An Experimental Study of Insider Attacks for the OSPF Routing Protocol*

Brian Vetter, Feiyi Wang, S. Felix Wu
Secure and Highly Available Networking Group (SHANG)
Dept. of Computer Science
North Carolina State University
{bmvetter,fwang2,wu}@eos.ncsu.edu

Abstract *It is critical to protect the network infrastructure (e.g., network routing and management protocols) against security intrusions, yet dealing with insider attacks are probably one of the most challenging research problems in network security. We study the security threats, especially internal/insider threats, for the standardized routing protocol OSPF. In OSPF, a group of routers collaborate, exchange routing information, and forward packets for each other. If one (and maybe more than one) router is evil or compromised, how can this router damage the whole network? In this paper, we analyze OSPF and identify its strengths and weakness under various insider attacks. Furthermore, to confirm our analysis, we have implemented and experimented one attack, the max sequence number attack, on our OSPF routing testbed. Our attack is very successful against two independently developed router products as it will block routing updates for 60 minutes by simply injecting one bad OSPF protocol data unit.*

1 Introduction

The recent growth in use of the Internet has brought with it a corresponding increased reliance on the network infrastructure which makes it all possible. Routing (e.g., RIPv2, EIGRP, BGP, and OSPFv2) and network management (e.g., SNMPv3/ng) protocols form the very heart of this infrastructure. Until recently, the security of these protocols has not been fully emphasized. However, there is a growing awareness of the potential consequences of attacks aimed the infrastructure, particularly the routing protocols.

* This research is supported in part by the U.S. Department of Defense Advanced Research Projects Agency and the U.S. Air Force Rome Laboratory under contract F30602-96-C0325, and in part by the Center for Advanced Communication and Computing under grant #5-30183.

Research in routing protocol attacks, especially insider attacks, is still in its infancy. Here, “insider” means a trusted entity participating the routing information exchange process. For example, an evil system administrator could misconfigure a router such that the network performance is seriously affected. Or, a compromised network administrator’s password could allow modifications of the routing software kernel itself.

Insider attacks are not well understood today, leading to either underestimates of the possible damage caused by such attacks, or disbelief that such problems can be solved practically and efficiently. Often, when designing a new routing protocol, insider attacks are considered a low priority requirement, or are simply ignored. Therefore, the protocol might have good stability and performance normally, but break down dramatically in the face of insider attacks. It has been pointed out that black hole routers will be a performance killer for distance-vector routing protocols like RIPv2. On April 27, 1997, a router from MAI Network services in Virginia absorbed about 50,000 network addresses which caused much of the internet to be disconnected from 20 minutes to 3 hours. A technical bug was blamed to the MAI’s Bay Network router, but the same attack is very feasible from an evil insider.

It has been identified in [12] that link state routing protocols are more robust against simple failures or even some simple insider attacks. Therefore, in this paper, we examine the security of the standardized Open Shortest Path First version 2 (OSPF)[1][2] routing protocol. OSPF is intended to become the preferred TCP/IP interior routing protocol used in the Internet. At the time of writing, OSPF was in the standards track, in the area director review stage of progression to full standard.

In this paper, we limit our discussion to the security issues concerning the routing protocol itself. Here a distinction must be made between the routing protocol and the process of routing packets. The former is responsible for distributing network topology information among the routers in a network, while the later is the process of forwarding a packet based on its destination address. We do not address direct attacks to this forwarding process, such as a router making intentionally incorrect forwarding decisions, selectively deleting user PDUs, etc. Furthermore, we do not address other threats to the network infrastructure, such as direct attacks to link function or to user data in transit. In no way do we wish to downplay the seriousness of such threats, which simply

lie outside of the scope of the routing protocol itself, and can and should be the responsibility of other security services. In fact, our SHANG group has active research into these areas as well.

In order to support our analysis, we have implemented and experimented with insider attacks in our JiNao/SHANG OSPFv2 routing testbed. Some of these attacks cause negligible effects, while others seriously damage the network. For example, the max-sequence number attack that we present later can block routing updates for one hour by just sending out one bad routing information packet. This attack is made possible by protocol implementation bugs, possibly due in part to unclear RFC specification as well. In fact, two independently developed router products reveal exactly the same weakness. The remaining attack experiments are to appear in [13].

In the next section, we will briefly review the OSPFv2 protocol. Then, we explain the threat models, both outsider and insider. Finally, we describe the implementation of the max sequence number attack and perform analysis about how this can happen.

2 Related Work

The objective of routing protocol security is to *prevent* both outsider and insider (i.e. trusted routers attacks to the network infrastructure. Perlman, in her thesis, proposed a link-state routing protocol that achieves *Byzantine Robustness* [9]. Her protocol is highly robust but with a very high overhead associated with public key encryption, which makes the protocol only theoretically interesting. Murphy and Badger from TIS proposed different digital signature schemes [3][4] to prevent tampered *link-state advertisements* (LSAs). Very recently, Hauser, Przygienda and Tsudik [14]proposed two interesting approaches to reduce the security cost in link-state routing protocols. Smith, Murthy and Garcia-Luna-Aceves recently developed security schemes [6][7] for distance-vector routing protocols with predecessor information (i.e. path finding algorithm(PFA)). Please note that PFA based protocols are in between link state and distance vector. In fact, we consider that PFA is still a link state protocol without fully replicating the routing tables. Furthermore, BBN recently developed one [8] for the Nimrod routing architecture. Most, if not all, secure schemes for attack prevention use some form of public-key encryption, which is quite expensive.

Another approach is to detect (instead of prevent) problems when the network infrastructure is under attacks. Cheung and Levitt at UCDavis developed protocols for detection network infrastructure [15]. Our works [17] are the first one to consider how to efficiently integrate security control with intrusion detection in one single system architecture. The detection protocol in the management plane is being implemented as part of the JiNao, network infrastructure intrusion detection project [16].

Murphy and Badger from TIS proposed a public key signature scheme to protect the integrity of LSAs flooded through the network. With a public key infrastructure, the source router uses its private key to sign the MD5 value for every LSA created. Since the intermediate routers do not know the private key of the source router, they can not tamper the LSAs without being detected. On the other hand, every receiver of LSAs must use the source router's public key to verify its integrity. Therefore, their scheme is very secure against compromised intermediate routers.

There are two potential problems with this public key signature approach: First, public key systems (e.g., RSA) are usually very expensive to run at least in software. Comparing to symmetric authentication schemes, RSA is about 1,000 times slower even with low RSA exponents. Second and more importantly, in order to implement the public key signature scheme in OSPF, we need to modify the standard and upgrade the implementation for all the routers. In the current IETF OSPFv2 draft, there is no protocol field left to store the signature for each LSA. (The only authentication field is for the OSPF PDU which might include many LSAs.) It is our understanding that even the new opaque OSPF option will not help in this case. Many people in the IETF OSPFv2 working group concern that it is still not clear how many serious and realistic attacks can be prevented by using public key signing LSAs. Reasonably, they are not willing to introduce new complexity into the already quite complex OSPFv2 routing protocol.

3 OSPF Background

OSPF is a large and complex protocol, and as such we only provide an overview of those properties of the protocol related to this discussion.

The purpose of any routing protocol is to efficiently distribute dynamic topological information among its participants to facilitate routing calculations upon which packet forwarding decisions

are then based. In a link-state routing protocol such as OSPF, each router is independently responsible for describing the state of its local neighborhood (e.g. links to neighboring networks, routers, and hosts) to the rest of the network.

In OSPF, the first step in the exchange of routing information is the creation of *adjacencies* between neighboring routers. A router first uses a Hello Protocol to discover its neighbors. Once neighboring routers have 'met' via the Hello Protocol, they then go through a Database Exchange Process to synchronize their databases with one another. Only then can neighboring routers become adjacent and exchange routing protocol information.

Information about the state of a router's local neighborhood is then assembled into a *link-state advertisement* (LSA), which is then distributed to every other router by reliable intelligent flooding. The basic flooding process is straightforward: upon receiving an advertisement from a neighbor, a router acknowledges receipt of the advertisement and, if new, forwards the advertisement to all other neighbors. Thus, after a short period of convergence, each router in the network will have an identical topological database of LSAs to be used for routing calculations.

OSPF is an interior routing protocol, designed to be used within a single *autonomous system* (AS). OSPF allows the AS to be divided into groups of networks called *areas*. Each area runs a separate copy of the basic link-state algorithm, and the topological details of the area are hidden from the rest of the AS, reducing routing traffic. All areas are connected by a single *backbone* area, in a logical hub and spoke configuration.

Routers belonging to a single area are called *internal routers*. Routers which belong to more than one area are called *area border routers* (ABRs). All ABRs belong to the backbone by definition. Any router which exchanges routing information with an external AS is called an *Autonomous System Boundary Router* (ASBR).

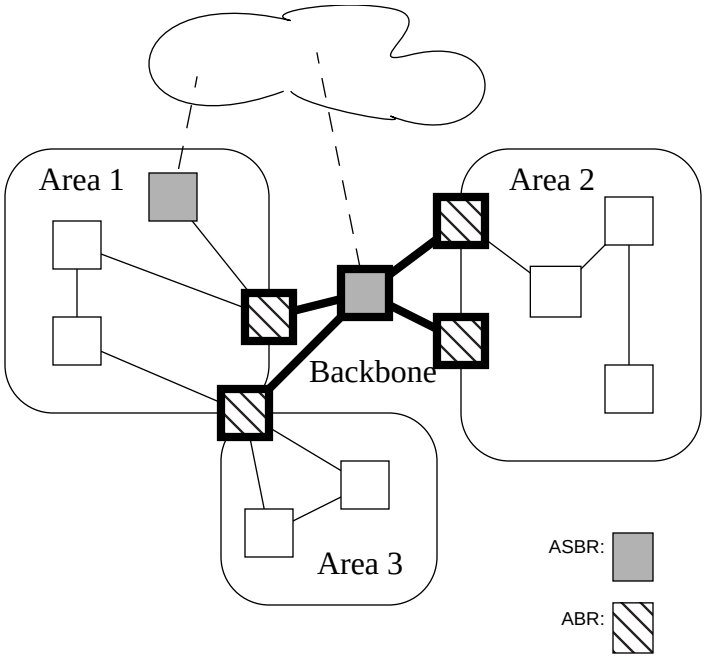

FIGURE 1. OSPF Terminology

OSPF defines five LSA types which correspond to router's respective roles. All routers generate *router links* (type 1) LSAs for each area they belong to, which describe the state and cost of a routers links to that area. Designated routers generate *network links* (type 2) LSAs which describe all routers attached to the transit network (subnet). ABRs generate *summary link* (type 3 & 4) LSAs, which inject into an area a single destination (a network or ASBR respectively) outside of that area. ASBRs generate *AS External* (Type 5) LSAs, which describe a single destination external to the AS. Of the five types, only AS external LSAs are flooded throughout the AS, all others are only flooded within a single area.

To prevent problems caused by 'immortal' LSAs, each contains an *age* field. An LSA's age is constantly incremented, both while being flooded and while installed in any link-state database. If an LSA's age reaches the value *MaxAge* (defined as one hour), it is removed from the router's link-state database and reflooded as a signal for other routers to remove it. An LSA's originator can *flush* the LSA from the system at any time by prematurely setting the LSAs age to MaxAge and flooding it.

It is possible for more than one instance of an LSA to exist in the system at any one time. Thus each LSA has a *sequence number*. When encountering multiple instances, the LSA with the greatest sequence number is considered newer. If the sequence numbers are equal, the age field and finally checksum are used as tie-breakers.

4 Security, Threats & Countermeasures

Providing security in a routing protocol involves protecting the *authenticity* and *integrity* of routing protocol messages. In our context, authenticity is a guarantee of the identity of the source of a particular piece of routing information, and integrity is an assurance that the information transmitted is consistent with the information received.

In any discussion of security, we must consider trade-offs. The vulnerabilities of the protocol must be identified and the risks these vulnerabilities present must be weighed against the costs of protecting them. Cryptography is a powerful security tool, but comes at a performance cost.

Primarily, we are concerned with threats to the integrity and authenticity of routing protocol information. These threats can generally be separated into two classes: *external* and *internal*.

4.1 External Threats

We define external threats as those from non-protocol participants (*outsiders* or *intruder*s), such as an attacker with access to a link between routers. Such an intruder might delete, modify, replay, or forge protocol packets. This poses a threat to the flow and content of routing updates (LSAs), to the neighbor relationships of existing routers, and of an outsider becoming a protocol participant. Threats from outsiders can be countered using *authentication*, i.e. requiring that routers participating in the protocol possess shared secrets which, by definition, an outsider does not have access to.

OSPF requires that all protocol exchanges be authenticated. One type of authentication used is a simple password scheme, with a plaintext password included in the OSPF header. This scheme has a minimal performance impact, but unfortunately is easily compromised since the secret password itself is transmitted in the clear, and is thus vulnerable to eavesdropping by an intruder.

Fortunately, [2] defines a much stronger *cryptographic authentication* type, in which OSPF routers on a common network / subnet share a secret key which is used to generate a cryptographic (e.g. keyed MD5) message digest for each protocol packet. In addition, a monotonically increasing sequence numbering scheme is used to prevent replay attacks. With cryptographic authentication, the integrity and authenticity of protocol exchanges between neighboring routers is secured from outside threats.

Because the secret is shared, symmetric cryptography is well suited for use in this application, and thus relatively fast schemes such as keyed-MD5 can be used. It is widely accepted that the substantial security benefits gained by using cryptographic authentication justify the performance costs involved.

While cryptographic authentication does prevent forging, replay, and modification of protocol messages by outsiders, an intruder can still block (delete) protocol messages transmitted on the link. This is an attack to link function, and as such is outside the scope of routing protocol security and of our discussion. However, we do make the following observations: if an intruder interferes with the neighbor relationship by deleting messages, the effect is to prevent the use of that link. Arguably, this is actually a desirable result, as it repels traffic from the link, effectively isolating the attacker.

4.2 Internal Threats

We define internal threats as those from protocol participants (*insiders*), such as subverted or faulty routers. As we have seen, the integrity and authenticity of routing information can be protected with cryptographic methods. However, in the case of insider threats, conventional cryptographic solutions are impractical because to uniquely authenticate every router in a symmetric scheme, each *pair* of routers must share a secret, requiring $O(N^2)$ keys.

Asymmetric solutions using public key cryptographic digital signatures [3][4]have been proposed, but the great computational burden of asymmetric cryptography has limited the acceptance of these proposals. Some have argued that the threat to LSAs in transit is a minor concern when facing a subverted router, as this is only one of many possible attacks in this situation, and thus the greater cost of a signed LSA scheme is not justified by the benefits. In order to evaluate the

cost of such a scheme realistically, we must first consider the threats which it is in intended to prevent, and those it cannot.

A router essentially plays two distinct roles in the routing protocol - generating local information (LSAs) and forwarding the LSAs of other routers. We first consider the possible threats to the protocol from an insider modifying or deleting the LSAs of other routers.

4.2.1 Modifying Information

Since multiple instances of an LSA are compared to determine which is newer, by examining their sequence number, age, and checksum fields, we know that *any* changes made by a bad of faulty intermediate router to an LSA in transit will result in one of two cases in the receiver(s) of the LSA. The modified LSA will be (1) *rejected*, i.e. considered to be an older instance or the same instance and is thus discarded, or (2) *accepted*, i.e. considered to be a new or newer instance and is thus installed in the receiver's database and propagated by flooding. Clearly, case (1) does not pose a threat to the protocol. In case (2), the possible results depend on the bad router's topological position in the network.

We first consider the case where the set of bad routers does not partition an area. That is, there must exist at least one good path between any two good routers in the area. It follows from this assumption that reliable flooding exists throughout the area. Thus, deletion of LSAs by bad routers is not a threat. We also know that because reliable flooding exists, a modified (bad) LSA will eventually be flooded back to the original (good) router which 'owns' the LSA. Thus the good router will recognize the bad LSA and treat it as an out of date self-originated LSA, either generating an updated LSA or flushing the bad LSA. Thus, in this case, any modified LSA can be *detected* by the originator (or the claimed originator if the LSA is a complete forgery), and furthermore the originator will attempt to fix the problem. If the attacker's modifications continue, the originator and the bad router will 'argue' back and forth, and the currently accepted instance of the LSA at any other router will depend on topology as long as the attack persists. Such a persistent attack is more detectable to higher level security services than a 'hit and run' attack.

We have seen that changes made to LSAs in transit can be detected under our assumptions. We can use this result to lighten the burden of digital signatures. Because an LSA is signed once but

verified many times, the idea of *selective verification* is attractive. Then we could require all LSAs to be signed, but only verified by the other routers if the originator has detected modification, in which case the originator 'enables' verification of its LSAs via a flooded signal. Or, LSAs could be left unsigned unless modification is detected, at which time the originator could switch to signed LSAs.

We now consider a bad internal router which *does* partition the area, i.e. the bad router is the only path between at least two good routers. In general, we can view the area as divided into several groups of good routers, or *fragments*, between which the bad router completely controls all communication.

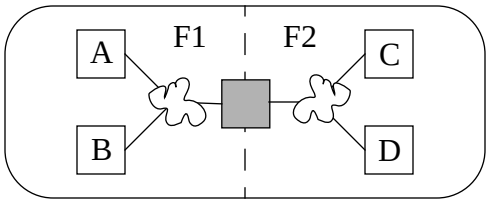

FIGURE 2. Area partition example

In the Figure 2 above, the bad router (shaded) partitions an area into fragments F1 and F2.

One obvious threat is that of the bad router failing to forward LSAs from one fragment into the other. This has the effect of isolating the bad router, i.e. causes it to not be used to forward user traffic, and we note that the resulting unreachability should detectable by higher level (e.g. network management) services.

When considering LSA modification, we first note that if the modified LSA is introduced into the *same* fragment as the LSA's originator, we have the case of a non-partitioned area which we have previously discussed, and the owner can detect the modification.

Now consider an LSA originated at A for an existing destination in fragment F1, which is modified and introduced into fragment F2. Clearly, this cannot be detected by A. We also note that this cannot affect routing internal to F2.

Finally, we can see that unless the change is to effectively delete the destination, routing from F2 to the destination in F1 will actually be correct, since the bad router is along the only path to the

mis-advertised destination, routing within the fragment. Thus any change in routing within the fragment can again only be to decrease reachabilty. In other words, it is vacuously true that the bad router cannot attract excess traffic, as it is already along the only path to the advertised destination. Thus we argue that in this case, the greatest threat to the protocol from LSA tampering is that of causing unreachabliliy, note that this helps to isolate the bad router and increases detectability.

4.2.2 Generating Bogus Information

It is fundamentally difficult for the protocol itself to provide protection against a bad router's origination of bogus local information. Cryptography, for instance, cannot prevent a valid router from generating incorrect information. Luckily, the nature of OSPF does provide some inherent protections from internal threats.

First, since routing within each area is independent, bad information in one area cannot affect routing in within other areas. Second, the natural duplication of information in a link-state routing protocol lends itself to *corroboration*. Any time several routers have access to the same set of information (e.g. two ABRs common to a particular area), they have the potential of checking each other and detecting problems[3][4]. Finally, the SPF calculation will not consider a link unless the database contains a corresponding LSA from the other end of the link, preventing a router from claiming non-existent transit networks.

Murphy's digital signature scheme also includes a mechanism to limit the destinations a router may advertise to a configurable range. This legal range information is provided by the *trusted entity* along with the public keys.

5 A 'Real Life' Attack Example

Implementation bugs can also pose serious security threats to a routing protocol. Our research has led to the discovery of just such a threat existing in at least two OSPF implementations. The result of the attack is that a subverted router can modify any routers LSA and the damage cannot be self-corrected for extended periods (up to one hour).

In the OSPFv2 specification [1], the sequence number field of the LSA header is a signed 32-bit integer used to detect old and duplicate link state advertisements. When an LSA is received by a router, it is compared with any existing instance of the LSA to determine which is newer. If the sequence numbers differ, the instance with the greater sequence number is kept.

A router uses `0x80000001` to number the first instance of any LSA it originates (the value `0x80000000` is reserved and unused). When a new instance of a particular LSA is generated, its sequence number is incremented by 1 (one), thus successive instances of an LSA have successive sequence numbers. When an attempt is made to increment the sequence number past the maximum value (`0x7fffffff`), the current instance of the LSA must first be flushed from the routing domain before the new instance of the LSA is generated with sequence number `0x80000001`. If the router did not first flush the existing LSA, a newly generated instance of the LSA would be rejected by other routers because the obsolete LSA has the larger sequence number and is thus, by definition, “newer”.

In normal operation, the chance of a router actually reaching the maximum sequence number is extremely small. The protocol mandates that a router not generate instances of a particular LSA more frequently than every MinLSInterval = 5 seconds. Thus it should take a minimum of $5*2^{31}$ seconds or about 340 years for an LSA’s sequence number to reach `0x7fffffff`. However, a malicious or faulty router could certainly speed up this process, as we have done in our experiments.

In OSPF, it is possible for a router to receive an obsolete instance of its own LSA with a sequence number greater than that of the current instance. This could happen if the router loses state (e.g. goes down and comes back up) or if the network has been partitioned. The router must then generate a new instance of that LSA with a greater sequence number, or if the LSA is no longer needed, flush the LSA from the system. Thus it is possible that a router could receive its own obsolete LSA with sequence number `0x7fffffff`, in which case it should be flushed before a new instance with a ‘greater’ sequence number (i.e. `0x80000001`) is generated. We have found that in two implementations of OSPF, this process is not implemented correctly. Specifically, the router does not first flush the obsolete LSA.

In the current OSPF protocol, a subverted router can mount an attack which takes advantage of this bug as follows. A subverted router can modify the contents of any LSA it receives at will, set its sequence number to `0x7fffffff`, and flood it back to the sender. Based on the sequence number, all other routers will accept this bad LSA as newer and replace the good LSA in the topological database. When the LSA's true originator receives the bad LSA, it will generate a corrected LSA with sequence number `0x80000001`. However, due to the bug, it will not flush the bad LSA before transmitting the new LSA, which is thus *rejected* by the other routers as 'older'. The bad LSA will therefore remain in the system until it naturally ages out, in a maximum of one hour. We note that the bad router need only send *one* copy of each LSA it wishes to attack. We also note that in order to correct the network before the bad LSA ages out naturally, all routers within an area would have to be reset *simultaneously* to remove the bad LSA from the system entirely, otherwise it would simply re-propagate throughout the system.

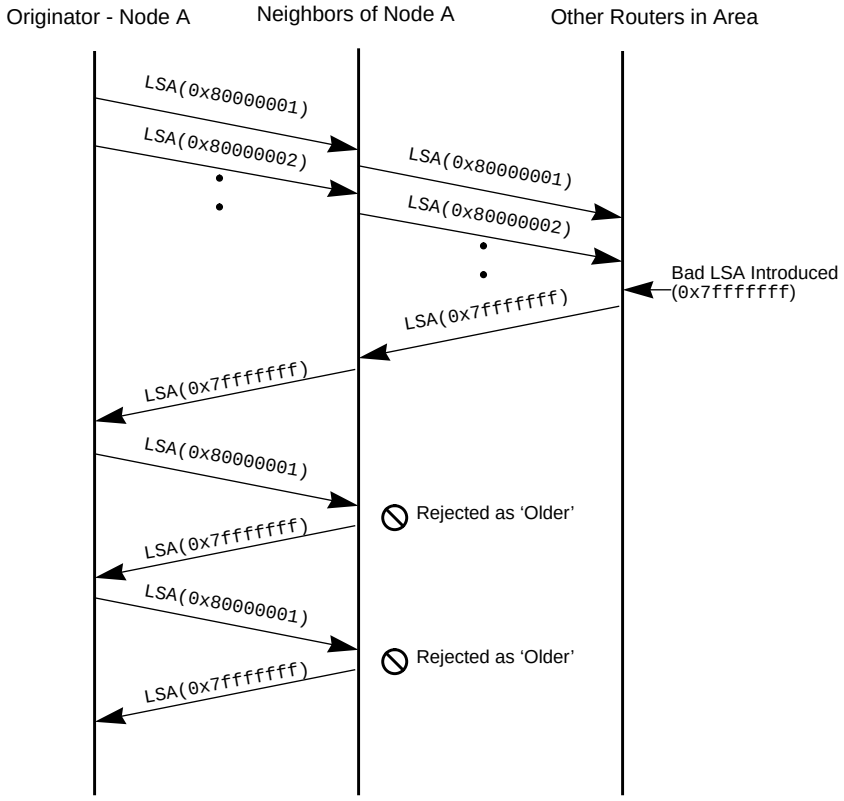

FIGURE 3. Illustration of max sequence attack

We have implemented this attack in the lab in our OSPF testbed, and have verified that two independent and well-known implementations are susceptible. Figure 4 is an excerpt of a router's log of an acutal attack from our OSPF Testbed, and Figure 3 shows this attack schematically. A diagram of the OSPF Testbed (Figure 5) is also included. ***We regret that we cannot publicly disclose the specific faulty implementations at this time. We will gladly do so after the vendors have been contacted.***

6 Remarks

Although OSPF has been considered as a very robust routing protocols against various failure models, we have demonstrated that a bad router (insider) feasibly could cause a global damage to the network with very little effort. Even worse, when the max sequence attack is launched, it is difficult to trace the source of the attack. The key problem is that in the OSPF routing protocol design we did not consider the case of insider attacks. We have also learned that, once the protocol has been standardized or even just stabilized, it is technically and politically hard to add new mechanisms to enhance the protocol's security. OSPF is one typical example, and our demonstrated attack is just the tip of the iceberg. In fact, most existing routing protocols are vulnerable one way or the other to the insider attacks. Please note that even in IPv6 these insider attacks will still be valid as IPSEC is designed for outsiders. The lesson we learned is that we need to consider insider attacks right from the beginning of the protocol development process. Security analysis can not be an add-on process.

One possible (and practical) solution is to detect and isolate insider attackers with intrusion detection systems and application-layer firewalls. In other words, the solution is offered in the network management plane and not modifying anything related to the target network control protocol (e.g., OSPF). Our preliminary research results [16][17] indicate that this is indeed possible to achieve. Please refer to our research web site http://shang.csc.ncsu.edu for further information along this line.

```
/* The following log file trace one particular SumNet LSA originated by Water to show
   the attack example. The first part show that water multicast a update packet which
   include one SumNet LSA with checksum 0x12b7 and seq # 0x80000001
 */

OSPF RECV 192.0.5.3(eth2) -> 224.0.0.5  Link State Update  Vers: 2  Len: 56
OSPF RECV RouterID: 192.0.5.3  Area: 0.0.0.0  Checksum: 0x1fd7
OSPF RECV Auth: Type: 0 Key: 00000000.00000000
OSPF RECV Advertisement count: 1
OSPF RECVSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 1
OSPF RECVLen:  28  Seq #: 80000001  Checksum: 0x12b7
OSPF RECVMask: 255.255.255
OSPF RECVTos 0 metric: 10

/* Jinao did the same thing which show that the LSA already installed in jinao's
   database.
 */

OSPF SENT 192.0.7.3(eth1) -> 224.0.0.5  Link State Update  Vers: 2  Len: 56
OSPF SENT RouterID: 192.0.7.3  Area: 0.0.0.0  Checksum: 0x1dd6
OSPF SENT Auth: Type: 0 Key: 00000000.00000000
OSPF SENT Advertisement count: 1
OSPF SENTSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 2
OSPF SENTLen:  28  Seq #: 80000001  Checksum: 0x12b7
OSPF SENTMask: 255.255.255
OSPF SENTTos 0 metric: 10

/* Attacker Shang changes the seqence number of this LSA from 0x80000001 to
   0x7fffffff as well as the metric, then sends it back to Jinao
 */

OSPF RECV 192.0.7.2(eth1) -> 192.0.7.3  Link State Update  Vers: 2  Len: 56
OSPF RECV RouterID: 152.1.61.120  Area: 0.0.0.0  Checksum: 0x6cf7
OSPF RECV Auth: Type: 0 Key: 00000000.00000000
OSPF RECV Advertisement count: 1
OSPF RECVSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 13
OSPF RECVLen:  28  Seq #: 7fffffff  Checksum: 0xb421
OSPF RECVMask: 255.255.255
OSPF RECVTos 0 metric: 16777215

/* Jinao accepts and acknoledges the LSA update  */

OSPF SENT 192.0.7.3(eth1) -> 224.0.0.5  Link State Ack  Vers: 2  Len: 44
OSPF SENT RouterID: 192.0.7.3  Area: 0.0.0.0  Checksum: 0x7b79
OSPF SENT Auth: Type: 0 Key: 00000000.00000000
OSPF SENTSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 13
OSPF SENTLen:  28  Seq #: 7fffffff  Checksum: 0xb421

OSPF RECV 192.0.5.3(eth2) -> 224.0.0.5  Link State Update  Vers: 2  Len: 56
OSPF RECV RouterID: 192.0.5.3  Area: 0.0.0.0  Checksum: 0x1fd7
OSPF RECV Auth: Type: 0 Key: 00000000.00000000
OSPF RECV Advertisement count: 1
OSPF RECVSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 1
OSPF RECVLen:  28  Seq #: 80000001  Checksum: 0x12b7
OSPF RECVMask: 255.255.255
OSPF RECVTos 0 metric: 10


OSPF RECV Area 0.0.0.0 192.0.5.3       -> 224.0.0.5: LS UPD: received less recent LSA
RECVSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 1
RECVLen:  28  Seq #: 80000001  Checksum: 0x12b7
HAVESumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 14=(13+1)
HAVELen:  28  Seq #: 7fffffff  Checksum: 0xb421


OSPF SENT RouterID: 192.0.7.3  Area: 0.0.0.0  Checksum: 0x7b65
OSPF SENT Auth: Type: 0 Key: 00000000.00000000
OSPF SENT Advertisement count: 1
OSPF SENTSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 21
OSPF SENTLen:  28  Seq #: 7fffffff  Checksum: 0xb421
OSPF SENTMask: 255.255.255
OSPF SENTTos 0 metric: 16777215

OSPF RECV 192.0.5.3(eth2) -> 224.0.0.5  Link State Update  Vers: 2  Len: 56
OSPF RECV RouterID: 192.0.5.3  Area: 0.0.0.0  Checksum: 0x1fd7
OSPF RECV Auth: Type: 0 Key: 00000000.00000000
OSPF RECV Advertisement count: 1
OSPF RECVSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 1
OSPF RECVLen:  28  Seq #: 80000001  Checksum: 0x12b7
OSPF RECVMask: 255.255.255
OSPF RECVTos 0 metric: 10
```

```
OSPF RECV Area 0.0.0.0 192.0.5.3       -> 224.0.0.5: LS UPD: received less recent LSA
RECVSumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 1
RECVLen:  28  Seq #: 80000001  Checksum: 0x12b7
HAVESumNetId: 192.0.2          AdvRtr: 192.0.5.3        Age: 20=(13+7)
HAVELen:  28  Seq #: 7fffffff  Checksum: 0xb421

OSPF SENT 192.0.5.2(eth2) -> 192.0.5.3  Link State Update  Vers: 2  Len: 56
OSPF SENT RouterID: 192.0.7.3  Area: 0.0.0.0  Checksum: 0x7b60
OSPF SENT Auth: Type: 0 Key: 00000000.00000000
OSPF SENT Advertisement count: 1
OSPF SENTSumNetId: 192.0.2         AdvRtr: 192.0.5.3        Age: 26
OSPF SENTLen:  28  Seq #: 7fffffff  Checksum: 0xb421
OSPF SENTMask: 255.255.255
OSPF SENTTos 0 metric: 16777215
```

FIGURE 4. Example Attack Log File from Router "Jinao"

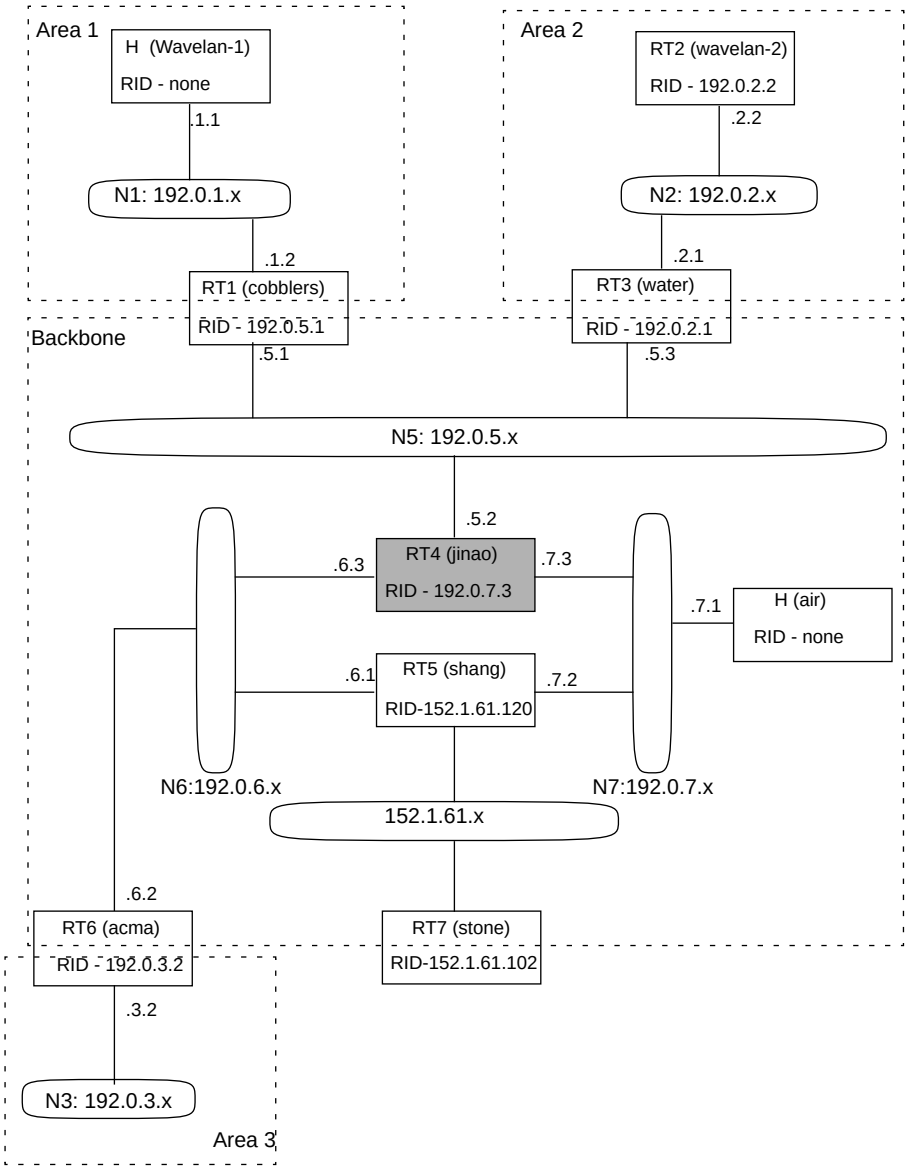

FIGURE 5. OSPF Testbed

References

[1] John Moy, OSPF Version 2, RFC 1583, March 1994.

[2] John Moy, OSPF Version 2, Internet Draft: draft-ietf-ospf-version2-11.txt, April 1997. *

[3] S.L. Murphy, Badger, M.R., and Wellington, B., OSPF with Digital Signatures, Internet Draft: draft-murphy-ospf-signature-03.txt, March 1997. *

[4] S.L. Murphy, "Digital Signature Protection of the OSPF Routing Protocol", *IEEE/ISOC Symposiums on Network and Distributed System Security*, 1996.

[5] OSPF Working Group Mailing List Archive, ftp://gated.cornell.edu/pub/lists/ospf

[6] Bradley R. Smith, J.J Garcia-Luna-Aceves, "Securing the Border Gateway Routing Protocol", *Proc. Global Internet'96*, London, UK, 20-21 November 1996.

[7] Bradley R. Smith, Shree Murthy, J.J. Garcia-Luna-Aceves, "Securing Distance-Vector Routing Protocols", *IEEE/ISOC Symposiums on Network and Distributed System Security*, 1996.

[8] Karen E. Sirois, Stephen T. Kent, "Securing the Nimrod Routing Architecture", *IEEE/ISOC Symposiums on Network and Distributed System Security*, 1997.

[9] Radia Perlman, "Network Layer Protocols With Byzantine Robustness," MIT/LCS/TR-429, October, 1988.

[10] Radia Perlman, "Fault-Tolerant Broadcast of Routing Information", *Computer Networks*, no. 7, p. 395-405, 1983.

[11] Brijesh Kumar, "Integration of Security in Network Routing Protocols", *SIGSAC Review*, vol. 11, no. 2, Spring, 1993.

[12] Gregory G. Finn, "Reducing the Vulnerability of Dynamic Computer Networks", Technical Report, University of Southern California, ISI, June 1988.

[13] Brian M. Vetter, "Routing Protocol Security", MS Thesis, in preparation, NCSU, July 1997.

[14] R. Hauser, T. Przygienda, and G. Tsudik. "Reducing the Cost of Security in Link-State Routing", In *Internet Society Symposium on Netowork and Distributed Systems Security*, 1997.

[15] Steven Cheung, Karl N. Levitt, "Protecting Routing Infrastructures from Denial of Service Using Cooperative Intrusion Detection", submitted to publication, 1997.

[16] Y. Frank Jou, et al., "Architecture Design of a Scalable Intrusion Detection System for the Emerging network Infrastructure", DARPA Order Number: E296, April, 1997

[17] S. Felix Wu, et al., "Intrusion Detection for Link-State Routing Protocols", IEEE Symposium on Security and Privacy 5 Minute Talks, 1997.

***** *Internet drafts should be considered works in progress.*