environment.

Graph-based intrusion detection system (GrIDS)

Researchers in the COAST laboratory have recently proposed a novel approach to intrusion detection based on the analysis of activity graphs. The Graph-Based Intrusion Detection System (GrIDS) is designed to analyze network activity in large networks for the presence of attacks [19]. GrIDS aggregates the actions of a networks users into the activity graphs. Based on a review of the structure of these graphs the system can identify patterns which indicate intrusive behavior. Information received from other intrusion detection devices and network monitors can be included in the attributes of the activity graphs.

Individual types of graphs will be maintained in graph spaces with the GrIDS system. Because there are a number of possible attacks on the network, multiple graph spaces must be maintained. Each graph space is dependent on a specific rule set which modifies the graphs within it's graph space based on inputs to the system. GrIDS is able to analyze activity on large networks because of it's ability to model networks as a series of hierarchies. Each area within the hierarchy has a GrIDS module which is responsible for that area. Any activity which crosses area boundaries will be passed up to the GrIDS in the next higher level for resolution. The GrIDS in that level builds reduced graphs which model the underlying structure on a smaller scale. This ability to model subhierarchies allows GrIDS to monitor networks of increasing complexity. The true promise in the GrIDS system is in its ability to assist users in creating rule sets for the system. GrIDS includes a policy language which enables administrators to translate organizational policies and guidelines into rule sets which are used to analyze the network activity.

Thumb printing

Thumb printing is a method of tracking intruders through a sequence of logins, referred to by the authors as a connection chain. While it is not intended to be an independent intrusion detection system, it could prove to be a valuable addition to other technologies. Thumb printing was developed by researchers at the University of California at Davis in response to a weakness in DIDS.

A current weakness in this approach is that it assumes that the content of the connections along the chain are the same. As a result, the use of different encryption techniques by two points would render the method useless.

Cooperating security managers

While DIDS takes a centralized security approach to network intrusion detection, Cooperating Security Managers (CSM) decentralizes the process. A separate CSM is run on each computer which is connected to the network. Each CSM consists of six elements. The heart of the CSM is the Security Manager (SECMGR). The SECMGR receives input from the various CSM components and coordinates with CSM's on other hosts as users pass through the network. The command monitor (CMNDMON) intercepts the commands from the user and forwards them to the host intrusion detection system (IDS). While CSM requires the presence of an intrusion detection system on each host, the actual mechanism is separate from the CSM and can therefore be any intrusion detection tool. Any intrusions detected by the IDS are reported to the SECMGR.

CSM's ability to utilize a variety of intrusion detection systems also prevents the system from being limited by any of the specific approaches to intrusion detection. As new approaches are developed which more efficiently process user information, they can be incorporated into the CSM, effectively upgrading the CSM as a whole.

The network administrators, the senior management, and the users of their systems, on a scale of 1 (minimal) to 10 (very significant). The respondents considered security to be a major concern with an average score of 8.9. The reported levels of perceived significance of the other categories diminished to the typical users level of concern for security rating of 2.4.

The survey respondents to rate six types of threats in order of concern from 1 (lowest perceived threat) to 6 (most significant threat to their network). The survey results indicated that most of the respondents utilize a combination of security devices on their networks. Ninety percent of the respondents utilize the existing security features which are present in their host operating system. Some types of intrusion detection system and firewall mechanism are used in seventy-two percent of the networks. The respondents reported that they were utilizing their security mechanisms to defend their networks from external penetrations, masquerades, internal attacks, viruses and denial-of-service attacks. Seventy-seven percent of the respondents reported that their networks had been attacked in the past. A further breakdown of that group indicated that sixty-nine percent reported the attack to a superior or other authority. The same group responded that seventy-two percent were utilizing a firewall mechanism at the time that the attack occurred. Among those respondents who were using a security mechanism when attacked, sixty-three percent reported that the mechanism had reduced the severity of the attack.

This often consisted of a timely notification of the security administrator which allowed active defensive measures to be conducted before significant damage could occur to the system. Additional security measures were implemented after the attack by sixty-three percent of the group. Most of these measures consisted of improved security education and training of the users and the correction of well-known system flaws.

The results of that intrusion detection systems should be capable of identifying various types of threats, or be capable of being seamlessly incorporated with other security mechanisms which can defend against those threats not addressed by the intrusion detection system.

Proposed Model

Interacting with a pop3 server

Downloading an email from a POP3 server is rather straight forward. The communication with a POP3 server uses only few commands and is easily human readable. Once a connection, possibly with SSL, is established, the client needs to provide a user name and password to enter the POP3 state TRANSACTION, called 'connected' in Pop3MailClient (Figure 7).

Error handling & tracing

To further help with the investigation of communication problems, a Trace event is raised. It shows commands and responses exchanged between PopClient and PopServer, including warnings. It is strongly recommended to use this feature in the beginning of a project, because RFC1939 gives the server implementer great freedom. It often provides additional information which can be seen in the trace.

Server settings

The Pop3MailClient requires server name, port, should SSL be used, username and password in the constructor and they cannot be changed. To get the demo code running, need to enter own credentials for username and password in the following line:

Pop3.Pop3MailClient DemoClient=new Pop3.Pop3MailClient("pop.gmail.com",995, true, Username@gmail.com, "password");

Reading raw email

The method GetRawEmail returns the complete email content for one particular message number. RFC1939 specifies that only ANSI characters can be used and therefore the raw email can be easily displayed.

Auto reconnect after server timeout

The isAutoReconnect property is set, the Pop3MailClient tries to reconnect exactly once after a timeout. That's all it usually takes, but notice that any emails marked for deletion are not deleted on the server.

SMTP trace listener

The SMTP Trace Listener class is derived from the TraceListener class found in the System.Diagnostics library. The methods **MUST** override are:

- public override void Write(string Message)
- public override void WriteLine(string Message)
- public virtual void Fail(string);
- public virtual void Fail(string, string);
- public virtual void Close();
- public virtual void Flush();

The new flexible tracing architecture provides us the ability to

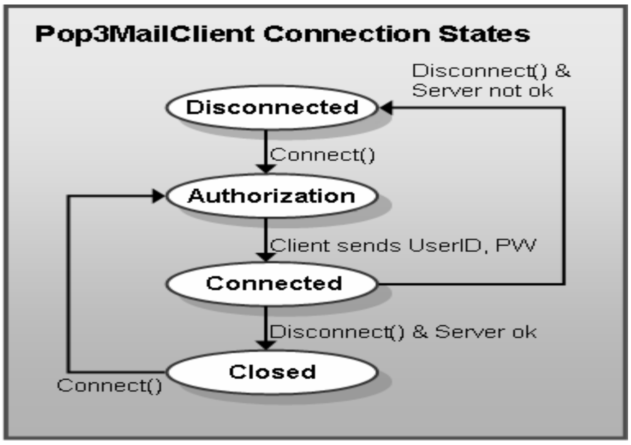


Figure 7: Interacting with POP3 Server.

Trace Listener. Other than TraceListener, new classes like TraceSource, TraceSwitch, and TraceFilter give us complete control over application tracing. The Trace Listener class, PortWriterTraceListener, that listens to Trace messages and sends the Trace messages to a UDP port.

Trace listener class enumerators

a. **Classes:**

TraceListener

TraceListenerCollection

TraceSource

TraceSwitch

TraceFilter

TraceEventCache

Trace

a. **Enumerators:**

TraceEventType

TraceLevel

TraceOptions

Port writer trace **listener**

The UDP protocol is efficient and fast, but less reliable. Tracing is not a mission-critical task, and reliability is not the highest priority criteria. In fact, there are only two abstract functions in the TraceListener class, Write and Writeline. Following is a functional Trace Listener class that, if used, will log the Trace in the event log [20].

Now, PortWriterTraceListener has two custom attributes called 'destination' and 'port'. These two custom attributes will be used to make the UDP connection. Trace Listener sends Trace messages to a UDP port using the UdpClient class of the System.Net.Sockets namespace. Need to do two things, connect to the destination computer, and send data to the port.

Trace**view**

The TraceView simply listens to the UDP port where PortWriterTraceListener sends the Trace messages and displays the messages in a ListView control. There are many useful features in TraceView, like logging Trace to a file, saving Trace to a file, search messages, filter messages, and copy messages to the Clipboard. SOAP Extensions to create reusable components to manage authorized users of Web Services and to track Web Services usage by those users (Figure 8). The figure below depicts the role of SOAP Extensions in the Web Services architecture:

The small solid circles in the above figure show the points in the serialization/de-serialization process where the incoming/outgoing messages can be tapped in SOAP Extensions (Figure 9).

Architecture of SOAP extensions: SOAP Extensions offer a rich and extensible mechanism for implementing reusable infrastructure components for Web Services. Their ease of implementation without losing flexibility has made them the "way to go" for the developer community. SOAP Extensions–based reusable components will be the backbone of Web Service offerings, and enforces business rules without distracting from the design goals defined at the start of this article. The following list provides a few broad categories where these types of components can be used:

- Data encryption
- Authentication and authorization
- Accounting/logging
- Monitoring system performance

The figure below depicts an example of this process flow (Figures 10-12):

Bandwidth

Bandwidth performance is one of the critical requirements for every website. In today's time major cost of the website is not hard disk space but its bandwidth. So transferring maximum amount of data over the available bandwidth becomes very critical and how can use IIS compression to increase bandwidth performance. How does IIS compression work?

The user requests for a 'Home.html' page which is 100 KB size. IIS serves this request by passing the 100 KB HTML page over the wire to the end user browser (Figure 13).

When compression is enabled on IIS the sequence of events changes as follows:-

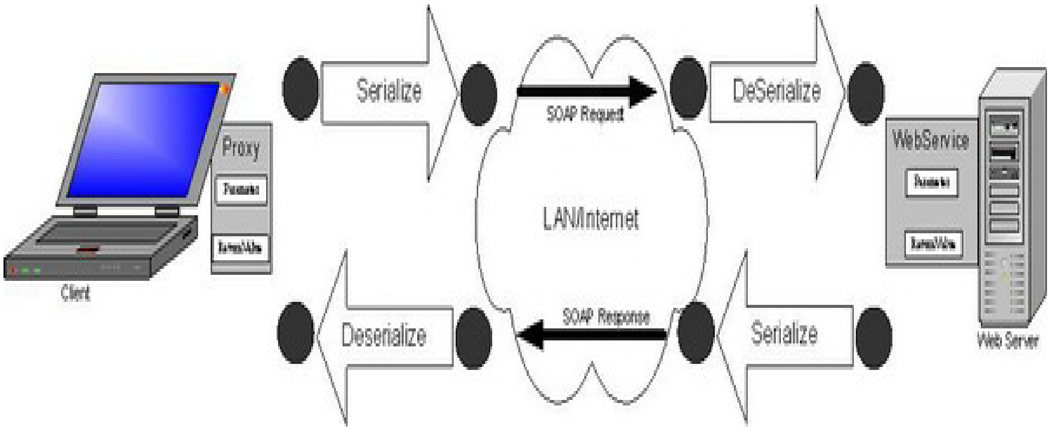


Figure 8: SOAP Extensions in the web service architecture.

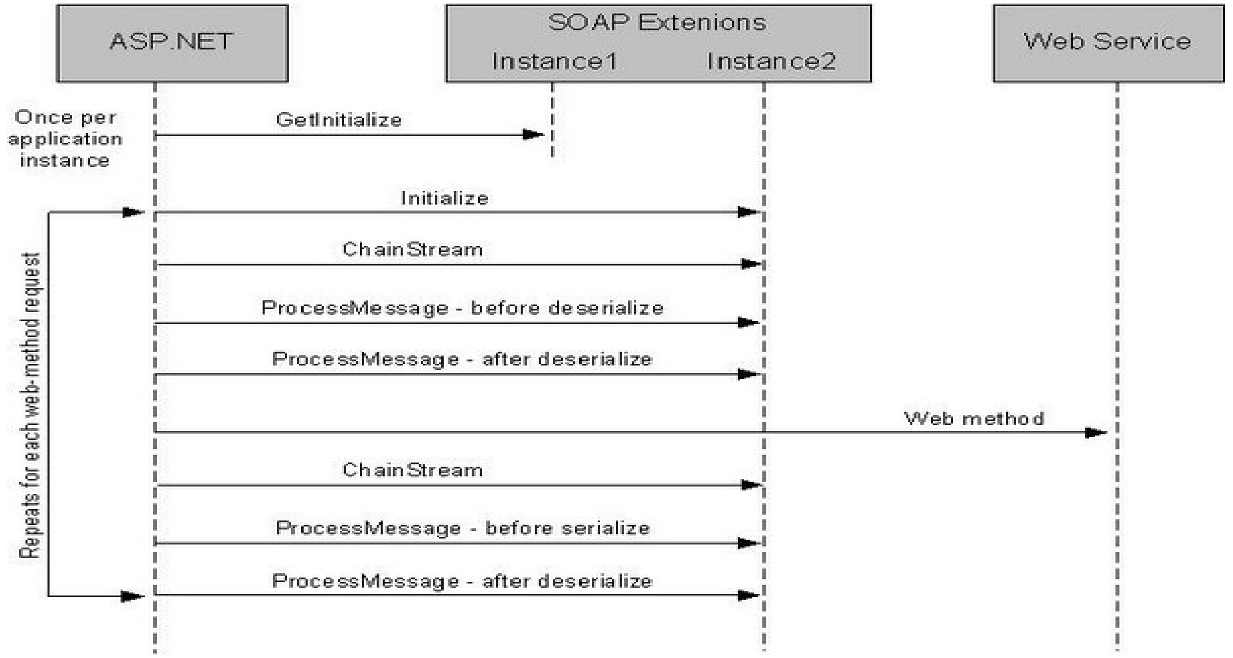

ASP.NET
SOAP Extenions
Instance1
Instance2
Web Service
Once per application instance
GetInitialize
Initialize
ChainStream
ProcessMessage - before deserialize
ProcessMessage - after deserialize
Web method
ChainStream
ProcessMessage - before serialize
ProcessMessage - after deserialize
Repeats for each web-method request

Figure 9: SOAP Extension Life Time.

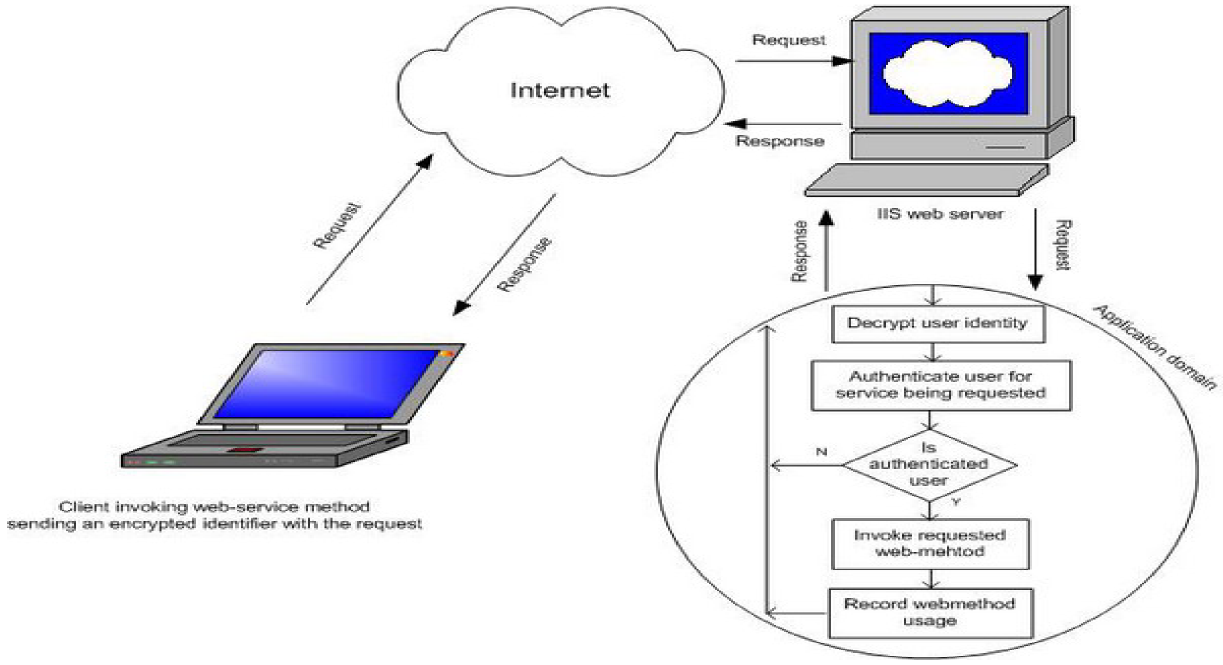

Internet
Request
Response
IIS web server
Request
Response
Request
Response
Client invoking web-service method sending an encrypted identifier with the request
Application domain
Decrypt user identity
Authenticate user for service being requested
Is authenticated user
N
Y
Invoke requested web-mehtod
Record webmethod usage

Figure 10: SOAP Extension architecture.

- User requests for a page from the IIS server. While requesting for page the browser also sends what kind of compression types it supports. Below is a simple request sent to the browser which says its supports gzip and deflate.

GET/questpond/index.htmlHTTP/1.1

Accept: image/gif,image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel*/*

Accept-Language:en-us

Accept-Encoding:gzip,deflate

User-Agent:Mozilla/4.0

Host:www.questpond.com

Connection: Keep-Alive

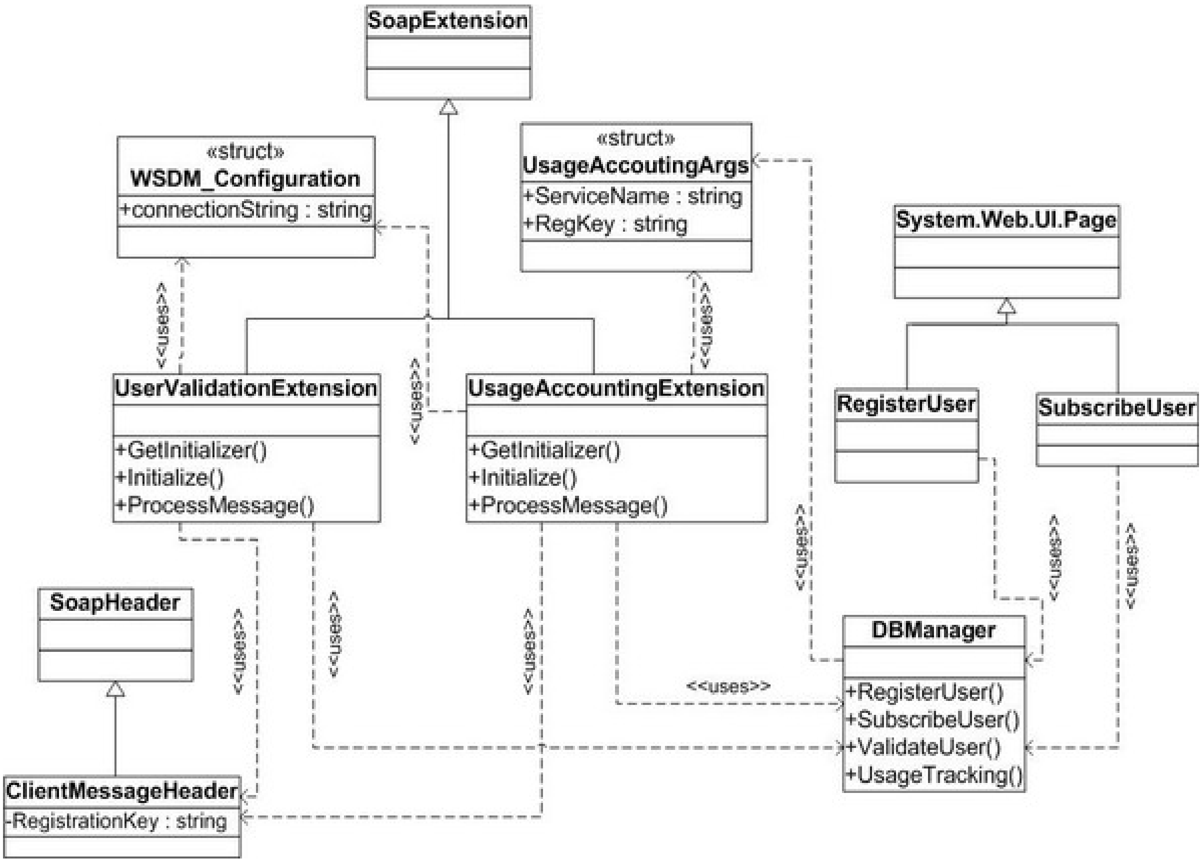


Figure 11: SOAP extension class diagram.

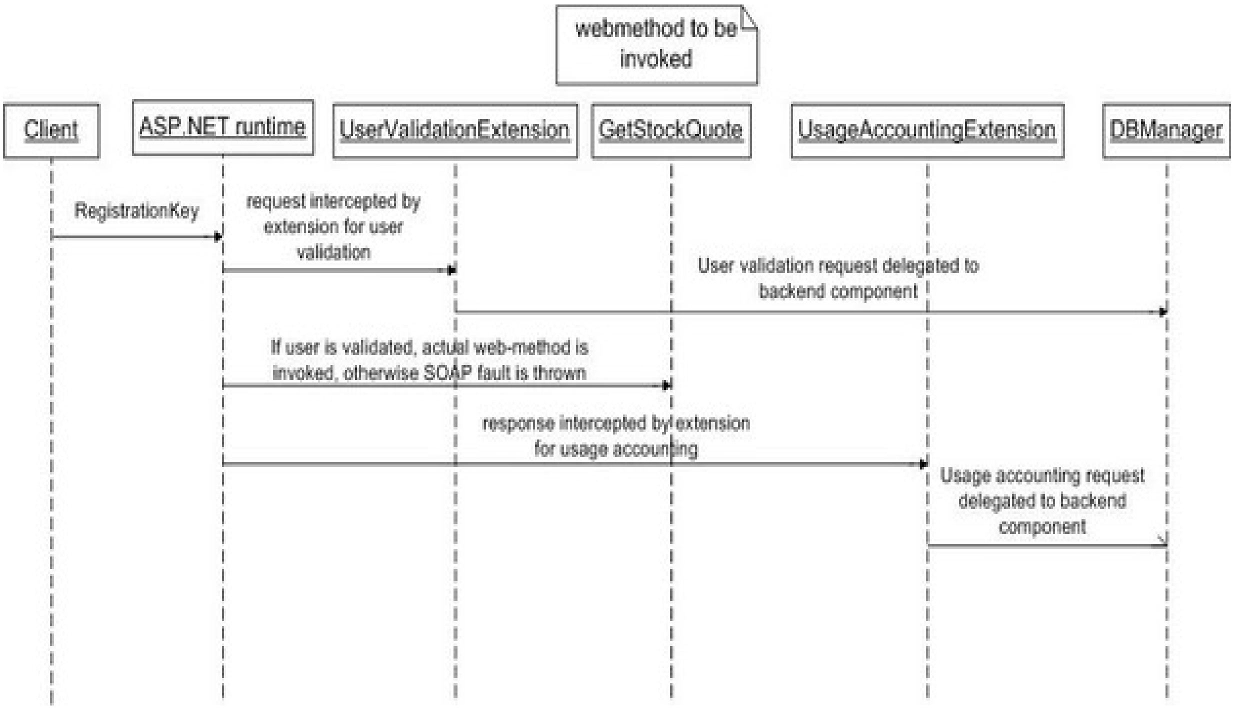

webmethod to be invoked
Client
ASP.NET runtime
UserValidationExtension
GetStockQuote
UsageAccountingExtension
DBManager
RegistrationKey
request intercepted by extension for user validation
User validation request delegated to backend component
If user is validated, actual web-method is invoked, otherwise SOAP fault is thrown
response intercepted by extension for usage accounting
Usage accounting request delegated to backend component

Figure 12: SOAP extension sequence diagram.

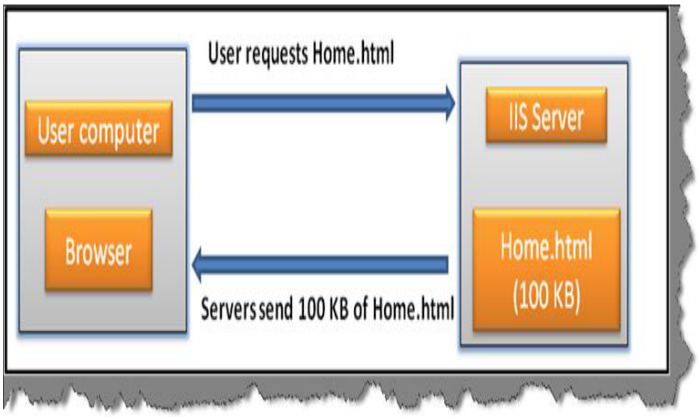


Figure 13: IIS compression.

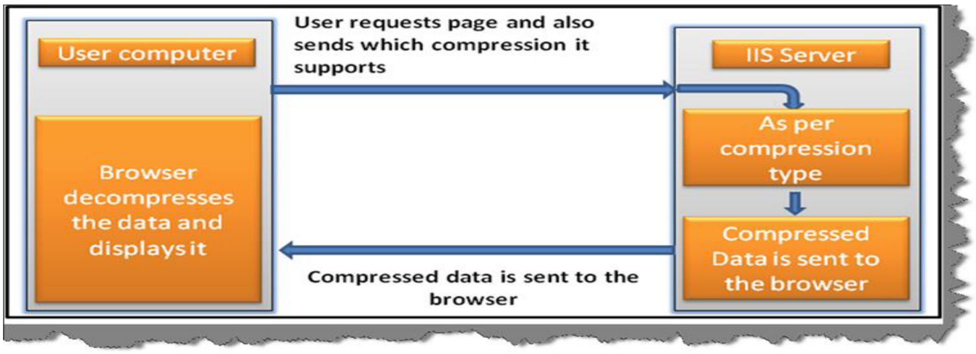

User computer
Browser decompresses the data and displays it
User requests page and also sends which compression it supports
IIS Server
As per compression type
Compressed Data is sent to the browser
Compressed data is sent to the browser

Figure 14: IIS compressed dataflow.

Deflate
=
LZ77
+
Huff Man

Figure 15: Extension over deflate.

- Depending on the compression type support sent by the browser IIS compresses data and sends the same over the wire to the end browser (Figure 14).
- Browser then decompresses the data and displays the same on the browser.

Compression fundamentals: - Gzip and deflate: IIS supports to kind of compressions Gzip and deflate. Both are more or less same where Gzip is an extension over deflate (Figure 15). Deflate is a compression algorithm which combines LZ77 and Huffman coding.

Gzip is based on deflate algorithm with extra headers added to the deflate payload (Figure 16).

Below are the headers details which is added to the deflate payload data. It starts with a 10 byte header which has version number and time stamp followed by optional headers for file name. At the end it has the actual deflate compressed payload and 8 byte check sum to ensure data is not lost in transmission (Figure 17).

Enabling IIS compression

a. **Step 1:- Enable compression**

The first step is to enable compression on IIS. So right click on websites à properties and click on the service tab. To enable compression need to check the below two text boxes from the service tab of IIS website properties. Below figure shows the location of both the checkboxes (Figure 18).

b. **Step 2:- Enable metabase.xml edit**

Metadata for IIS comes from 'Metabase.xml' which is located at "%windir%\system32\inetsrv\". In order to make changes to this XML file need to direct IIS to gives us edit rights. So right click on IIS server root à go to properties and check 'enable direct metabase edit' check box as shown in the below figure (Figure 19).

c. **Step 3:- Set the compression level and extension types**

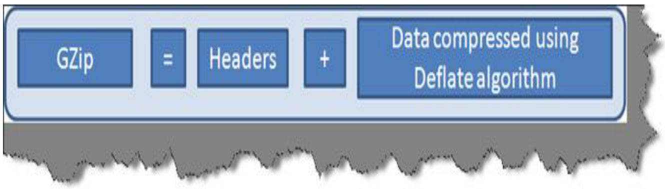


Figure 16: Extension over Gzip.

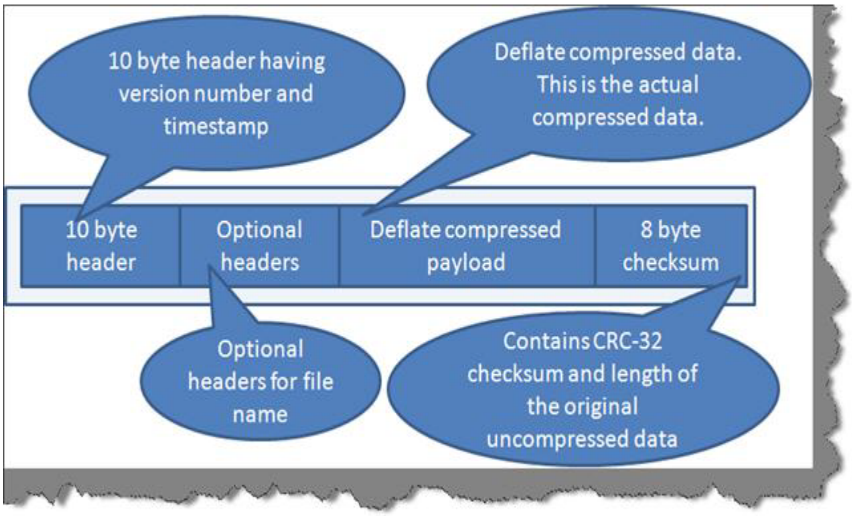
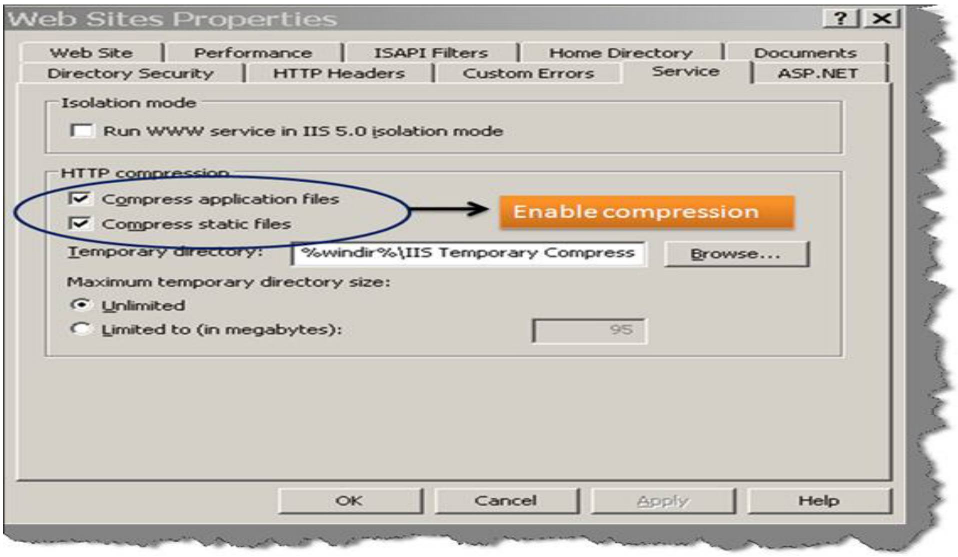

10 byte header having version number and timestamp
Deflate compressed data. This is the actual compressed data.
10 byte header
Optional headers
Deflate compressed payload
8 byte checksum
Optional headers for file name
Contains CRC-32 checksum and length of the original uncompressed data

Figure 17: Deflate payload data.



Isolation mode

□ Run WWW service in IIS 5.0 isolation mode

HTTP compression

☑ Compress application files

☑ Compress static files

Temporary directory: %windir%\IIS Temporary Compress

Maximum temporary directory size:

● Unlimited

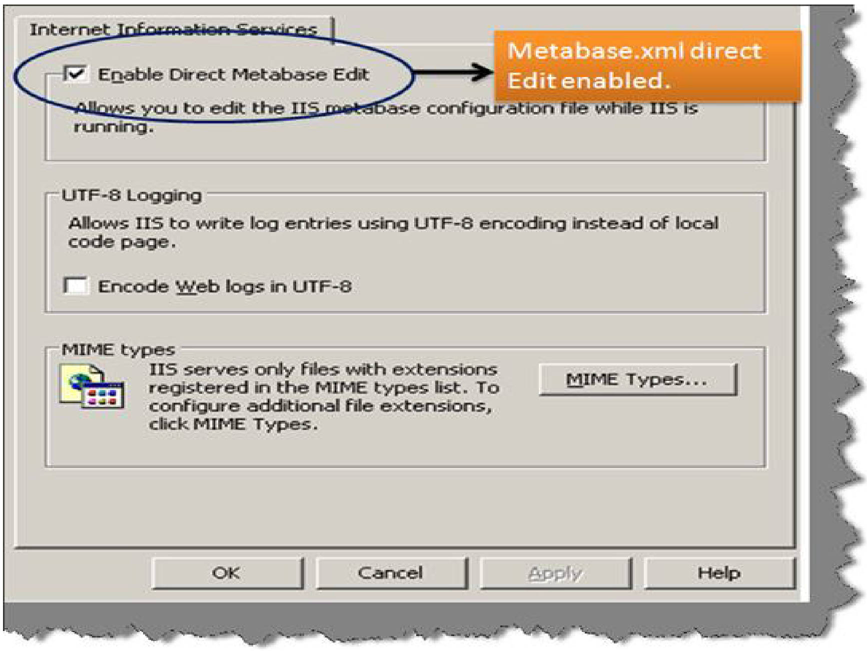
○ Limited to (in megabytes): 95

Figure 18: Enable compression.

Next step is to set the compression levels and extension types. Compression level can be defined between 0 to10, where 0 specifies a mild compression and 10 specifies the highest level of compression. This value is specified using 'HcDynamicCompressionLevel' property. There are two types of compression algorithms 'deflate' and 'gzip' (Figure 20).

Need to also specify which file types need to be compressed. 'HcScriptFileExtensions' help to specify the same. For the current scenario specified that need to compress ASPX outputs before sent to the end browser (Figure 21).

d. **Step 4:- Does it really work?**



Internet Information Services

☑ Enable Direct Metabase Edit

Allows you to edit the IIS metabase configuration file while IIS is running.

UTF-8 Logging

Allows IIS to write log entries using UTF-8 encoding instead of local code page.

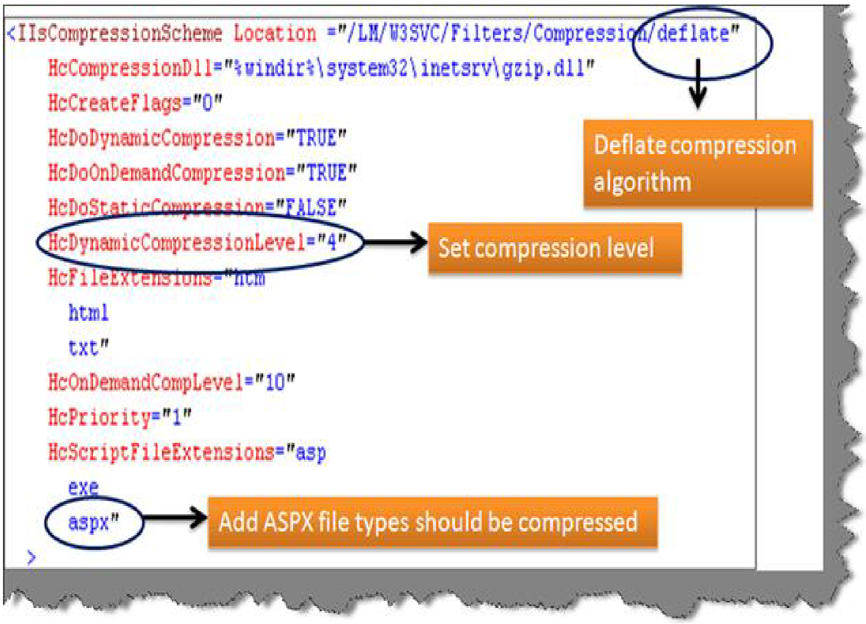
□ Encode Web logs in UTF-8

MIME types

IIS serves only files with extensions registered in the MIME types list. To configure additional file extensions, click MIME Types.

Metabase.xml direct Edit enabled.

Figure 19: Enable matabase.xml edit.



```
<IIsCompressionScheme Location ="/LM/W3SVC/Filters/Compression/deflate"
   HcCompressionDll="%windir%\system32\inetsrv\gzip.dll"
   HcCreateFlags="0"
   HcDoDynamicCompression="TRUE"
   HcDoOnDemandCompression="TRUE"
   HcDoStaticCompression="FALSE"
   HcDynamicCompressionLevel="4"
   HcFileExtensions="htm
     html
     txt"
   HcOnDemandCompLevel="10"
   HcPriority="1"
   HcScriptFileExtensions="asp
     exe
     aspx"
 >
```

Deflate compression algorithm

Set compression level

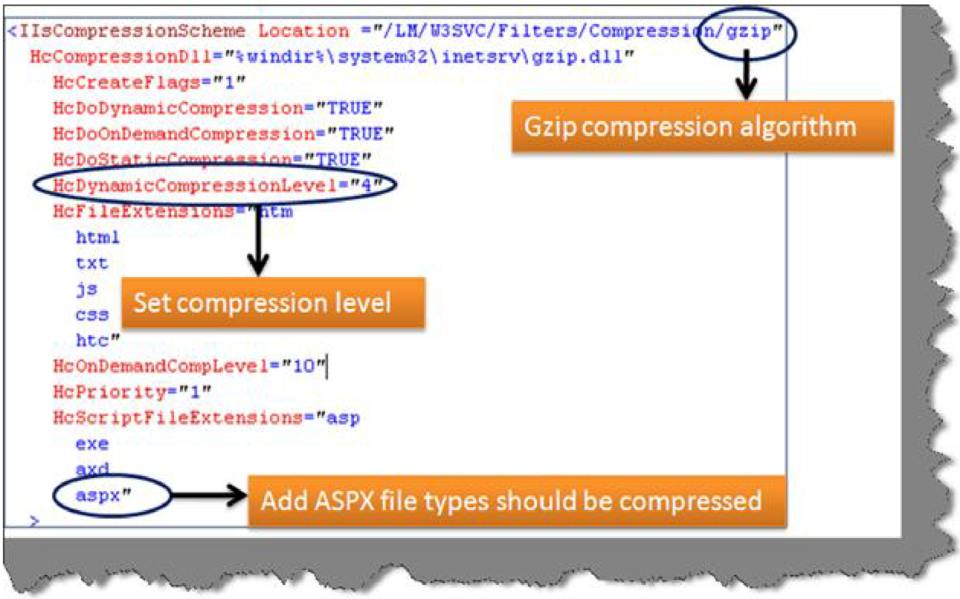
Add ASPX file types should be compressed

Figure 20: Set the compression level.

In order to see the difference before compression and after compression will run the fiddler tool as run ASP.NET loop page. Below screen shows data captured by fiddler without compression and with compression. Without compression data is "80501 bytes" and with compression it comes to "629 bytes". I am sure that's a great performance increase from bandwidth point of view (Figure 22).

If site is only serving compressed data like 'JPEG' and 'PDF', it's probably not advisable to enable compression at all as CPU utilization increases considerably for small compression gains. On the other side need to balance compression with CPU utilization. The more increase the compression levels the more CPU resources will be utilized. Different data types needs to be set to different IIS compression levels for optimization. In the further coming section will take different data types, analyze the same with different compression levels and see how CPU utilization is affected. Below figure shows different data types with some examples of file types (Figure 23).

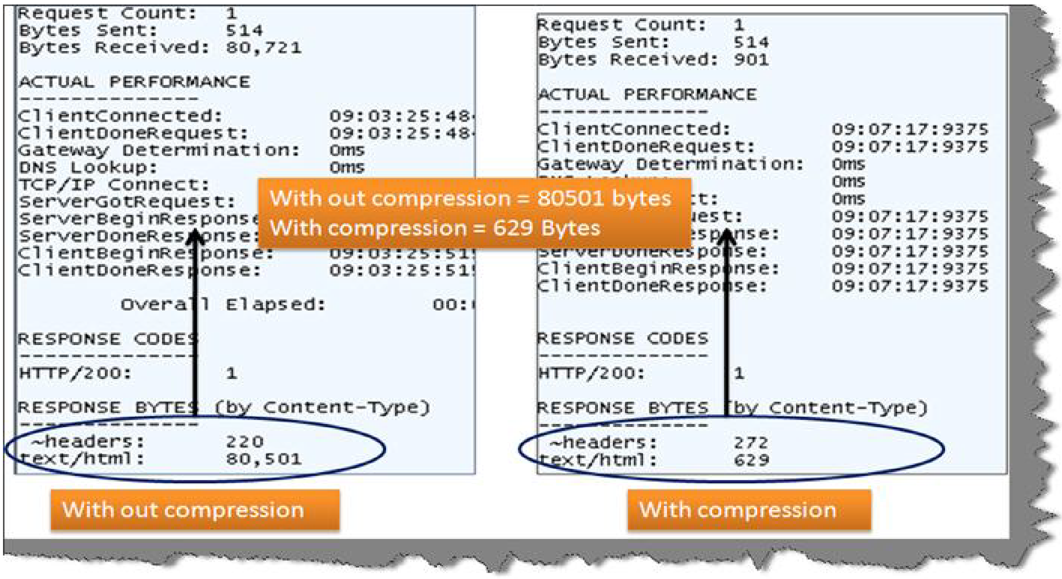
Static data compression: Let's start with the easiest one static content type like HTML and HTM. If a user requests for static page



```
<IIsCompressionScheme Location ="/LM/W3SVC/Filters/Compression/gzip"
  HcCompressionDll="%windir%\system32\inetsrv\gzip.dll"
    HcCreateFlags="1"
    HcDoDynamicCompression="TRUE"
    HcDoOnDemandCompression="TRUE"
    HcDoStaticCompression="TRUE"
    HcDynamicCompressionLevel="4"
    HcFileExtensions="htm
      html
      txt
      js
      css
      htc"
    HcOnDemandCompLevel="10"
    HcPriority="1"
    HcScriptFileExtensions="asp
      exe
      axd
      aspx"
>
```

Gzip compression algorithm
Set compression level
Add ASPX file types should be compressed

Figure 21: Set the extension type.



```
Request Count:   1
Bytes Sent:      514
Bytes Received:  80,721

ACTUAL PERFORMANCE
---------------
ClientConnected:            09:03:25:48
ClientDoneRequest:          09:03:25:48
Gateway Determination:      0ms
DNS Lookup:                 0ms
TCP/IP Connect:
ServerGotRequest:
ServerBeginResponse
ServerDoneResponse:
ClientBeginResponse:        09:03:25:51
ClientDoneResponse:         09:03:25:51

        Overall Elapsed:          00:

RESPONSE CODES
---------------
HTTP/200:        1

RESPONSE BYTES (by Content-Type)
---------------
 ~headers:       220
text/html:       80,501
```

```
Request Count:   1
Bytes Sent:      514
Bytes Received:  901

ACTUAL PERFORMANCE
---------------
ClientConnected:            09:07:17:9375
ClientDoneRequest:          09:07:17:9375
Gateway Determination:      0ms
                            0ms
              t:            0ms
              uest:         09:07:17:9375
              sponse:       09:07:17:9375
ServerDoneResponse:         09:07:17:9375
ClientBeginResponse:        09:07:17:9375
ClientDoneResponse:         09:07:17:9375

RESPONSE CODES
---------------
HTTP/200:        1

RESPONSE BYTES (by Content-Type)
---------------
 ~headers:       272
text/html:       629
```

With out compression = 80501 bytes
With compression = 629 Bytes
With out compression
With compression

Figure 22: Compression performance.

from IIS who has compression enabled, IIS compresses the file and puts the same in '%windir%\IIS Temporary Compressed Files' directory. Below is a simple screen which shows the compressed folder snapshot. Compression happens only for the first time. On subsequent calls for the same compressed data is picked from the compressed files directory (Figure 24).

Below are some sample readings taken for HTML files (Table 1) of size range from 100 KB to 2048 KB and set the compression level to '0' (Figure 25).

Dynamic data compression: Dynamic data compression is bit different than static compression. Dynamic compression happens every time a page is requested. So, the users need to balance between CPU utilization and compression levels (Table 2).

The above readings do not show anything specific, its bit messy. So plotted the graph using the above data and hit the sweet spot. Even after increasing the compression level from 4 to 10 the compressed size has no effect. So the conclusion from this is, setting value '4' compression level for dynamic data pages will be an optimized setting (Figure 26).

Compressed file and compression: Compressed files are file which are already compressed. For example files like JPEG and PDF are already compressed. The compressed files after applying IIS compression did not change much in size.

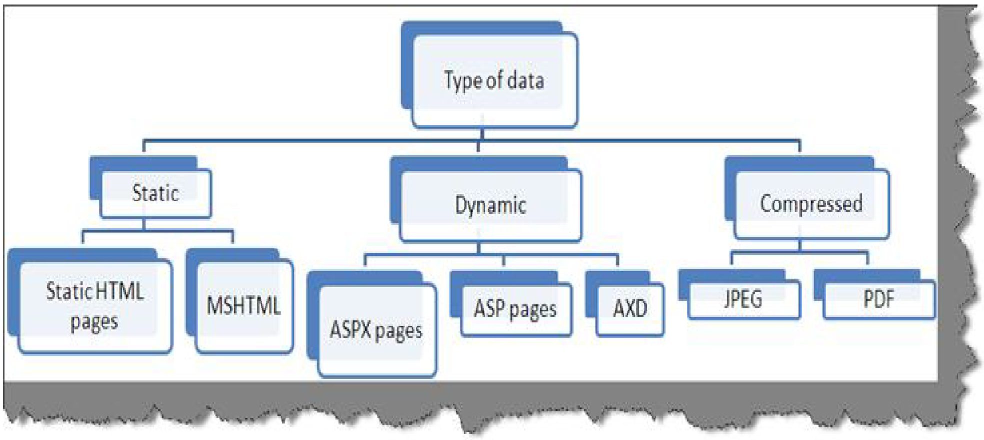
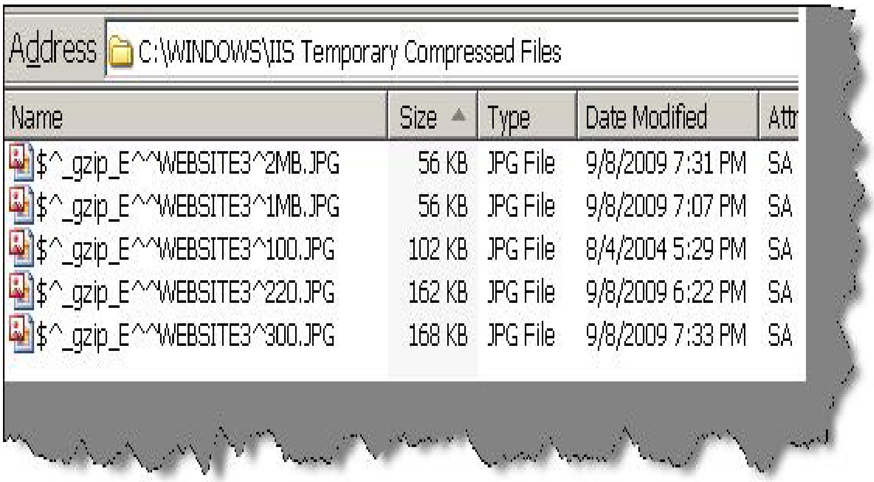


Figure 23: Different data types.



Address C:\WINDOWS\IIS Temporary Compressed Files

Name	Size	Type	Date Modified	Attr
$^_gzip_E^^WEBSITE3^2MB.JPG	56 KB	JPG File	9/8/2009 7:31 PM	SA
$^_gzip_E^^WEBSITE3^1MB.JPG	56 KB	JPG File	9/8/2009 7:07 PM	SA
$^_gzip_E^^WEBSITE3^100.JPG	102 KB	JPG File	8/4/2004 5:29 PM	SA
$^_gzip_E^^WEBSITE3^220.JPG	162 KB	JPG File	9/8/2009 6:22 PM	SA
$^_gzip_E^^WEBSITE3^300.JPG	168 KB	JPG File	9/8/2009 7:33 PM	SA

Figure 24: Static data compression.

The compression benefits are very small. End up utilizing more CPU processor resource and gain nothing in terms of compression (Table 3 and Figure 27).

So the conclusion can draw for compressed files is that can disable compression for already compressed file types like JPEG and PDF.

Proposed System Result

The result of the Intrusion Detection Expert System (IDES) has become a regular in intrusion detection systems. Many current systems are based in partly on IDES prototype technology, The Next -Generation Intrusion Detection Expert System (NIDES) is the comprehensive enhancement to IDES. NIDES is a real-time intrusion detection application which integrates a statistical analysis -based anomaly detector and a rule-based misuse detection system [21]. The combination gives the flexibility to observe penetrations from internal and external attacks. NIDES additionally includes a comprehensive program that allows access to any or all the applications capabilities, similarly as a context -sensitive facilitate system.

While NIDES is considered the present progressive during a combined anomaly and misuse detection system, the applying retains the problem possessed by all similar models in detection cooperative attacks, long-term penetration situations and virus propagation. Another potential disadvantages is that NIDES retains the reliance on the system's audit record for input.

Future expansions of the rulebase and the development of profiles of entities aside from users ought to scale back the potential vulnerabilities that don't seem to be adequately addressed by the present system.

One of the main challenges to attempting implement and validate a new intrusion detection methodology, is to assess it and compare its performance therewith of alternative out there approaches. It is noticeable that this task is not restricted to A-NIDS, however is additionally applicable to NIDS (and even to IDS sometimes) generally. The requirement for test-beds that provide robust and reliable metrics to quantify NIDS has been prompt, for instance, by the National Institute for Standards and Technology (NIST). An advantage of assessment in real environments is that the traffic is sufficiently realistic; however, this approach subject to: (a) the risk of potential attacks, and (b) the possible interruption of the system operation due to simulated attacks. On the other hand, the evaluation of NIDS methodologies in experimental environments involves the generation of synthetic traffic as well as background traffic representing legal users, that is much from being a trivial endeavor.

Actual KB	Compressed in KB
100	24
200	25
300	27
1024	32
2048	41
Compression level set to '0'	

Table 1: File Compression range.

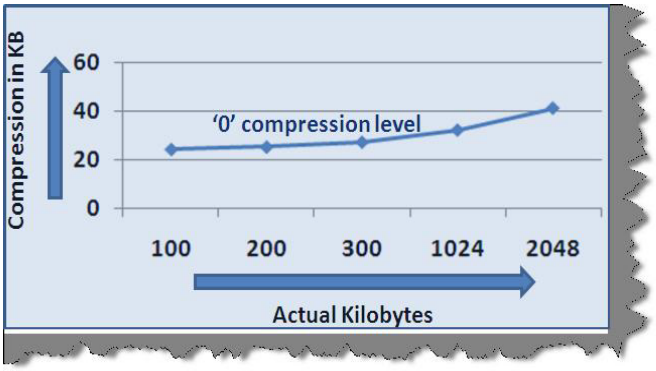


Figure 25: Compression level chart.

	File size				
Compression Levels	**100 KB**	**200 KB**	**300 KB**	**1 MB**	**2 MB**
0	32,774	35,496	37,699	52,787	109,382
1	30,224	32,300	34,104	46,328	92,813
2	29,160	31,004	32,673	43,887	87,033
3	28,234	29,944	31,628	42,229	83,831
4	26,404	27,655	29,044	34,632	44,155
5	25,727	26,993	28,488	33,678	42,395
6	25,372	26,620	28,488	33,448	41,726
7	25,340	26,571	28,242	33,432	41,678
8	25,326	26,557	28,235	33,434	41,489
9	24,826	26,557	28,235	33,426	41,490
10	24,552	25,764	27,397	32,711	42,610

Table 2: CPU utilization and compression levels.

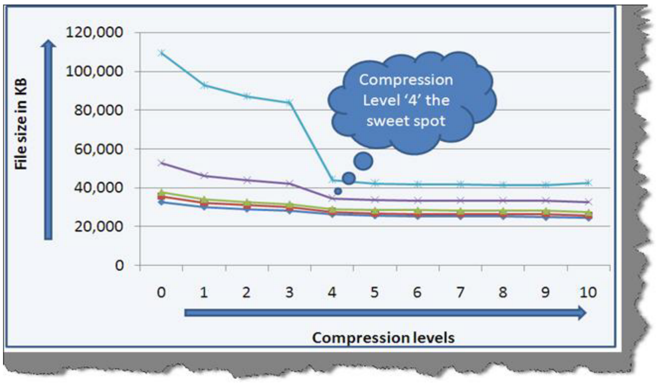


Figure 26: Dynamic data compression report chart.

Other network traffic related studies deal with the problem of standardizing the acquisition and use of real traffic for validating NIDS environments. During this respect, that contributes some proposals on a general methodology to amass and organize traffic datasets, to outline AN analysis framework to check the performance of anomaly-based NIDS [19,22]. The significant try created up to now in NIDS assessment proof of its importance. However, it remains an open issue and a big challenge.

A new methodology that would achieve more accuracy than the existing six classification patterns (Gaussian Mixture, Radial Basis Function, Binary Tree Classifier, SOM, ART and LAMASTAR),called Hierarchical Gaussian Mixture Model[HMM] for IDM. Development of host-based anomaly intrusion detection, focusing on system call based HMM training. This was later enhanced with the inclusion of data pre-processing for recognizing and eliminating redundant sub sequences of system calls, resulting in less number of HMM sub models. Experimental results on three public databases showed that training cost can be reduced by 50% without affecting the intrusion detection performance. False alarm rate is higher yet reasonable compared to the batch training method with a 58% data reduction. An anomaly detection system comprising of detection modules for detecting anomalies in each layer. The anomaly detection results of the neighbor node(s) is taken by the current node and its result in turn is sent to the neighbor node(s).Experimental results revealed increased detection rate and reduced false alarm positives, compared to other methods [23].

The new framework builds the patterns of network services over datasets labeled by the services [22]. With the built patterns, the framework detects, attacks in the datasets. This approach is independent of attack-free training datasets, but assumes that each network service has its own pattern for normal activities.

The biometrics-based intrusion detector model to provide a light-

Actual compressed file size	File size after IIS compression
100	102
220	210
300	250
1024	980
2048	1987

Table 3: IIS compression.

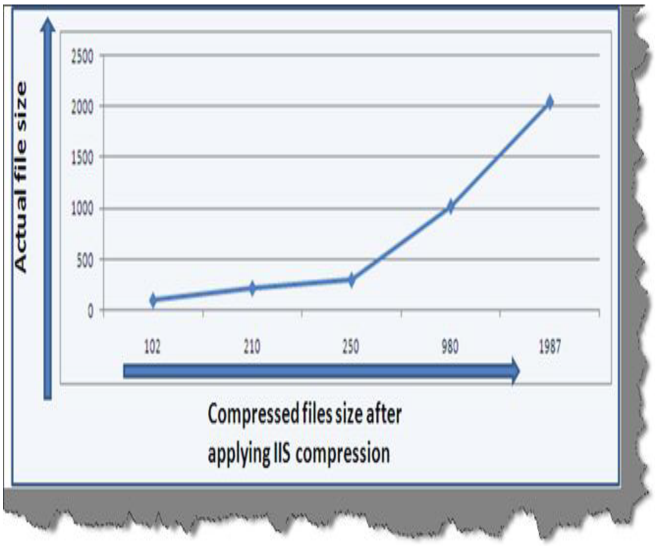


Figure 27: Compression files size after applying IIS compression.

weight and self-contained module for detection user identities misuse. System-calls and network traffic monitoring systems ought to be combined to the present detector to achieve the best solutions.

The proposed a technique to detect anomalies at all layers of a network stack in a sensor network, segregating the service at various levels. Physical layer intrusion is detected by using RSSI values of neighbors (dependant on background noise, weather conditions etc). Targeting MAC layer will work for schedule based and sleep/wake-up based MAC protocols whereas IASN protocol is geared toward the routing layer.

Experiments show that IASN is used for supply started routing protocols, table driven routing protocols and data dissemination mechanisms like directed diffusion. The probability of detection increases linearly with the amount of nodes running IASN. Nodes guard each other from masquerade at application layer. Depending on the resource availability, any combination of the above methods can be employed, as they are independent of one another. All technique are energy efficient as they have very low false positive rates (except RSSI and round trip time) and low overhead.

Combining multiple independent data sources and studying combined traditional intrusion attack and anomaly intrusion, the anomaly intrusion traffic detection provided the statistical wavelet based detection mechanism. The properties like attack length, packet count, packet rate, and dominant protocol kind match with the two data sets, as is showed by attack structure. At lean and significant traffic situations, the demand capability of the server was determined to administer higher clarity of anomaly intrusion detection though server period of time. Analysis of many traffic anomaly properties that is not possible victimization ancient intrusion measurements is performed by a brand new model that used anomaly intrusion attack measurements.

Windows Host Anomaly Detection System, which is used as a supplement for other security mechanisms under windows. It can only detect intrusions which invoke an anomaly sequence by programs. The Statistical anomaly detection technology called that HIDE with hierarchical multitier multi-observation window system to monitor network traffic parameters simultaneously, using a real-time probability distribution function (PDF) for each parameter, collected during the observation window. The similarity measurements of measured PDF and reference PDF are combined into an anomaly status vector classified by a neural network. This technique detects that attacks and soft faults with traffic anomaly intensity as low as 3 to 5 percent of typical background traffic intensity, thereby generating an early warning.

The anomaly based mostly intrusion detection system for mobile networks, supported simulation results of quality profiles for enhancing ABID in mobile wireless networks. If the quality behavior of users has not been accurately found, the choice of specific values for key parameters, like sequence length and cluster size is absurd. One potential strategy for enhancing the characterization of users and addressing construct drift (keeping official up-to-date), is to take care of a window of the fresh determined sequences (analogous to the exponential weighted moving average) which will then be wont to update the coaching patterns sporadically and, thus scale back the false positives.

An intrusion detection algorithm and its architecture (two layered, global central layer and a local layer, together performing data collection, analysis and response), based on data mining and useful in real time for network security, By filtering out the known traffic behavior (intrusive and normal) the IDS focuses on analysis on unknown data thereby reducing false alarm rates. The model supported contiguous professional selection rule ways observe most anomalies, unsuccessful match does not imply AN abnormity, as traditional rules might not cowl all traditional data. Detection rates in this is not commendable but it has vast future scope for improvement.

In recent literature, anomaly detection through a Bayesian Support Vector Machine is found as interesting machine learning model for anomaly detection. Use of a SVM with one-class to detect the system anomalies at their early stage is studied along with drift output classification probabilities. Experimentally, absence of failure training data under one-class SVM leads to quick detection of unknown anomalies. Initially dividing the training data into multiple unrelated lower dimensional models, the test data will be evaluated on each model separately thereby revealing outliers in different capacities (as is used to evaluate the posterior class probabilities in Bayesian framework) [24-27].

Results of the Various Algorithms

The experimental results that have obtained with the assorted algorithms. the assorted rules designed and whose results are bestowed the Brute-Force algorithm, the Karp-Rabin rule, the Boyer-Moore rule and also the Knuth-Morris-Pratt rule.

The results of the running time of these algorithms vary the input size, where the input is the words. The number of patterns to be matched remains the same. The running time (in milliseconds) for the various algorithms is recorded within the following Table 4.

From the Table 4 and the graph (Figure 28), there is no trend within the performance between the algorithms. Solely that KMP performs the worst and Boyer-Moore performs the simplest.

Conclusion

The extension of the model with additional attributes can help to unearth further mistakes. The analysis of statistical properties of router configurations appears to be a promising approach to help operators in detecting mistakes. Unlike most of the current research, which use only one agent per engine for detection of various attacks, the proposed system is constructed by several agents in a single engine. The NIDS will broaden its read on completely different behaviors of the network traffic by every of the agents with its own strength on capturing a form of network behavior. Firewall policy rules are one in every of most significant part of network security system. It plays the very important role in management of any organization's network and its security infrastructure.

Thus the management of policy rule could be a vital task for the network security. There are many tools and techniques won't to perform anomaly detection and rule editing by using given set of existing policy rules. However, one in every of the idea and so its limitation is that

Input Size	Running Time (in milliseconds)			
	Brute-Force	Karp-Rabin	Knuth-Morris-Pratt	Boyer-Moore
20000	15	15	17	15
60000	46	45	46	40
100000	74	73	79	68
140000	102	102	108	102
180000	129	132	139	119
200000	144	144	159	133

Table 4: Running time for various algorithms.

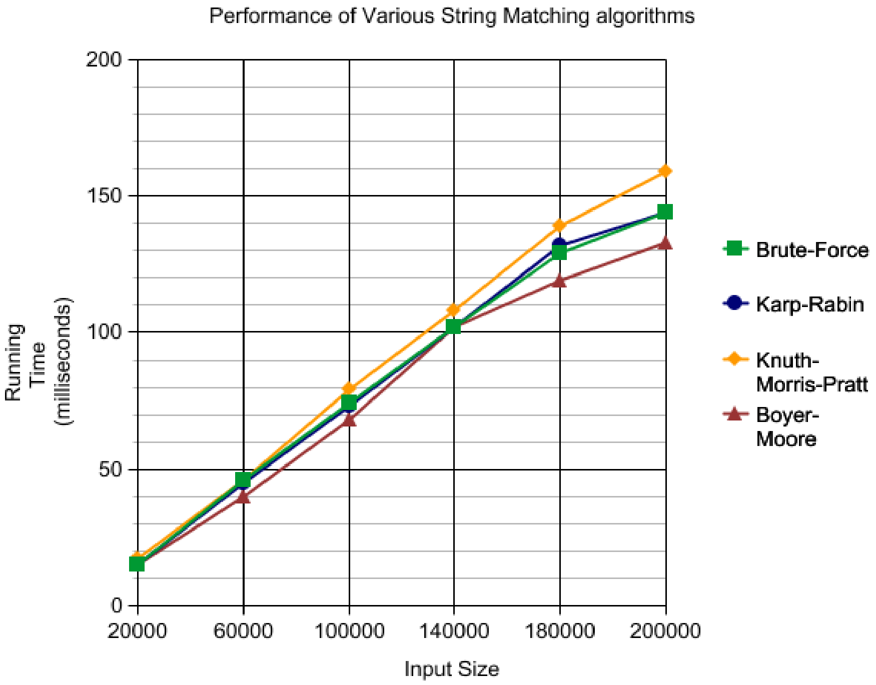


Figure 28: Performance between the Algorithms.

firewall and its rules are set to be static and so while not a capability to replicate the network behavior determined by firewall.

References

1. Shirbhate RS, Patil PA (2012) Network Traffic Monitoring Using Intrusion Detection System. International Journal of Advanced Research in Computer Science and Software Engineering 2: 1-5.

2. Raju B, Srinivas B (2012) Network Intrusion Detection System Using KMP Pattern Matching Algorithm. International Journal of Computer Science and Telecommunications 3: 33-36.

3. Shrivastava N, Richariya V (2012) Ant Colony Optimization with Classification Algorithms used for Intrusion Detection. IJCEM International Journal of Computational Engineering & Management 15: 54-63.

4. Ranjan R, Sahoo G (2014) A New Clustering Approach for Anomaly Intrusion Detection. International Journal of Data Mining & Knowledge Management Process (IJDKP) 4: 29-39.

5. Kumar N, Angral S, Sharma R (2014) Integrating Intrusion Detection System with Network Monitoring. International Journal of Scientific and Research Publications 4: 1-4.

6. Shelokar ND, Ladhake SA (2010) Network Intrusion detection using correlation functional dependency. Oriental Journal of Computer Science & Technology 3: 185-188.

7. Sivakumar V, Yoganandh T, Mohan Das R (2012) Preventing Network From Intrusive Attack Using Artificial Neural Networks. International Journal of Engineering Research and Applications (IJERA) 2: 370-373.

8. Kabila R (2008) Network based Intrusion Detection and Prevention Systems in IP-Level Security Protocols. World Academy of Science, Engineering and Technology 2: 661-667.

9. Prasad KM, Mohan Reddy AR, Jyothsna V (2012) IP Traceback for Flooding attacks on Internet Threat Monitors (ITM) Using Honeypots. International Journal of Network Security & Its Applications (IJNSA) 4: 13-27.

10. Al-Dabagh NB, Fakhri MA (2014) Monitoring and Analyzing System Activities Using High Interaction Honeypot. International Journal of Computer Networks and Communications Security 2: 39-45.

11. Laing B (2000) Internet Security Systems, How to guide –Implementing a network based Intrusion Detection System.

12. Yadav MR, Kumbharkar PB (2014) Intrusion Detection System with Supervised Learning Algorithms. International Journal of Advanced Research in Computer Science and Software Engineering 4: 305-310.

13. Khari M, Gaur M, Tuteja Y (2013) Meticulous Study of Firewall Using Security Detection Tools. International Journal of Computer Applications & Information Technology 2: 1- 9.

14. Sharma S, Kumar S, Kaur M (2014) Recent trend in Intrusion detection using Fuzzy-Genetic algorithm. International Journal of Advanced Research in Computer and Communication Engineering 3: 6472-6475.

15. Gaidhane R, Vaidya C, Raghuwanshi M (2014) Survey: Learning Techniques for Intrusion Detection System [IDS]. International Journal of Advance Foundation and Research in Computer [IJAFRC] 1: 21-28.

16. Dixit U, Gupta S, Pal O (2012) Speedy Signature Based Intrusion Detection System Using Finite State Machine and Hashing Techniques. International Journal of Computer Science Issues 9: 387-391.

17. Goel R, Sardana A, Joshi RC (2012) Parallel Misuse and Anomaly Detection Model. International Journal of Network Security 14: 211-222.

18. Meenatchi I, Palanivel K (2014) Intrusion Detection System in MANETS: A Survey. International Journal of Recent Development in Engineering and Technology 3: 42-50.

19. Shuang-can Z, Chen-jun H, Wei-ming Z (2014) Multi Agent Distributed Intrusion Detection System Model Based on BP Neural Network. International Journal of Security and Its Applications 8: 183-192.

20. Jawale DR, Bhusari VK (2014) A Novel Approach for classification and Detection of attacks in Network Intrusion Detection System Using ANN. International Journal of Advanced Research in Computer Science and Software Engineering 4: 802-806.

21. Cannady J, Harrell J A Comparative Analysis of Current Intrusion Detection Technologies.

22. Jaisankar N, Saravanan R, Swamy KD (2009) Intelligent Intrusion Detection System Framework Using Mobile Agents. International Journal of Network Security & its Applications (IJNSA) 1: 72-88.

23. Saini P, Godara S (2014) Modelling Intrusion Detection System using Hidden Markov Model: A Review. International Journal of Advanced Research in Computer Science and Software Engineering 4: 542-548.

24. Das V, Pathak V, Sharma S, Sreevathsan, Srikanth MVVNS, et al. (2010) Network Intrusion Detection System based on Machine Learning Algorithms. International Journal of Computer Science & Information Technology (IJCSIT) 2: 138-151.

25. Mudholkar SS, Shende PM, Sarode MV (2012) Biometrics Authentication Technique for Intrusion Detection Systems Using Finger Print Recognition. International Journal of Computer Science, Engineering and Information Technology (IJCSEIT) 2: 57-65.

26. Golmah V (2014) An Efficient Hybrid Intrusion Detection System based on C5.0 and SVM. International Journal of Database Theory and Application 7: 59-70.

27. Khanbabapour H, Mirvaziri H (2014) An Intelligent Intrusion Detection System Based on Expectation Maximization Algorithm in Wireless Sensor Networks. International Journal of Information and Communication Technology Research 4: 1-10.