Detecting Insider Attacks on OSPF

using statistical anomaly detection

(Master Thesis Project)

by

R.C. Koning

July 1999

Delft University of Technology
Faculty of Information Technology and Systems
Telecommunication and Traffic Control Systems Group

Professor at Delft University of Technology
Prof.ir. J.W.J. van Till

Members of the Graduation Committee
Ir. J.A.M. Nijhof (Delft University of Technology)
Ir. H.A.M. Luiijf (TNO)

Mentor at AT&T Laboratories
Prof.dr. E.G. Amoroso

Detecting Insider Attacks on OSPF[†]

using statistical anomaly detection

René C. Koning

Telecommunication and Traffic-Control Systems Group
Faculty of Information Technology and Systems
Delft University of Technology, Delft, The Netherlands
r.c.koning@its.tudelft.nl
August 1999

Abstract

This paper describes seven insider attacks, amongst which four new ones, on the Internet Protocol based Open Shortest Path First (OSPF) routing protocol. The attacks can cause severe degradation of network service and even complete denial of service. The attacks were implemented and launched on a small test network. To guard against the attacks an intrusion detection system is proposed, described and implemented. The statistical anomaly algorithms from the Next-Generation Intrusion Detection Expert System (NIDES) developed by SRI International is used as a basis for the developed intrusion detection system. The implementation of the intrusion detection system is done in N-code as Network Flight Recorder (NFR) back-ends. The results that were achieved with the described approach show that the intrusion detection system is very effective in detecting the seven implemented attacks.

1. Introduction

Special-purpose computers called routers form the heart of many network infrastructures. As data is forwarded from one place in the network to another, it is the routers that make the decisions as to where and how the data is forwarded. The rule set that dynamically informs the routers of the paths that the data should take are called routing protocols. It is the job of these protocols to react quickly to changes in the network's infrastructure, such as transmission lines going in and out of service, routers crashing and so on.

Many of today's networks are based on distance-vector routing protocols, particularly the Routing Information Protocol (RIP) [1]. However, the rapid growth and expansion of today's networks has pushed RIP to its limits. RIP has certain limitations that could cause problems in large networks [2]. To resolve the limitations of RIP, the OSPF (Open Shortest Path First) protocol [3] was developed by the IETF. Both protocols are interior gateway protocols (IGP) that exchanging routing information between routers within an autonomous network. The OSPF protocol is based on link-state technology, which is a departure from the Bellman-Ford vector-based algorithms used in traditional Internet routing protocols such as RIP. While OSPF is getting more and more established in the networking community there are already many networks running OSPF today, for example the AT&T WorldNet network with more than 1.5 million subscribers is routed by OSPF.

The consequence of the ever-growing networks is that it has brought with it an increased reliance on the network infrastructure. When the routing infrastructure is attacked the network service can be degraded or, in the worst case, denied. Instead of attacking just one particular host or even a subnet, which is usually the focus of traditional host and network security, the purpose of routing attacks is to sabotage the network service and infrastructure. Until recently routing protocol developers did not recognize the need for secure routing protocols. Even today, research in routing protocol attacks is still in its infancy.

Not nefarious outsiders, but insiders are the most common source of attacks. In the business community insiders have been blamed for causing 70 to 80 percent of the incidents and most of the damage [4][5]. Insider attacks are very hard to guard against. One approach to prevent insider attacks is to digitally sign the exchanged information [6]. The objective of this prevention approach is to guarantee the integrity and authenticity of the data. However, the prevention approach has been rejected by the

[†] The research described in this paper was done as a master thesis project for Delft University of Technology in The Netherlands. The actual research was done at AT&T Laboratories located in Florham Park, New Jersey, USA.

IETF for the OSPF protocol for both technical and political reasons, including: complexity, high overheads and backward compatibility.

Since insider attacks are very hard to guard against, the best security engineers can do today, is to study the various indicators from the network itself, which might signal the occurrence of an attack. This leads to the approach of defending routers against insider attacks with intrusion detection systems (IDS). Intrusion detection is the process of identifying and responding to malicious activity targeted at computing and networking resources. The intrusion detection approach is relatively more acceptable to the industry as it requires no changes to routers or the routing protocols.
The focus in this paper is on detecting attacks against the OSPF routing protocol and in particular insider attacks. For that purpose an inventory of the various security aspects of the protocol was made as well as an investigation on how to detect the attacks.

A project, called *Jinao* [7][8], conducted by a research team at North Carolina State University (NCSU) and Microelectronics Center of North Carolina (MCNC) was used as a starting point for the research. Their project demonstrated that the statistical component of the Next Generation Intrusion Detection Expert System (NIDES) [9], developed at SRI, was effective in detecting routing protocol attacks. They had discovered three vulnerabilities in the OSPF protocol. We adopted their approach and attacks. A further analysis of the OSPF protocol resulted in four vulnerabilities in the protocol. The vulnerabilities were exploited and with these attacks an insider is able to route traffic through a predetermined node (perhaps for sniffing or monitoring purposes) or route the traffic along a non-optimal path in order to deny, degrade or delay network services.

To guard against the implanted attacks an intrusion detection system was developed. The implementation was done in N-code as back-ends for the commonly available framework Network Flight Recorder (NFR) [10]. The NFR back-ends can run on arbitrary nodes in an OSPF network. A modified version of the NIDES statistical anomaly detection algorithm forms the heart of the back-ends.

As will be presented in this thesis, the implemented attacks can be very hazardous to the network performance. The presented intrusion detection proves to be very effective in detection the insider attacks on the OSPF protocol.

2. OSPF Basics

Open Shortest Path First (OSPF) is a routing protocol developed for Internet Protocol (IP) based networks by the interior gateway protocol working group of the Internet Engineering Task Force (IETF). This section serves only as a reminder of the operations of the OSPF routing protocol, a complete description can be found in [3].

OSPF is designated to be run internal to a single Autonomous System. The OSPF protocol is based on link-state technology this implies that each OSPF router maintains an identical database, which describes the Autonomous System's topology. From this database, a routing table is calculated by constructing a shortest path tree, using Dijkstra's algorithm [11].

2.1. Hierarchical routing

One of the most important features of the OSPF protocol is its capability to use a hierarchical routing structure. Using hierarchical routing the routing table sizes usually decrease from $O(n)$, typical for flat routing, to $O(\log(n))$, where n is the number of involved routers. OSPF implements a two-level hierarchy routing scheme trough the deployment of OSPF areas. The two-level hierarchy consists of a backbone area, which interconnects all other areas. When the Autonomous System is split into OSPF areas, the routers are also further divided. Four different types of routers (backbone, internal, area border and autonomous system border) designate the hierarchical routing structure used by OSPF (see Figure 1). Each has a unique role and set of defining characteristics with the hierarchy.

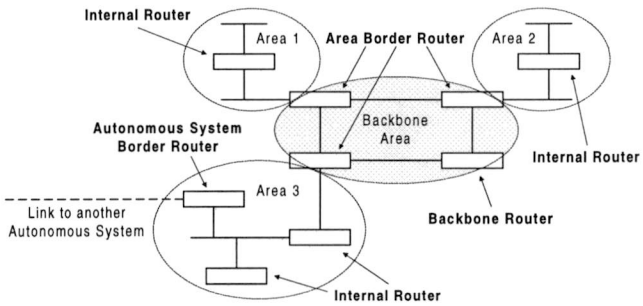

Figure 1 OSPF Hierarchy.

In order to minimize the amount of information exchange on a particular segment, OSPF elects one router to be a designated router (DR) and one router to be a backup designated router (BDR) on each multi-access segment. The BDR is elected for redundancy reasons, whenever the DR fails the BDR will take over. The idea behind designated routers is that routers have a central point of contact for

information exchange. Instead of each router exchanging updates with every other router on the segment, every router will exchange the information with the DR. The DR will relay the information to all other routers in the segment. This would cut the information exchange from $O(n^2)$ to $O(n)$ where n is the number of routers on a multi-access segment.

2.2. OSPF packets

The OSPF protocol uses 5 different types of packets. Each packet has its own distinct function in the OSPF protocol (see Table 1). OSPF uses *Hello* (1) packets to discover and maintain neighbor relationships using OSPF's Hello protocol. The *Database Description* (2) and *Link State Request* (3) packets are used in the forming of adjacencies. OSPF's reliable update mechanism is implemented by using the *Link State Update* (4) and *Link State Acknowledgment* (5) packets.

Table 1 OSPF packet types.

	Packet name	Protocol function
1	Hello	Discover/maintain neighbors
2	Database Description	Summarize database contents
3	Link-State Request	Database download
4	Link-State Update	Database update
5	Link-State Acknowledge	Flooding acknowledgment

The OSPF protocol runs directly over the Internet Protocol (IP), using protocol 89 in the IP header field. OSPF runs on OSI's network layer, therefore all OSPF packets are solely encapsulated by IP and local data-link headers.

2.3. Link State Advertisements

Link-State Advertisements (LSA) are used by OSPF to provide routing information to other OSPF routers. The collected link state advertisements of all routers and networks forms the protocol's link-state database. For a router, this advertisement includes the state of the router's interfaces and adjacencies. Each link state advertisement is flooded throughout the routing domain. There are six different and distinct link-state formats. Each one of these advertisements is generated for a different purpose that helps keep the routing table intact and accurate.

The 20-byte header of the LSA contains enough information to uniquely identify the LSA. This information is required when multiple instances of the LSA exist in the routing domain at the same time. It is then necessary to determine which instance is more recent. This is accomplished by examining the LS Age, LS sequence number and LS checksum fields that are contained in the LSA header.

2.4. OSPF security

Since routing is the core heart of the network infrastructure, integrity and authenticity is of uttermost importance. All OSPF protocol exchanges can be authenticated. The OSPF packet includes an authentication type field, and 64-bits of data for use by the appropriate authentication scheme. However, the IETF OSPF working group has rejected a strong authentication scheme. Nevertheless OSPF has three intrinsic mechanisms that make it very robust and resilient to failures, even to some malicious attacks on the protocol.

2.4.1. Flooding and information least dependency

To propagate the LSAs OSPF uses a flooding mechanism. This reliable mechanism ensures that all routers in the same area have the same topological database. In case of either a single point (router) failure or an intruder trying to fake or modify other router's information, as long as there is an alternate path, 'healthy' routers can always receive messages, though they could be conflicting messages. This triggers an interesting feature of the OSPF protocol: fight-back. When a router receives incorrect information about it itself or its links it will fight-back this information by sending out the correct information.

Another consequence of flooding individual LSAs is *information least dependency*. This means that every router only uses the information from the original advertiser instead of aggregated information from neighbors. The originator can be retrieved from the OSPF header field that contains the router ID of the router. Each router is assigned an unique router identification (ID) number, similar to an IP address. This method gives security advantages over other (pure distance vector based) routing protocols.

2.4.2. Hierarchical routing and information hiding

The primary goal of hierarchical routing is to deal with routing scalability issues (reduce the routing table size, link bandwidth and router computing resources). But, it also has both robustness and security advantages. The implication of two-level routing is that an internal router does not need to know the topology of the outside area but only its own area. If an internal router is compromised, the damage it can do is limited to the area, leaving routers in other areas functioning normally.

2.4.3. Procedural checking and constraint

The checking procedure for OSPF protocol to accept a packet is rigorous. Generally, it must pass three checking gates (IP, OSPF header and OSPF packet) as depicted in Figure 2. The validation procedures are necessary for a protocol to operate correctly and be robust. On the other side the procedures increase

the difficulty of sabotage. The chance of simply pitching some fields in a packet and hoping it works is rare.

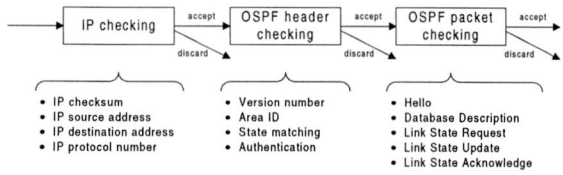

Figure 2 OSPF packet checking procedure.

3. ATTACKS

Three vulnerabilities were discovered by a research group of North Carolina State University and Microelectronics Center of North Carolina. They exploited the vulnerabilities with attacks called the Seq++, MaxSeq and MaxAge attack. After scrutinizing the OSPF protocol four more vulnerabilities came to light. The attacks will be described in detail later in this section.

3.1. Attack environment

The attacks were launched on a small isolated test network where each node in the network was configured as a router. The network was solely used for the OSPF attacking project. The topology of the test-bed is depicted in Figure 3.

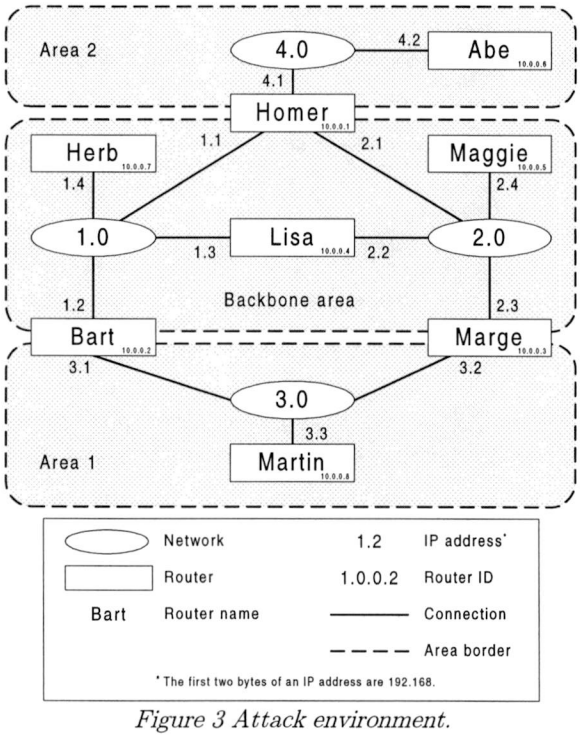

Figure 3 Attack environment.

All the routers in Figure 3 are installed with FreeBSD 2.2.8 [12]. FreeBSD is an advanced UNIX-type operating system for 'PC-compatible' computers. FreeBSD offers advanced networking, performance, security and compatibility features, which are still missing in other operating systems, even some of the best commercial ones. This was one of the reasons for choosing FreeBSD. FreeBSD has one particularly interesting feature in its packet filtering firewall (`ipfw`): the kernel packet diversion mechanism (`divert socket`). By reading from and writing to a divert socket, OSPF packets can be intercepted at the application level. The basic operation of this feature is depicted in Figure 4.

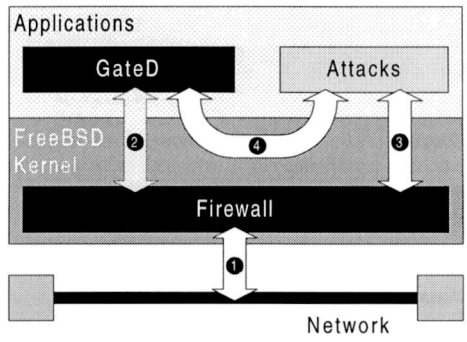

Figure 4 Diversion feature of the FreeBSD firewall.

The firewall collects all packets from the network for further investigation (1) to control access to or from a protected network. In most firewalls when a packet has passed the policy rules it is passed on to the appropriate deamon at the application level (2). In this project the routing deamon GateD (version 3.5.7) from by the Merit GateDeamon Project Group [13] was used. One of the possible policy rules in FreeBSD's firewall code is to divert a socket (3). Divert sockets are similar to raw IP sockets, except that they can be bound to a specific divert port. The IP address is ignored and only the port number is significant. A divert socket bound to a divert port will receive all packets diverted to that port. Packets may also be written to a divert port, in which case they re-enter kernel IP packet processing (4).

3.2. Insider attacks

The attacks, which are implemented and described below, are insider attacks. Here, *insider* refers to a trusted entity participating in the routing information exchange process or an outsider with the capability to intercept and modify the information exchange channels. The seven attacks that are described below were launched in the environment described in section 3.1 and were implemented in C/C++ on the FreeBSD platform. The last three attacks (Sequence Number Plus, Maximum Sequence Number and Maximum LS Age) are borrowed from the Jinao project [7][8].

3.2.1. Missing Neighbor Attack
When two routers A and B have established a bi-directional communication using the Hello protocol they expect to see themselves (their router ID) in the neighbors' Hello packet. Now lets assume a machine X sends out a Hello packet. Note that machine X does not have to be a router. However, machine X poses as router A (by spoofing the IP address and router ID of router A). Additionally in the neighbor list of the Hello packet machine X leaves out the router ID of router B. When router B receives this Hello packet it will think that router A is the originator of this packet, since the IP address and router ID indicate router A as the originator. Router B also finds that it's router ID is not listed in the neighbor list. Router B will therefore assume the bi-directional communication with router A is not longer valid and will drop router A from its database and shall try to setup a new connection with router A. As a result of this no packets will be routed over the link from A to B. The attack will work as long as machine X will continue to send the forged Hello packets. If the launches of the attack are timed correctly (within `HelloInterval`) and machine X is persistent, there will never be a new full connection between router A and router B. For this attack we need to create the IP header, the OSPF header and the forged Hello packet.

3.2.2. ACK Flood Attack
This attack is a straightforward packet flooding attack where a machine X tries to saturate router A by sending a huge amount of packets. For this attack to be effective, machine X (which does not have to be another router) has to be a faster machine than router A, in terms of packet processing power. Another condition is that the flooding should be point-to-point, so a Hello flooding attack is not possible because the Hello packets are broadcasted to one of OSPF's two multicast addresses and the attacker would flood himself. The Acknowledge packet is a packet that is sent directly between two routers and contains only one field (LSA headers). In this attack machine X continuously sends a huge amount of Acknowledge packets to router A. The content of the LSA headers in the Acknowledge packet is completely arbitrary. When router A gets flooded with the packets from machine X it has no longer time to process other packets that it receives. After a `RouterDeadInterval`, which is 10 seconds by default, the neighboring routers of router A will assume Router A is dead and will drop Router A from their databases. Although it is not necessary for the effectiveness of the attack, machine X covers its origin by spoofing the IP address and router ID of router B (an arbitrary router in the network). The attack will last as long as machine X continues to send the Acknowledge packets. For this attack the IP header, the OSPF header and the forged Acknowledge packet have to be created. Using this attack, the attacker can take out a router from the routing process, which will result in redirected packets taking non-optimal routing paths.

3.2.3. Sequence Number Mismatch Attack
This attack takes place when two routers are at the end of the Database Exchange process and have established a full bi-directional communication. Both routers have sent and received an entire sequence of Database Description packets. According to the OSPF specifications, in this state both routers do not accept any new Database Description packets. Any Database Description packet that is sent and is not already received (duplicate) or does not match the neighbors Options field will result in a `SequenceNumberMismatch` event. This means that the Database Exchange is not completed and the routers will have to start over again. For the attack to work properly, it has to fulfill several conditions. First both routers have to have established a full-state connection. Secondly the I-bit, M-bit and MS-bit in the Options Field should be set to 0. The sequence number is set to `MaxSequenceNumber` so this will always result in a `SequenceNumberMismatch` event. So what happens is that a malicious machine X sends the forged Database Description packet posing as router B (by spoofing its IP address and router ID) to router A. Where router A is the victim router. Router A will drop router B from its database because of the `SequenceNumberMismatch`. This attack will last as long machine X will continue to send the forged Database Description packets so that routers A and B cannot complete their Database Exchange process. Using this attack, an insider can manipulate the routing path and redirect packets to pass through a specific route.

3.2.4. Bad Link State Request Attack
After exchanging Database Description packets with a neighboring router, a router may find that parts of its link-state database are out-of-date. Then the Link State Request packet is used to request the pieces of the neighbor's database that are more up-to-date. When a router sends a Link State Request packet it knows exactly which instance of the database it is requesting. Each instance is defined by its LS Sequence Number, LS checksum and LS age. When a router receives a Link State Request and the router cannot find the requested LSA in its database, the router assumes that something has gone wrong in the Database Exchange process and will drop the bi-directional communication with the requesting router. So in this attack a malicious machine X poses as router B (by spoofing routers B IP address and router ID). Then machine X sends out a Link State Request packet with an arbitrary LSA (it has to be

unknown to router A) to router A. Router A will receive the Link State Request (assuming the originator was router B) with an unknown requested LSA. As a result of the unknown LSA router A will drop the connection with router B. Routers A and B will now try to setup a new bi-directional connection but as long as machine X keeps sending the forged LSR packets the connection can not be re-established. For this attack the IP header, the OSPF header and the forged LSR packet have to be created. Using this attack, we can manipulate the routing path and redirect packets to pass through a specific route.

3.2.5. Sequence Number Plus Attack

When a router receives an LSA, it can modify the LSA content and increase the LSA sequence number by 1 (SequenceNumber++). This attacking LSA, because it has a bigger LSA Sequence Number, will be considered 'fresher' by other routers. If the receiving router also changes the metric for the link in the LSA to a very high value, all receiving routers will consider the link unusable. Eventually the LSA will also be propagated to the legal originator. The originator, according to the OSPF specification, will 'fight-back' with a new LSA carrying correct link metric information and an even fresher Sequence Number. The receiving router will keep generating the SequenceNumber++ LSA each time it receives the even fresher LSA from the originator. Using this attack strategy, we have to wait 15 minutes on average to launch an attack and then keep attacking. Too speed up the attack, a fake Hello packet is sent out. All neighboring routers will respond within 10 seconds. When this incoming Hello packet is received, the attacker can modify the whole OSPF packet and make it an OSPF LSU packet. Then the attacker puts the attack LSA in the modified LSU packet and gives this LSA, whatever it is, the lowest possible sequence number (`0x80000001`). The receiving router will reply with an LSA with the current sequence number and link status information. The attacker can then modify this outgoing LSA content and increase the LSA sequence number by 1. To implement this strategy, an entire forged OSPF packet with both the OSPF checksum and LSA checksum need to be re-computed. Additionally, the IP packet length in the IP header and the IP header checksum must be re-computed. The effect of this attack is an unstable network topology if the attacker keeps generating 'Seq++' LSAs. For example, all the routers at one point will think the link status is X, but then the fight-back LSA from the originator will tell them the link status is Y. The fighting back and forth will be seen in the entire area. Using this attack, an attacker can manipulate the routing path and redirect packets to pass through a specific route.

3.2.6. Maximum Sequence Number Attack

The wrapping around of the sequence number after it reaches the maximum sequence number `0x7FFFFFFF` should be handled differently from other sequence number updates according to the OSPF specifications. In order to wrap around (using two complement) from `0x7FFFFFFF` to `0x80000001` (`0x80000000` is reserved and not used), the originator of the LSA has to purge it by sending out a `MaxAge` (3600 seconds) LSA. After this `MaxSequenceNumber` LSA is purged, the originator then sends out the LSA with sequence number `0x80000001` and the normal sequence number update follows afterward. If the OSPF is implemented correctly, the Maximum Sequence Number attack is similar to the Sequence Number Plus attack. It actually belongs to one of the persistent attacks. However, since most of the routing protocol implementations do not purge the `MaxSequenceNumber` LSA before sending out a new LSA with the sequence number `0x80000001`, Maximum Sequence Number attack becomes one of the hit-and-run attacks.

When a LSA is received, the LSA content is modified, by setting the LSA's sequence number to `0x7FFFFFFF`. The LSA, OSPF and IP header checksums need to be re-computed before we inject the tampered LSA into the system. Using this attack strategy, the attacker has to wait 15 minutes on average to launch an attack.

To speed up the attack, a fake Hello packet is sent out. This quicker strategy will be effective within 10 seconds. When a responding Hello packet is received, the whole OSPF packet is modified and made into an OSPF LSU packet. Then the attack LSA is put in the modified LSU packet. This LSA, whatever it is, is assigned the maximum sequence number `0x7FFFFFFF`. This LSA will stay in every router's topology database except the legal originator's. Depending on the implementation of the OSPF protocol, the attacker can put what link status information he wants in each router's topology database and the attacking LSA will stay in the database for one hour before it reaches its maximum age. The fighting back and forth will be seen in the network where the victim locates. Using this attack, the attacker may manipulate some routing paths and redirect the packets to pass through a specific router.

3.2.7. Maximum Link State Age Attack

When a LSA is received, the LSA content is modified. The LS Age field in the LSA header is set to its maximum value of 3600 seconds (`MaxLSAge`). The attacker also needs to re-compute both the LSA and OSPF checksums before the tampered LSA is re-injected into the system. This attacking LSA, because it has MaxAge, will cause all routers to

purge the corresponding LSA from their topology database. Eventually, it will be propagated to the legal originator of this particular LSA. The originator, according to the OSPF specification, will 'fight-back' with a new LSA carrying correct link status information and a fresher sequence number. The attacker will keep generating the `MaxAge` LSA every time he receives a fresher LSA. Using this attack strategy, we have to wait 15 minutes on average to launch the attack. Too speed it up, we send out a fake Hello packet. This quicker strategy will be effective within 10 seconds. When a responding Hello packet is received, the attacker modifies the whole OSPF packet and makes it an OSPF LSU packet. Then the attack LSA is put in the modified LSU packet. The sequence number of the LSA is set to the minimum sequence number (`0x80000001`). When the originating router receives this LSA it will respond with an LSA with the current sequence number and link status information. Then the content of this LSA is modified by setting the LS Age field to 3600 (`MaxLSAge`). To implement this strategy, the attacker needs to create the fake OSPF packet with both the OSPF checksum and LSA checksum re-computed and must also modify the IP packet length in the IP header and re-compute the IP header checksum. Using this attack, the attacker can manipulate the routing path and re-direct all the packets to pass through a specific router.

3.2.8. Attack environment conditions

In the above described attack descriptions (except for the ACK Flood attack) the network status regarding which router is the designated router (DR) and which one is the backup designated router (BDR) will affect the attack result. Actually, the adjacencies between the attacker and the victim are important to the success of this type of attack. There are always adjacencies between DR (BDR) and other routers in the network. If there are no adjacencies between the attacker and the target, the fighting will not even start. Another factor, the `MinLSArrival` constant, affects the outcome of the attack sometimes. If the attacking LSA is received within the `MinLSArrival` time period, the attacking LSA will be discarded. The fighting will not continue.

4. Intrusion detection development

To defend routers against insider attacks the intrusion detection system [14] approach was chosen. This approach is most likely to be more attractive to the industry compared to traditional approaches since the intrusion detection system approach doesn't require any modifications to routers nor protocols.

4.1. Statistical anomaly detection

An intrusion detection method that has been reported in various projects in the literature is the statistical method. The *NIDES* project at SRI [9], *Wisdom and Sense* at Los Alamos National Laboratory [15], and the *Haystack* project [16] at Haystack Laboratories are examples of the statistical methods that can be found. Among these examples, the NIDES project at SRI is most extensive in its scope and development. It also has the most complete publicly available documentation available. With the understanding of statistical analysis's general applicability, NIDES's statistical algorithm was taken as a starting point and modified and expanded it where necessary.

The basic idea in statistical anomaly detection is to compare a subject's short-term behavior with the subject's historical or long-term behavior. A subject is context-dependent, which can be a user of a computer system, a credit card holder, or one of the neighbor routers in the case of this project. Whenever short-term behavior deviates significantly from long-term behavior, a warning flag is raised to indicate a potential intrusion. In general, short-term behavior is somewhat different from long-term behavior, because short-term behavior is more concentrated on specific activities and long-term behavior is distributed across many activities. To accommodate this expected deviation between short-term and long-term behavior, the statistical component should account for the amount of deviation that it has seen in the past between a subject's short-term behaviors and long-term behaviors. The statistical component issues a warning only if the current short-term behavior is very unlike long-term behavior relative to the amount of deviation between these types of behaviors that it has seen in the past.

4.1.1. Components of the statistical approach

In this section, the components of the statistical approach are introduced. This includes various measures, half-life in updating both short-term and long-term probability distributions, scoring statistics, and the computational algorithm in obtaining these statistics.

Measures: Aspects of subject behavior are represented as measures (e.g., packet arrival frequencies in terms of their types or sources). For each measure, a probability distribution of short-term and long-term behaviors is created. For example, for the packet types received, the long-term probability distribution would consist of the historical probabilities with which different types of packets have been received, and the short-term probability distribution would consist of the recent

probabilities with which different types packets have been received. In this case, the categories to which probabilities are attached are the names of packet types, which are learned by the system as they are received. The measures can be classified into three categories:

- *Intensity* measures
- *Counting* measures
- *Record distribution* measures

These three types of measures serve different dimensional purposes. The *intensity* measures determine whether the volume of general activity generated in the recent past is normal. These measures can detect bursts of activity or prolonged activity that is abnormal, primarily based on the volume of generated data. The *counting* measures are measures for which the outcomes are counts. For example, counting measures might include CPU time or packet length. The behavior over the recent measures is compared to a historical profile of behavior to determine if the recent behavior is normal. The *record distribution* measure determines whether, for recently observed activity (say, the last few hundred records received), the types of actions being generated across neighbors are normal. For example, one might find that the last 200 routing packets received contained 120 of Hello packets, 15 of Database Description packets, 10 of Link State Request packets, 35 of Link State Update packets, and 20 of Acknowledgement packets. This data is compared to a profile of previous activity (generated over a long time) to determine whether or not the distribution of activity types generated in the recent past (the last few hundred records) is unusual.

Half-life: The specification of a half-life determines the number of packets or days of network activity that constitute short-term and long-term behavior. If the half-life for a long-term probability distribution is set to 30 profile updates, packets that were gathered 30 updates in the past contribute half as much weight toward the probability distribution as do the most recent packets. Packets that were gathered 60 updates in the past contribute one-quarter as much weight, and so forth. Thus, the most recent activity contributes more than the more distant activity, and eventually the long-term profile 'forgets' about very distant behavior. For the long-term profile, a long-term aging factor is applied to the historical data at each update, and then the new information is folded in. For the short-term profile, a short-term aging factor is applied to the profile with each packet and the current packet information is folded in.

The S statistic: For each activity caused by a subject, the statistical module generates a single test statistic value, denoted S, which summarizes the degree of abnormality in the subject's behavior in the near past.

The Q statistic: The degree of difference between a long-term profile and short-term profile for a measure is quantified using a χ^2-like statistic, comparing observation (the short-term profile) to expectation (the long-term profile). The resultant numerical value is called Q in NIDES. Each S measure is derived from a corresponding Q statistic. In fact, each S measure is a 'normalizing' transformation of the Q statistic so that the degree of abnormality for different types of measures can be added on a comparable basis. Two different methods for transforming the Q statistics into S values are used. One method is used for computing the values for S corresponding to intensity measures; a second method is used for computing the values of S corresponding to the other two measures. The computation of Q statistics and the transformations from Q to S statistics will be explained in the following sections.

4.1.2. Computing the Q Statistic for the Intensity Measures

When a neighbor router is first brought up and audited, that router has no history. Consequently, some convenient value to begin the Q statistic history must be chosen. For instance, we might initially let each Q measure be zero or some value close to the mean value for other routers.

Each Q statistic for intensities is updated each time a new packet is received. Let Q_n be the value for Q after the n^{th} packet, and Q_{n+1} be the value for Q after the $(n+1)^{th}$ packet. The formula for updating Q is:

$$Q_{n+1} = 1 + 2^{-r(t_{n+1}-t_n)}Q_n \qquad (1)$$

where the variable t_n represents the timestamp of the n^{th} packet. The decay rate r determines the half-life of the measure Q. Large values of r imply that the value of Q will be primarily influenced by the most recent packets. Small values of the decay rate r imply that Q will be more heavily influenced by packets in the more distant past. For example, a half-life of 10 minutes corresponds to an r value of $r = -[\log_2(0.5)]/10 = 0.1$.

Q is the sum of packet activity over the entire past activities, exponentially weighted so that the more current activity has a greater impact on the sum. Q is more a statistic of near past behavior than of distant past behavior. One important property of this Q statistic is that it's not necessary to keep extensive information about the past to update Q.

Noted that the intensity measures use clock time as the unit by which age is calculated. This is important because the intent of this measure is to assess the extent to which bursts of activity are normal.

4.1.3. Computing the Q Statistic for the Counting and Record Distribution Measures

Suppose that there are M activity types. For each activity a long-term historical relative frequency of occurrence must be calculate, denoted f_m, for that activity type. For instance, suppose that over the last six months, 35% of all received packets are Hello packet type. Then f_m for the Hello packet type would be 0.35. The algorithm used to compute f_m on the k^{th} day is equal to:

$$f_{m,k} = \frac{1}{N_k} \sum_{j=1}^{k} W_{m,j} 2^{-b(k-j)} \tag{2}$$

where b is the decay factor (similar to r which was defined before), and $W_{m,j}$ is the number of packets received on the j^{th} day that indicate that the m^{th} packet type was received. N_k is the exponentially weighted total number of packets that have been received since the router was first monitored. The formula for N_k is:

$$N_k = \sum_{j=1}^{k} W_j 2^{-r(k-j)} \tag{3}$$

where W_j is the number of packets received on the j^{th} day.

The Q statistic compares the short-term distribution of the types of packets that have been received with the long-term distribution of the same types. In the simplest situation, Q_n (the value of the Q statistic when the n^{th} packet is received) is defined as follows:

$$Q_n = \sum_{m=1}^{M} \frac{(g_{m,n} - f_m)^2}{V_m} \tag{4}$$

where $g_{m,n}$ is the relative frequency with which the m^{th} packet type has been received in the recent past (which ends at the n^{th} packet), and V_m is the approximate variance of the $g_{m,n}$. If we view $g_{m,n}$ as the short-term profile for the packet distribution and f_m as the long-term profile for packet distribution, then Q_n is larger whenever the distribution of packet types in the recent past differs substantially from the historical distribution of packet types, where 'substantially' is measured in terms of the statistical variability introduced because the near past contains relatively small sample size. The value of $g_{m,n}$ is given by the formula:

$$g_{m,n} = \frac{1}{N_r} \sum_{j=1}^{n} I(j,m) 2^{-r(n-j)} \tag{5}$$

or by the recursion formula:

$$g_{m,n} = 2^{-r} g_{m,n-1} + \frac{I(j,m)}{N_r} \tag{6}$$

where $I(j,m) = 1$ if the j^{th} packet indicates packet type m has been received and 0 otherwise. The decay rate r for Q determines the half-life for the Q statistic.

N_r is the sample size for the Q statistic, which is given by the formula:

$$N_r = \sum_{j=1}^{n} 2^{-r(n-j)} \tag{7}$$

which rapidly approaches an asymptotic value of $1/(1-2^{-r})$. The value of V_m is given by the formula:

$$V_m = \frac{f_m(1-f_m)}{N_r} \tag{8}$$

except that V_m is not allowed to be smaller than $0.01/N_r$.

4.1.4. *Computing the Frequency Distribution for Q*

The first step in calculating the historical distribution for Q is to define bins into which Q can be classified. N_b bins will be used for a Q statistic. Let Q_{max} be the maximum value that we ever expect to see for Q. This maximum value depends on the particular types of measures being considered. The cut points for the N_b bins are defined on either a linear or geometric scale. In the original NIDES algorithm a magic value of 32 is chosen for N_b. For example, when a geometric scale is used, bin 0 extends from 0 to Q_{max}^{1/N_b}, bin 1 extends from Q_{max}^{1/N_b} to Q_{max}^{2/N_b} and bin N_b–1 extends from $Q_{max}^{(N_b-1)/N_b}$ to infinity.

Let P_m denote the relative frequency with which Q is in the m^{th} interval (bin). Each Q statistic is evaluated after each packet is received (whether or not the value of Q has changed). The formula for calculating P_m on the k^{th} day after a router was brought up is:

$$P_{m,k} = \frac{1}{N_k} \sum_{j=1}^{k} W_{m,j} 2^{-b(k-j)} \tag{9}$$

where k is the number of days that have occurred since the router was first monitored; $W_{m,j}$ is the number of packets on the j^{th} day for which Q was in the m^{th} bin; N_k is the exponentially weighted total number of packets that have been received since the router was first monitored. The formula for N_k is:

$$N_k = \sum_{j=1}^{k} W_j 2^{-b(k-j)} \tag{10}$$

where W_j is the number of packets received on the j^{th} day.

The computations for $P_{m,k}$ and N_k can be simplified by using the following recursion formulas:

$$P_{m,k} = \frac{2^{-b} P_{m,k-1} N_{k-1} + W_{m,k}}{N_k} \quad (11)$$

$$N_k = 2^{-b} N_{k-1} + W_k \quad (12)$$

4.1.5. *Deriving S from Q for the Intensity Measures*

For the intensity measures, the value of Q corresponding to the current packet represents the number of packets received in the recent past. Here, 'recent past' corresponds to the last few minutes for the Q statistic with a half-life of one minute and to the last several hours for the Q statistic with a half-life of one hour. In addition to knowing the current value for Q, the statistical module maintains a historical profile of all previous values of Q. Thus, the current value of Q can be compared to this historical profile to determine whether the current value is anomalous. The transformation of Q to S for the intensity measures requires knowledge of the historical distribution of Q. For example the following historical distribution for the intensity measures Q with a half-life of one minute might be found:

- 1% of the Q values are in the interval 0 to 10 packets
- 7% are in the interval 10 to 20 packets
- 35% are in the interval 20 to 40 packets
- 18% are in the interval 40 to 80 packets
- 28% are in the interval 80 to 160 packets
- 11% are in the interval 160 to 320 packets

The S statistic would be large value whenever the Q statistic was in the interval 0 to 10 or was larger than 320 (either because it is a relatively unusual value for Q or the value has not occurred historically). The S statistic would be close to zero whenever Q was in the interval 20 to 40, because these are relatively frequently seen values for Q. The algorithm for deriving S values from Q statistics for the intensity measures is as follows:

1. Let P_m denote the relative frequency with which Q belongs to the m^{th} interval. Using the previous example, the first interval is 0 to 10 and the corresponding P value (P_0) equals 1%. There are N_b values for P_m, with $0 \le m \le N_b$ and N_b is the number of intervals used.
2. For the m^{th} interval, let $Tprob_m$ denote the sum of P_m and all other P values that are smaller than or equal to P_m in magnitude. In our example, *Tprob* for the interval $40 \le Q \le 80$ is equal to 37% (18% + 11% + 7% + 1%).
3. For the m^{th} interval, let s_m be the value such that the probability that a normally distributed variable, with mean 0 and variance 1 is larger than s_m in absolute value, equals $Tprob_m$.

The value of s_m satisfies the equation:

$$P\left(\left|N(0,1)\right| \ge s_m\right) = Tprob_m \quad (13)$$

or

$$s_m = \Phi^{-1}\left(1 - \frac{Tprob_m}{2}\right) \quad (14)$$

where $\Phi^{-1}(x)$ is the inverse cumulative distribution function of a $N(0,1)$ variable x. For example, if $Tprob_m$ is 5% then s_m is set equal to 1.96, and if $Tprob_m$ is equal to 10% then we set s_m equal to 1.28. Suppose that after processing a packet, the Q value is in the m^{th} interval, then S is set equal to s_m.

4.1.6. *Deriving S from Q for the Counting and Record Distribution Measures*

For all measures other than the intensity measures, Q compares short-term behavior to long-term behavior. For both the intensity measures and the other measures, a long-term profile for Q using N_b intervals have to be calculated. For example, for a non-intensity measure such as types of packets used we might find a probability distribution for Q similar to the one displayed earlier for an intensity Q, except that the range of values would be expressed in terms of the degree of similarity between the short-term profile of types of packets used and the long-term profile of types of packets used, with larger numbers representing less similarity.

Because of the difference in the way that Q is defined for intensity measures and other measures, the transformation of Q to S is slightly different for non-intensity measures. For non-intensity measures, we let $Tprob_m = P_m + P_{m+1} + \ldots + P_{N_b-1}$.

In the previous example, if Q were a non-intensity measure, the *Tprob* value of the interval $40 \le Q \le 80$ would be equal to 18% + 28% + 11% = 49%. Thus, in these cases, S is a simple mapping of the percentiles of the distribution of Q onto the percentiles of a half-normal distribution. In practice, the Q tail probability calculations done only once, at update time. Each interval for Q is associated with a single s value, and when Q is in that interval, S takes the corresponding s value.

4.1.7. Training and Updating

Training is the process by which the statistical component learns normal activity for a subject. It consists of *measure training* wherein the algorithm learns the observed values for each measure and Q *training* wherein the system builds an empirical distribution for the Q statistic, which measures the measure-by-measure difference between the long- and short-term profiles. Both phases have a minimum training period before anomaly scoring begins. A

subject's long-term profile is considered trained when it has gone through the both training phases. At this point anomalies may be reported.

Number of bins: A challenge in this statistical approach is to choose an N_b, which is the total number of bins. In the original NIDES algorithm, a magic number of 32 is chosen. In this modified algorithm it is left to the user to make the decision. There are some tradeoffs when choosing N_b. If N_b is too small, the bins are too 'coarse'. In some cases, one may find most values reside in the one or two bins, which results in insensitivity to intrusions. On the other hand, if N_b is too big, the bins tend to be too 'fine', so that lots of them would be associated with very small probabilities, which will result in an unnecessary requirement for more system resources to do the bookkeeping.

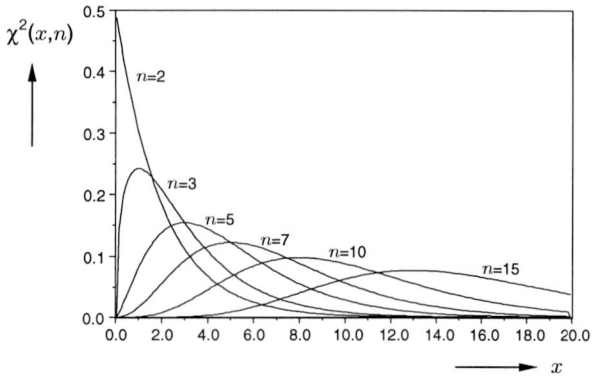

Figure 5 χ^2-distribution.

Another performance penalty will be introduced if N_b is too big. As a simple example, let's assume independence between all these bins. Then, the Q distribution will be χ^2 with N_b–1 degrees of freedom. The larger the degree of freedom is, the fatter the χ^2 curve will be as shown in Figure 5. This leaves the Q distribution very hard to track and handle.

Bin redistribution: The space that the values (either measure values or Q values) span limited by a maximum value which is f_{max} for value measures and Q_{max} for Q values. While the algorithm is learning the values it might happen that the maximum values are not set correctly and more than 1% of the values is in the last bin. The algorithm does not allow the last bin to contain more than 1% of the measures. To avoid the inconvenience of starting the learning phases all over again a redistribution mechanism is introduced. In the following example Q_{max} is used but the same formulas apply for f_{max}. For linear bins the new Q_{max} is set to:

$$Q_{max,new} = Q_{max,old} \frac{N_b}{N_b - 1} \tag{15}$$

where N_b is the total number of bins that is used. When the new Q_{max} is set, the values in the bins have to be redistributed over the new bins.

For example, we might have a measure where the value space is cut into 5 linear bins and Q_{max} is set to 20. The measures are distributed as depicted in Figure 6.

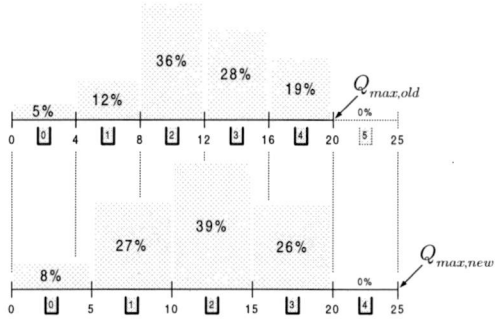

Figure 6 Redistribution of the bin values.

Before the new relative frequencies P_m can be calculated an empty temporary bin is added to the profile (bin 5). Using the previous mentioned formula, the new Q_{max}=25. The formula to calculate the new relative frequency P_m is:

$$P_{m,new} = \left(1 - \frac{m}{N_b - 1}\right) P_{m,old} + \left(\frac{m+1}{N_b - 1}\right) P_{m+1,old}$$

$$\text{with } 0 \le m \le N_b\text{-}1 \tag{16}$$

After the redistribution the last bin always contains 0% of the values. Because of the different properties of the geometric bins the formulas for profiles using geometric bins are different.

For a geometric bin the new Q_{max} value is calculated using:

$$Q_{max,new} = Q^2_{max,old} \tag{17}$$

and the redistribution (after adding an empty temporary bin) of the relative frequency P_m is calculated with:

$$P_{m,new} = \begin{cases} P_{m,old} + P_{m+1,old} & \text{for } m = 0 \\ P_{m+1,old} & \text{for } 0 < m < N_b - 1 \\ 0 & \text{for } m = N_b - 1 \end{cases} \tag{18}$$

After the redistribution the last bin contains 0% of the values.

Alarm thresholds: When the long-term behavior deviates too much from the recent behavior the S statistic will exceed a certain threshold. Since S is a mapping of Q, which has a χ^2-like distribution, defining a threshold for S is the equivalent to setting a significance level α for a χ^2 goodness-of-fit test.

Translated to the NIDES algorithm this means that:

$$Tprob_m < \alpha \quad \rightarrow \quad S < \Phi^{-1}\left(1-\frac{\alpha}{2}\right) \qquad (19)$$

So if the significance level of the test is set to α=0.01 then the S value must exceed 2.58 for the algorithm to call it an anomaly.

4.2. NFR back-end development

The intrusion detection algorithm is implemented as back-ends for Network Flight Recorder (version 2.0.3 Research) [10]. Network Flight Recorder (NFR) is a network-monitoring framework that is developed by Network Flight Recorder Inc. NFR includes a generalized and filtering language called N-Code [17], as well as the ability to trigger alerts and log packet information. NFR also provides capabilities for usage measurement as well as auditing. By implementing the algorithm as back-ends in the commonly available framework of NFR, rapid feedback from colleagues around the world as to the usefulness of the method can be received. And the platform independency of the N-code also provides the possibility to rapidly deploy the back-ends on many different types of machines.

4.2.1. Implementation

The statistical anomaly detection algorithms that are described above are implemented as back-ends for NFR. The back-end language N-code was created with packet filtering in mind. Because of this the N-code offers some nice mechanisms to retrieve bits and bytes from packets, buts lacks the flexibility and data types one would like to see in a programming language. The greatest shortage of N-code is the lack of a floating-point data type. The algorithms that are used depend heavily on a data type that can handle decimals. To overcome this problem the integer values are multiplied by a constant factor *CF*. Doing this another limitation of the N-code appears: integers in NFR are limited from -2^{31} to $+2^{31}$. So we want *CF* to be big to get a high accuracy but *CF* cannot be to big because the integer values will 'wrap around'. *CF* is empirically set to 10000 (the equivalence of 4 decimals). Another way to prevent the integers from wrapping around is to use a different order of precedence in calculations with multiplications and divisions. For example:

$$N_r\left(\frac{(f_m-g_m)^2}{V_m}\right)=\frac{\left(N_r(f_m-g_m)^2\right)}{V_m}=\frac{(f_m-g_m)^2}{\left(\frac{V_m}{N_r}\right)} \qquad (20)$$

Logically these three expressions all evaluate to the same value but in N-code, depending on the values of f_m, g_m, V_m and N_r, the expressions could cause an overflow or could evaluate to zero because the denominator is too large.

Because the N-code only uses integers, divisions will cause loss of information. Using the following theoretical property of the profiles:

$$\sum_i P_i = \sum_i f_i = \sum_i g_i = 1 \qquad (21)$$

the affected profiles can be reconditioned using:

$$P_{m,new}=\frac{P_{m,old}}{\sum_i P_{i,old}},\ f_{m,new}=\frac{f_{m,old}}{\sum_i f_{i,old}},\ g_{m,new}=\frac{g_{m,old}}{\sum_i g_{i,old}} \qquad (22)$$

The N-code also lacks mathematical functions. The statistical algorithm needs a power and logarithm function. Since x^a is the same as $e^{a\cdot\log(x)}$ the functions that we implemented are the exponential function and the logarithm function. The functions are implemented as a table-look-up algorithm with first-order interpolation. Another function that is needed by the algorithm is the inverse cumulative distribution function $\Phi^{-1}(x)$ for the normal distribution. This function is also implemented as a table-look-up algorithm with first-order interpolation.

The starting time for all profiles has a random offset (at most 30s). This is necessary to prevent a NFR back-end CPU overflow at update time of the profiles. The random function is implemented as a linear congruential generator of the form $X_n=(aX_{n-1}+b) \bmod m$. With a=1291, b=4621 and m=21870 the generator is a maximum period generator (with period m) and passes the spectral test for randomness.

For a specific system, we typically have a set of statistical measures that can be used for intrusion detection and the performance of the IDS depends on which measures are chosen. The following sections will discuss the implemented measures and the reasons why they were chosen. If one would like to use different measures, for any reason, new measures can easily be added to a back-end.

4.2.2. Intensity measures

The first measure is the intensity measure. This measure is chosen to measure data volume. The measures measure the inter-arrival time of OSPF packets. In a stable routing domain, routers exchange routing information regularly. The volume of the traffic tends to have a fixed pattern. The described attacks may invoke 'fight-back', which will cause additional routing traffic. In fact most anomalies in the OSPF routing domain tend to manifest their abnormal behaviors trough variations in data volume. Therefore this measure is useful in detecting these anomalies. The data volume measures that were chosen can be found in Table 2.

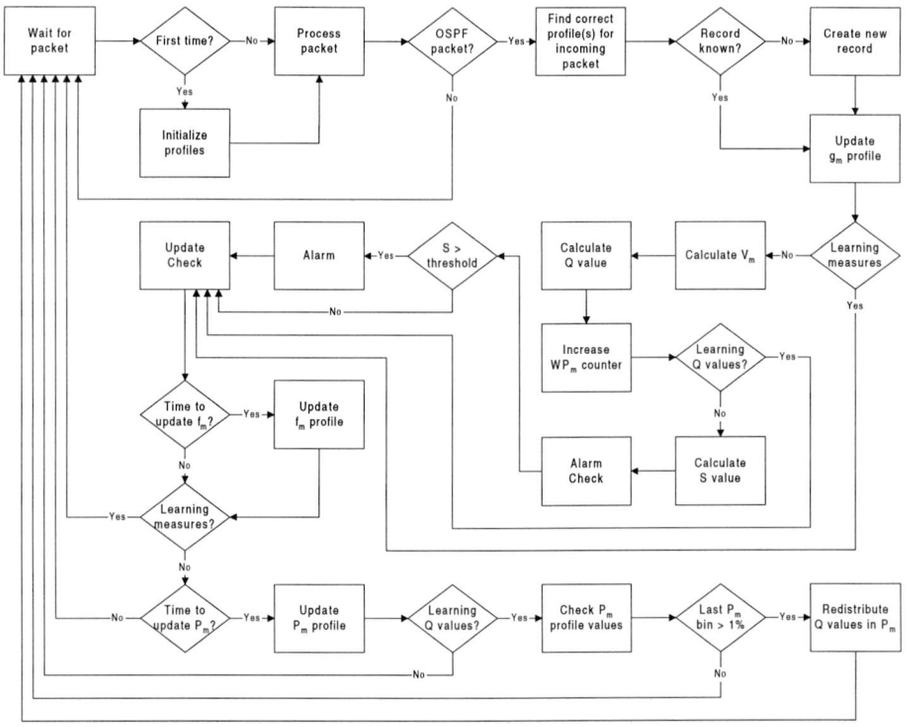

Figure 7 Intensity measures back-end flowchart.

As can been seen from the table incoming packets are measured separately from outgoing packets.

Table 2 Intensity measures.

Description of measure	IN	OUT
Data volume of all OSPF packets	✓	✓
Data volume of Hello packets	✓	✓
Data volume of Database Description packets	✓	✓
Data volume of Link State Request packets	✓	✓
Data volume of Link State Update packets	✓	✓
Data volume of Acknowledge packets	✓	✓

A flowchart of the N-code implementation is depicted in Figure 7. The back-end sniffs every packet that passes by on the network segment. If the packet is the first packet that arrives to the back-end, then the intensity profiles are initialized. During this initialization, parameters such as the type of bin (linear or geometric), the number of bins, half-life times, learning periods and initial profiles are set. The initial profiles can be empty or already learned. Now that the parameters are set and the profiles are initialized the packet and all following packets are further analyzed. If the packet is an OSPF packet the back-end will further investigate the packet. To check if the packet is an OSPF packet the protocol field in the IP header is probed (OSPF has protocol number 89). If the packet is an OSPF packet, the type field in the OSPF header is examined to determine the type of the OSPF packet. The source IP address of the packet is used to determine the direction of the packet (incoming or outgoing). From the type field in the OSPF header the type of the OSPF packet can be determined. Now, based on direction and type, the appropriated profiles are updated.

The Q value corresponding to a profile is updated as described in section 4.1.2. The only difference in the implementation is that a multiplication factor 10 is introduced to prevent against integer wrapping. Now that the new Q value is known, the long-term counter W in the accompanying bin has to be increased. When the algorithm is not learning the Q values, the S value for the profile is calculated, as described in section 4.1.5. If the S value exceeds a threshold an alarm is raised. Now the algorithm measures how long the profiles have been learning. When the learning time has exceeded the configured learning period a flag is raised and from this point on the algorithm will be able to generate alarms. If the algorithm is still learning a check is done to see if it is necessary to update P_m, the frequency distribution for Q. If P_m needs to be updated and the algorithm is still learning the Q values then the last bin is inspected to see if it contains more than 1% of the Q values. Whenever this is the case, P_m is redistributed and a new Q_{max} value is calculated, as described in section 4.1.7.

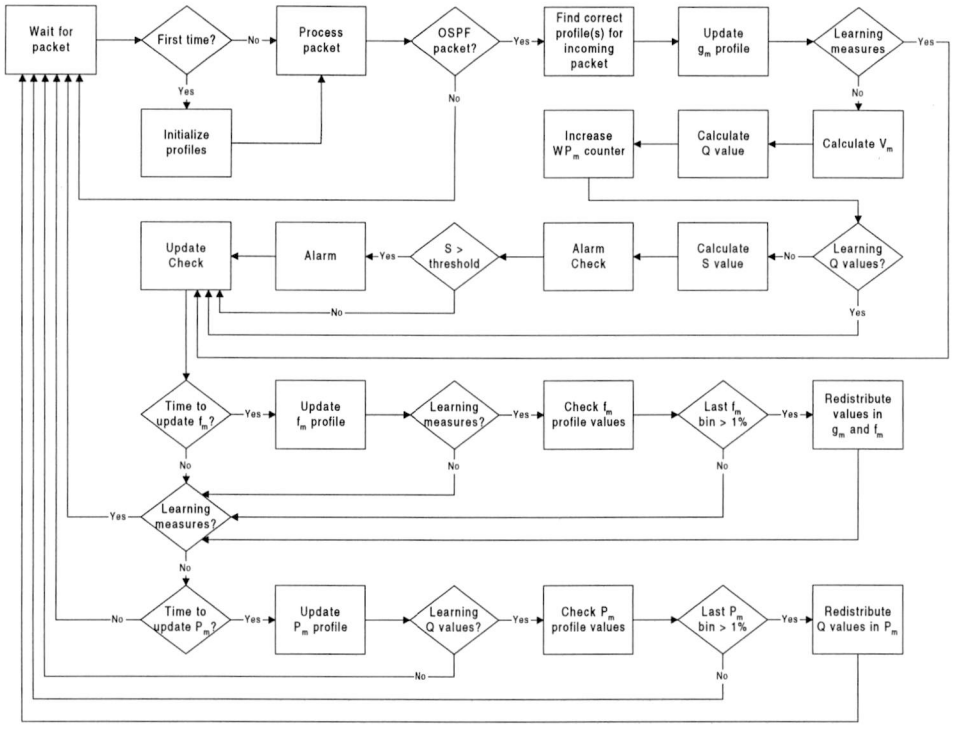

Figure 8 Counting measures back-end flowchart.

4.2.3. Counting Measures

The second measure is the counting measure. Counting measures are measures for which the outcomes are counts. These measures assess the extent to which the behavior is normal in the recent past. The types of counting measures that were chosen can be found in Table 3. Again incoming packets are processed separately from outgoing packets.

Table 3 Counting measures.

Description of measure	IN	OUT
OSPF packet length	✓	✓
OSPF packet frequency	✓	✓
LS Age	✓	✓

In a stable routing domain the routing information is exchanged at fixed intervals. The frequency of the traffic tends to have a fixed pattern. Another property of a stable network is that the length of OSPF packets for each packet type will tend to be the same which also goes for the Link-State Ages. In case of an attack it is likely that the packet lengths, and the packet frequency will be anomalous. When an attack is launched on the LS Age field the LS Age measure will be very useful in detecting the resulting anomalies.

The statistical algorithm for the counting measures, as described in section 4.1.3 and 4.1.6, is implemented in N-code as a back-end for NFR. A flowchart of the implementation is depicted in Figure 8. The back-end sniffs every packet that comes along. If the packet is the first packet that arrives to the back-end then the counting profiles are initialized. During this initialization, parameters such as the type of bin (linear or geometric), the number of bins, half-life times, learning periods and initial profiles are set. The initial profiles can be empty or already learned. Now that the parameters are set and the profiles are initialized the packet is further analyzed. When the packet is an OSPF packet the back-end will further investigate the packet.

To check if the packet is an OSPF packet the protocol field in the IP header is probed (OSPF has protocol number 89). If the packet is indeed an OSPF packet, the type field in the OSPF header is examined to determine the type of the OSPF packet. The source IP address of the packet is used to determine the direction of the packet (incoming or outgoing). Now, based on direction and type, the appropriated profiles are updated.

First the short-term profile g_m is updated as described in section 4.1.3. When the algorithm is not learning the measure values the variance V_m is calculated and the long term profile f_m is compared to the just updated short term profile g_m. This will result in a Q value. Now that the new Q value is known, the long-term counter W in the

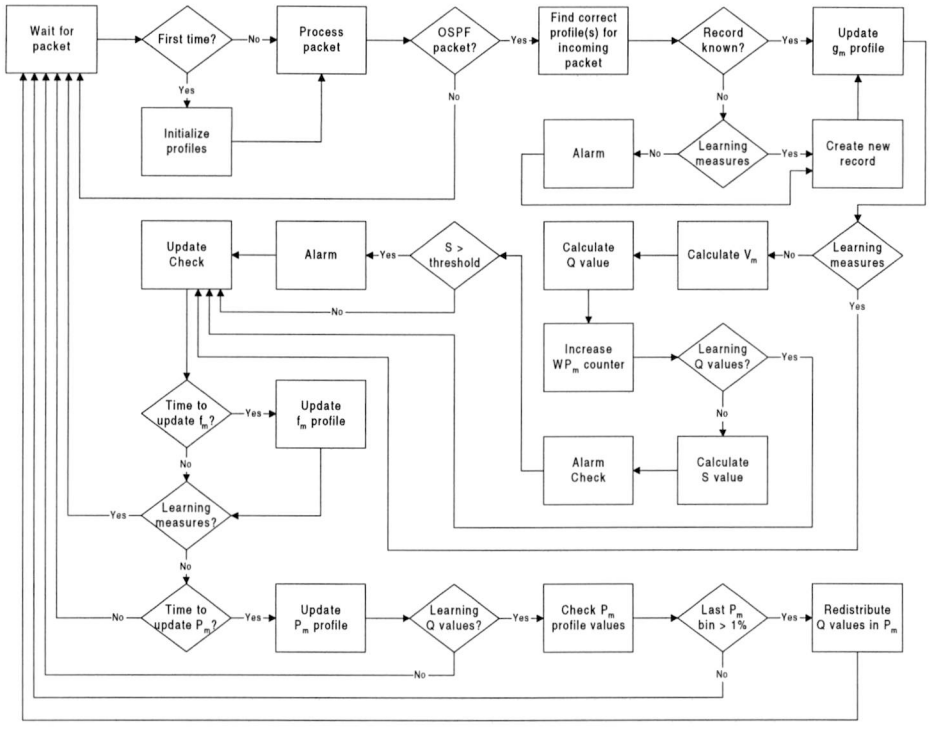

Figure 9 Record distribution measures back-end flowchart.

accompanying bin has to be increased. If the algorithm is not learning the Q values, the S value for the profile is calculated, as described in section 4.1.6. In the case that the S value exceeds a threshold an alarm is raised. Now the algorithm measures how long the profiles have been learning. When the learning time has exceeded the configured learning period a flag is raised and from this point on the algorithm will be able to generate alarms. In the case that the algorithm is still learning then a check is done if it is necessary to update the long-term profile f_m or the frequency distribution for Q, P_m. When f_m or P_m needs to be updated and the algorithm is still learning the associated values then a check is done to see if the last bin contains more than 1% of the values. If this is the case then the bins are is redistributed and a new maximum bin value (Q_{max} or f_{max}) is calculated. If f_m needs to be redistributed g_m will also be redistributed.

4.2.4. Record Distribution Measures

The last type of measure is the record distribution measure. These measures assess the extent to which the distribution of records is normal in the recent past.

Table 4 Record distribution measures.

Description of measure	IN	OUT
OSPF packet type	✓	✓
IP source address	✓	✓
IP destination address	✓	✓

The types of record distribution measures that were chosen can be found in Table 4. Again incoming packets are processed separately from outgoing packets. The OSPF protocol has five different types of packets: Hello, Database Description, Link State Request, Link State Update and Acknowledge. In a routing domain, each OSPF router will send Hello packets to its adjacent neighbors at fixed intervals. Type 2 (DD) and 3 (LSR) packets are used only when a new adjacency is initialized. Type 4 (LSU) packets are sent out by each router at another fixed interval to maintain freshness of its link-state database. Each updated LSA has to be acknowledged (type 5).

This acknowledgment does not have to follow the update directly. A router may choose to delay the acknowledgment to aggregate the acknowledgments of a few LSAs, to save some bandwidth. Under normal conditions the distribution of the packet types is very stable. Another distribution that the back-end keeps track of are the IP addresses. In a stable routing domain the IP addresses that a router receives and sends packets to belong to an unchanging set of IP addresses. Also the amounts of traffic from these IP addresses have a stable distribution under normal conditions. For example all Hello packets normally sent to the multicast address `224.0.0.5`. So these measures are chosen to detect anomalies in these distributions.

The statistical algorithm for the record distribution measures, as described in section 4.1.3 and 4.1.6, is implemented in N-code as a back-end for NFR. A flowchart of the N-code implementation is depicted in Figure 9.

The back-end also sniffs every packet that comes along. If the packet is the first packet that arrives to the back-end then the record distribution profiles are initialized. During this initialization, parameters such as the type of bin (linear or geometric), the number of bins, half-life times, learning periods and initial profiles are set. The initial profiles can be empty or already learned. If the profile is empty the back-end will learn the records during run-time. Now that the parameters are set and the profiles are initialized the packet is further analyzed. If the packet is an OSPF packet the back-end will further investigate the packet. Again the IP header is probed to determine if the packet is an OSPF packets. If it is an OSPF packet, the type field in the OSPF header is examined to determine the type of the OSPF packet and the source IP address of the packet is used to determine the direction of the packet (incoming or outgoing). Now, based on direction and type, the appropriated profiles are updated.

The first thing is to check if the incoming packet type and IP addresses are already known to the back-end. If the back-end is still learning the different records then a new record is created. If the back-end is no longer learning measures and a new type is received then an alarm is raised.

Now, the short-term profile, g_m, is updated. If the algorithm is not learning the measure values then the variance V_m is calculated and the long term profile f_m is compared to the just updated short term profile g_m, as described in section 4.1.3. This will result in a Q value. Now that the new Q value is known, the long-term counter W in the accompanying bin has to be increased. If the algorithm is not learning the Q values, the S value for the profile is calculated, as described in section 4.1.6. When the S value exceeds a threshold an alarm is raised. Now the algorithm measures how long the profiles have been learning. When the learning time has exceeded the configured learning period a flag is raised and from this point on the algorithm will be able to generate alarms. In the case that the algorithm is still learning then a check is done if it is necessary to update the long-term profile f_m or the frequency distribution for Q, P_m. In the case that P_m needs to be updated and the algorithm is still learning the associated values then a check is done to see if the last bin contains more than 1% of the values. If this is the case then the bins are is redistributed and a new maximum bin value (Q_{max}) is calculated. Because of the properties of record distribution measures this is not necessary for the g_m or f_m profiles.

5. Results

This section covers the experimental results of the statistical anomaly detection algorithm in detecting the attacks on the OSPF protocol.

5.1. Attack Environment and Configuration

For the attacks the network topology as depicted in Figure 3 was used. The intrusion detection algorithm ran on Herb (192.168.1.4). The attacks were launched from Lisa (192.168.1.3).

The statistical algorithm has a lot of parameters that have to be determined. Table 5 gives an overview of these parameters for the three back-ends and their profiles.

Determining the number of bins is a delicate problem. Because of the characteristics of the intensity measures their sample space is cut into 32

Table 5 Profile parameters.

Profile	Type of measure	Type of bin	Number of bins	Update period (mins)	Short half-life (packets)	Long half-life (sec)	Q half-life (sec)	Measure learning (min)	Q learning (min)
OSPF packets	intensity	GEOMETRIC	32	60	-	-	1800	1440	1440
Hello packets	intensity	GEOMETRIC	32	60	-	-	1800	1440	1440
DD Packets	intensity	GEOMETRIC	32	60	-	-	1800	1440	1440
LSR packets	intensity	GEOMETRIC	32	60	-	-	1800	1440	1440
LSU packets	intensity	GEOMETRIC	32	60	-	-	1800	1440	1440
ACK packets	intensity	GEOMETRIC	32	60	-	-	1800	1440	1440
Packet length	counting	LINEAR	20	60	1000	1800	1800	1440	1440
Packet frequency	counting	LINEAR	20	60	1000	1800	1800	1440	1440
LS Age	counting	LINEAR	20	60	1000	1800	1800	1440	1440
Packet type	record	LINEAR	20	60	1000	1800	1800	1440	1440
Source IP	record	LINEAR	20	60	1000	1800	1800	1440	1440
Destination IP	record	LINEAR	20	60	1000	1800	1800	1440	1440

bins on a geometric scale. The sample spaces of the counting and record distribution measures are cut into 20 bins on a linear scale.

Another problem is setting the half-life parameters. The short-term half-life is set to 1000 packets. This means that a packet that was gathered 1000 packet updates in the past contributes half as much weight toward the probability distribution as do the most recent packets. The window of 1000 packets must at least as large as the OSPF cycle, which is normally the `LS Refresh Time` of 1800 seconds.

An average OSPF packet arrival frequency in the test network of 0.25 Hz justifies the 1000 packet half-life. The long-term and the Q half-life parameters are set to half an hour, 1800s, accordingly.

Making the half-life larger will make the profile less sensitive to intrusions and on the other side making the half-life smaller will cause a higher false positive rate (discussed later) because the profile is too sensitive.

The learning periods are set to 24 hours (1440 min) for all measures. The two learning phases, measure training and Q value training, are sequentially, thus making the total training period per profile two days.

A back-end will raise an alarm if the S value exceeds a threshold. There are two levels of alarms: a yellow and a red alarm. The yellow alarm is raised if the tail probability of the Q distribution exceeds 10%. This means $\alpha > 10\%$ which results in a yellow alarm threshold of $S_{yellow} = 1.65$. The red alarm is raised if the tail probability of the Q distribution exceeds 1%. In the same way, this means $\alpha > 1\%$ which results in red alarm threshold of $S_{red} = 2.58$.

5.2. False Positives

Because of the statistical nature of the adapted algorithm it is probable that there will be false positives.

When a value is close to a bin boundary a small change can cause the value to fall into the adjacent bin (possibly causing an alarm). The false positive rate is defined as:

$$r_{false\ positives} = \frac{N_{alarm}}{N_{total}} \tag{23}$$

Where N_{alarm} is the number of observed alarms and N_{total} is the total number of observations in the same period of time. The false positive rates for the used measures are depicted in Table 6.

The cause of the false alarms is that if a Q value is close to a bin boundary, a small change in the Q value will make it jump just to the other side of the boundary. If the bin on the other side of the boundary is rare then the jump causes a false positive.

But not all intrusions will produce an identifiable anomaly. The term identifiable has two meanings: either the intrusion behavior pattern does not deviate too much from normal behavior pattern (from a measure's point of view), or the threshold of a measure is too conservative (in order to keep low false positive rate, for example) to catch the anomaly. Also, different measures have different sensibilities for different kinds of attacks. Therefore the attacks durations alternate between 20 and 100 (times or seconds). To ensure the validity of the results all attacks were repeated three times.

Table 6 False positive rate.

Measure	Incoming		Outgoing	
	yellow	red	yellow	red
Intensity measures				
Total number of OSPF packets	0.34 %	0.07 %	0 %	0 %
Total number of Hello packets	0 %	0 %	0 %	0 %
Total number of DD packets	0 %	0 %	0 %	0 %
Total number of LSR packets	0 %	0 %	0 %	0 %
Total number of LSU packets	0 %	0 %	0 %	0 %
Total number of ACK packets	0.01 %	0 %	0 %	0 %
Counting measures				
OSPF packet length	0 %	0 %	0 %	0 %
OSPF packet frequency	0 %	0 %	0 %	0 %
Link-State Age	0.58 %	1.01 %	0 %	0 %
Record distribution measures				
OSPF packet type distribution	0 %	0 %	0 %	0 %
SRC IP address	0 %	0 %	0 %	0 %
DST IP address	0 %	0 %	0 %	0 %

5.3. Results for a Router Restart

As described above, the statistical anomaly detection algorithm will raise an alarm whenever the recent behavior deviates too much from the long-term behavior. The raised alarms may not always indicate an attack but just an anomaly. A router restart or a dead link are examples of these anomalies. The algorithm should be able to detect these anomalies. Table 7 gives an overview of the alarms that were raised during all attacks and a router restart. The first row indicates the alarms for the router that ran the IDS algorithm (Herb: 192.168.1.4) and the second row shows the alarms for a 'normal' router (Homer: 192.168.1.1). When a router goes down the routers will update each other about it and will look for alternative routes, causing heavy routing traffic. When the restarted router comes up again it will try to establish adjacencies with its neighbors, again

Table 7 Detection results.

R Red alarm
→ Incoming packets
← Outgoing packets

		Intensity												Counting						Record distribution					
		Total number of all OSPF packets		Total number of Hello packets		Total number of DD packets		Total number of LSR packets		Total number of LSU packets		Total number of ACK packets		OSPF packet length		OSPF packet frequency		Link State Age		OSPF packet type distribution		Source IP address		Destination IP address	
Attack	**Duration**	→	←	→	←	→	←	→	←	→	←	→	←	→	←	→	←	→	←	→	←	→	←	→	←
Router Restart	IP=1.4	R		R	R	R	R		R	R	R	R	R							R†	R†			R†	R†
	IP=1.1	R		R	R	R		R		R		R	R	R						R†				R†	R
Missing Neighbor	N=20	R		R		R		R				R	R							R†				R†	
	N=100	R		R	R	R		R		R		R	R	R						R†				R†	
Acknowledge Flood	t=20s	R										R		R						R				R†	
	t=100s	R										R		R						R		R		R†	
BAD Link State Request	N=20	R			R	R		R					R							R†				R†	
	N=100	R		R	R	R		R		R		R	R	R						R†				R†	
Sequence Number Mismatch	N=20	R		R		R		R				R	R							R†				R†	
	N=100	R		R	R	R		R		R		R	R	R						R†				R†	
Sequence Number Plus	N=20	R								R		R	R	R						R				R†	
	N=100	R	R							R	R	R	R	R	R					R	R			R†	
Maximum Sequence Number	t=20s									R	R													R†	R†
	t=100s	R								R	R							R	R	R				R†	R†
Maximum LS Age	N=20	R		R						R		R	R	R				R		R				R†	
	N=100	R	R	R						R		R	R	R	R			R		R	R			R†	

causing heavy routing traffic. This anomalous traffic is detected by the intensity measures. The record distribution measures detect that packets were send that are atypical in normal operation of the protocol (Database Description packets and Link-State Request packets). The record distribution measures also detect that packets are send to IP addresses that are not used in normal circumstances, when a router tries to establish adjacencies with it's neighbors the packets are send directly from host to host.

5.4. Results for the Missing Neighbor Attack

In the missing neighbor attack a router will drop the connection with router that does not list its router ID in the neighbor list. The alarms that were raised when the missing neighbor attack was launched can be found in Table 7. The 'dropped' router will try to re-establish the connection with its neighbors. This causes the intensity measures to raise alarms. During the trial to re-establish the connection Database Description and Link State Request packets are sent, which will cause anomalies in the distribution of the packets. Therefore the record distribution measures will also raise alarms in both the packet type and the destination IP measures. The results for both 20 and 100 attacks launches are listed.

5.5. Results for the ACK Flood Attack

In the ACK flood attack a router is flooded with Acknowledge packets. The results of this attack are listed in Table 7. The first row for the ACK Flood attack gives the results for an attack period of 20s and the second for 100s. Because there will be a overflow of acknowledge packets the OSPF packets measure and the ACK packets measure will raise alarms. The content of the packets is random causing the packet length measure to raise an alarm. Due to the flood of Acknowledge packets the packet distribution profile will also raise an alarm.

5.6. Results for the Bad Link State Request Attack

The Bad Link State Request attack confuses a router by requesting a Link State that is not in the router's database. Therefore the router will drop the connection with the requesting router and will try to re-establish the connection. The results for this attack are shown in Table 7. The re-establishing of the connection causes the intensity measures to raise alarms as well as the record distribution measures in the same way as in the previous attack. The first

† The record distribution measure also raised an alarm because a new record was added.

row for this attack shows the results if the attack is launched 20 times and the second row shows the results for an attack repetition of 100.

5.7. Results for the Sequence Number Mismatch Attack

The sequence number mismatch attack causes a `SequenceNumberMismatch` event in the victim router causes it to drop the connection. The results are depicted in Table 7. Similar to the previous attacks the router will try to re-establish the connection causing alarms in the intensity measures and the record distribution measures. Again the results are given for 20 and 100 attacks respectively.

5.8. Results for the Sequence Number Plus Attack

In the sequence number plus attack a router modifies the metric in a LSA causing the originator of the LSA to fight back the incorrect information. This will result in instable network topology. The results for the attack are listed in Table 7. The fight-back involves only Link-State Update and Acknowledge packets. This will cause the accompanying measures to raise an alarm.

The total flow of OSPF packets will also be larger than normal causing the intensity measures to raise alarms. These packets will have a length that is not normal thus the packet length measure will raise an alarm. The record distribution measures will also raise alarm because the packet distribution is no longer what normally would be expected. Again the attack was launched 20 and 100 times.

5.9. Results for the Maximum LS Age Attack

In the maximum LS Age attack a router modifies the metric as well as the age field in a LSA to its maximum value of 3600s. Because of this MaxAge LSA all router receiving this LSA will drop that LSA from their topology database. The results for this attack are shown in Table 7. Because the age of the LSA is set to the very rare `MaxAge` the LS Age measure will raise an alarm. The attacked router will fight back the incorrect LSA causing anomalous traffic. This will result in various intensity measure alarms as well as record distribution alarms. Again the attack is launched 20 and 100 times.

5.10. Results for the Maximum Sequence Number Attack

In the maximum sequence number attack the attacking router intercepts a LSA and modifies the metric and the sets the sequence number to the maximum value of `0x7FFFFFF`. Because of a bug in many OSPF implementations the sequence number will not wrap around and the LSA will stay the most 'fresh' LSA for up to 3600s (`MaxAge`). The results of the attack are also given in Table 7. After the attack (which only has to be launched once) the victim router will try to fight back in vain. The fight-back causes a lot of Link-State Update packets and the associated measure will therefore raise alarms. The destination IP record distribution measure also raises alarms because packets are sent host to host which will result in unseen IP addresses for the IDS. The results of the attack are monitored for 20s and 100s, but note that the attack carries on for up to MaxAge.

6. Conclusion

This paper described seven insider attacks on the OSPF protocol as well as an intrusion detection system that is able to detect the insider attacks using statistical anomaly detection. As point of departure the approach of the Jinao project, which was developed at NC State University and MCNC was adopted. Their project demonstrated that the statistical component of the Next Generation Intrusion Detection Expert System (NIDES), developed at SRI, was effective in detecting routing protocol attacks. They had discovered three vulnerabilities in the OSPF protocol. After a thorough analysis of the OSPF protocol four more vulnerabilities on the OSPF protocol were found. The vulnerabilities were exploited in a test network with eight nodes running FreeBSD and GateD as the routing software. The implemented attacks are insider attacks and they give an insider the ability to route traffic through a predetermined node or route the traffic along a non-optimal path in order to degrade, delay or in the worst case deny network services. The Missing Neighbor, Bad LSR and Sequence Number Mismatch attacks modify packet information and spoof their source IP address, causing routers to drop each other from their routing database. This results in non-optimal routes and degraded network service. The ACK Flood attack floods a router with packets (Acknowledge) causing an overload on the receiving router by which the router will not be able to handle any normal routing traffic. The Sequence Number Plus, Maximum Sequence Number and Maximum LS Age attacks, which are borrowed from the Jinao project, modify the routing advertisement packets (LSA) causing an instable network topology.

The intrusion detection system was implemented as three back-ends for the commonly available framework Network Flight Recorder (NFR). NFR was chosen because of its platform independency, which makes it possible to deploy the back-ends easily on wide variety of machines. The developed statistical anomaly detection algorithm is based the NIDES statistical component. Some modifications and improvements to the algorithm were necessary to improve to detection capabilities for our purposes. The algorithm was implemented in three different

back-ends, one for each measure (intensity, counting and record distribution measures). If one would like to use different measures, for any reason, new measures can easily be added to the back-ends. The implementation of the back-ends in NFR's language (N-code) was not without hassle. The N-code lacked functionality that was necessary to implement the algorithm. The most important features that are not included in N-code for our programming needs are a floating-point data type and mathematical functions. To overcome the floating-point need all integers were multiplied with a constant factor to simulate numbers with decimals. Implementing the needed mathematical functions as table look-ups solved the shortcoming of mathematical functions.

The main issue in anomaly detection systems is the selection of threshold levels so that occurrence of false positives and false negatives is minimized. The detection results show that the amount of false alarms is reasonable low and acceptable. The results of the developed intrusion detection system show that the system is very effective in detecting the attacks on the OSPF routing protocol. All attacks caused red alarms in at least five different profiles. The counting measure *packet frequency* did not catch any intrusion. This is due to the randomness of the arrival frequencies causing an almost uniform distribution. Another interesting thing about the results depicted in Table 7 is that only red flags were raised. The attacks cause such chaos in the network that all profiles hit through the roof. The results of the intrusion detection system show potential but there is room for improvement and some questions remain open.

Anomaly detection systems are computationally expensive because of the overhead of keeping track of and possibly updating several system profile metrics. Another issue in anomaly detection is that for new users or new systems new initial profiles have to be created. These initial profiles require an estimation of the expected behavior. Since we only had an isolated network at our disposal, the question remains if NFR and the back-ends scale well when they are deployed in a network with high traffic volumes.

An important issue in intrusion detection is incident response. Matching the triggered alarms against attack signatures, by correlating the different alarms, could improve the quality of information provided to the security agent. Instead of informing the agent with which alarm was triggered information on which attack was launched could be provided.

Another improvement can be achieved if NFR would extend the N-code language with a floating-point data type and some mathematical functions. This would improve the accuracy of the profiles, possibly resulting in even less false positives or negatives.

REFERENCES

[1] Malkin, G., *RIP Version 2.* Network Working Group Request for Comments: 2453, November 1998.

[2] Halabi, S., *OSPF Design Guide.* San Jose, CA, USA: Cisco Systems, April 1996.

[3] Moy, J., OSPF Version 2. Network Working Group Request for Comments: 2328, April 1998.

[4] Lewis, P.H., *Threat to Corporate Computers Often the Enemy Within.* New York, NY, USA: New York Times, March 1998.

[5] Morrissey, J., *Watch Your Back for Insider Network Attacks*, PC Week, September 1997.

[6] Murphy, S.L, and M.R. Badger, *Digital Signature Protection of the OSPF Routing Protocol.* Proceedings of the 1996 Symposium on Network and Distributed Systems Security, IEEE, 1996.

[7] Vetter, B., F. Wang and S. Wu. *An Experimental study of insider attacks on the OSPF protocol.* IEEE International Conference of Network Protocols, p. 293-300, Atlanta, October 1997.

[8] Jou, F., et al. *Architecture Design of a Scalable Intrusion Detection System for the Emerging Network Infrastructure.* Technical Report E296, Advanced Network Research, MCNC, April 1997.

[9] Javitz, H.S., and A. Valdes, *The NIDES Statistical Component Description and Justification.* Annual Report A010, SRI, March 1994.

[10] *Network Flight Recorder*, Version 2.0.3 Research. Network Flight Recorder Inc.

[11] Sedgewick, R., *Algorithms.* Reading, MA, USA: Addison-Wesley, 1984.

[12] *FreeBSD*, Version 2.2.8. Walnut Creek CDROM.

[13] *GateD*, Version 3.5.7. Merit GateDaemon Consortium.

[14] Amoroso, E.G., *Intrusion Detection*, Sparta, NJ, USA: Intrusion.Net Books, January 1999.

[15] *Wisdom and Sense Guidebook*, Los Alamos National Laboratory, Los Alamos, New Mexico.

[16] Smaha, S.E., *Haystack: An Intrusion Detection System.* Proceeding IEEE Fourth Aerospace Computer Security Applications conference, Orlando, FL, USA, December 1988.

[17] NFR N-code reference, Network Flight Recorder Inc. *http://www.nfr.net/nfr/nfr-2.0.2-research/nfrlibrary/reference/n-code/n-code.htm*