DISTRIBUTED DETECTION OF INTRUSIONS

Zoltán Czirkos – Gábor Hosszú

SUMMARY

In this article we present a distributed network intrusion detection method. Network hosts implementing this method create a peer-to-peer, application level network of nodes with equal responsibility, which is then used to share data of detected intrusion attempts. Information collected by the nodes therefore does not remain scattered at the probes; rather it is collected and used to create a collective knowledge base. Being able to process data in a distributed manner, nodes are able to enhance security of each other.

NETWORK INTRUSION DETECTION

Well known and widely used distributed intrusion detection systems are usually centralized and are used only to collect data. Decentralized systems, which are capable of taking preventive measures, too, only appeared recently.

A complex range of issues in distributed intrusion detection originates from the problem of the collection and global processing of intrusion data originating from various sites. In our terminology we say that *suspicious events* processed together give the opportunity to detect an *attack or intrusion* – that is, a single suspicious event does not imply necessarily an attack on its own. Also, attack patterns intended to be detected by distributed intrusion detection can be very diverse. The more common properties we try to find among suspicious events detected at different sites, the more likely it is to be unable to recognize the relation between events, which can be in turn related to the same attack – and thus we lose the power of distributed intrusion detection itself.

RELATED WORK

The distributed intrusion detection system named Komondor presented in our article creates a peer-to-peer based distributed database, which is used by its participants to share information of suspicious events and process them in a distributed manner.

A well-known group of peer-to-peer based application level networks is the group of *distributed hash tables*. DHT's store key-value pairs: for every piece of information stored (*value*) is assigned a *key*, which is used to refer to it. Keys of these key-value pairs are mapped to a large range of numbers using a hash function, which range is usually as large as 128 or 160 bits for most networks. Each node is also assigned an identifier from this numerical range. Also every node stores those key-value pairs, which have their keys hashed closest to the identifiers of the nodes themselves. Thus every piece of information is mapped to one (or more, if necessary) nodes in the network. Of course for a key-value pair to be looked up, the exact key is required to be known precisely.

The e-mail spam detection method named Spamwatch is also built on a DHT [4]. This application is a plugin module for an e-mail client. The filtering of spam messages itself is actually carried out by the users by hand. If an e-mail is tagged as spam by any user, Spamwatch stores this information in the Tapestry-based distributed database built by running instances of the plugin. This way, the same spam message can be deleted automatically at other users. Unfortunately this method is not that effective, as the content of specific spam messages can be different.

The PROMIS intrusion detection system (which is not based on a DHT, but on an unstructured P2P network) can be used to enhance protection of computers against viruses [3]. Nodes of this system do not share information of specific intrusion attempts, but rather their number and frequency. The aggregated frequency of suspicious events detected by each node is used to automatically fine-tune the security level of the web browsers. This method gives an overall protection against malware, but decreases usability of computers at the same time.

THE KOMONDOR INTRUSION DETECTION SYSTEM

Our proposed intrusion detection system is built on a DHT network, namely Kademlia. The main goals of the proposed system are the following:

(A) To create a *stable and fast* application level network to share data of intrusion attempts.

(B) To enhance the protection of each individual participator in the system by the global processing of these events.

(C) To create a decentralized network, so that by attacking only some of the nodes, it is not possible to neutralize the whole system.

The nodes participating in the system organize themselves into an application level network, called the overlay. The reliability and speed of sharing information largely depends on the topology of this network. To implement decentralization, we have chosen to use a peer-to-peer based distributed hash table, opposed to a centralized system, which would pose a higher risk of failure. The DHT network named Kademlia was selected, as our research showed that it is the most feasible for being the substrate of Komondor. Kademlia is a DHT with 160 bit node identifiers and a binary tree topology [1].

DHT networks are capable of distributing data to be stored among nodes evenly. The Komondor intrusion detection method developed by us assigns a properly selected key to each detected intrusion attempt or other suspicious event. This way, even if they are originated from different sites (probes) in a network, they will be collected at a single node, where the global processing can take place.

In the case of Komondor, the keys are IP addresses of suspected attackers, values are the reports of suspicious events detected. A node detecting a suspicious event hashes the IP address of the attacker, and the key generated this way is used to send the report into the DHT. Thus if an attacker tries to attack more than one host protected by the Komondor system in a short time, the reports generated at different points of the network will be collected at a single node, as having generated the same key, any node will store the report at the same node of the DHT. This one, called the collector node, will have as much information as possible to decide whether the overlay detected an attack or not. If an attack is detected, other nodes must be alerted to enable them enhancing their protection against the recognized attacker.

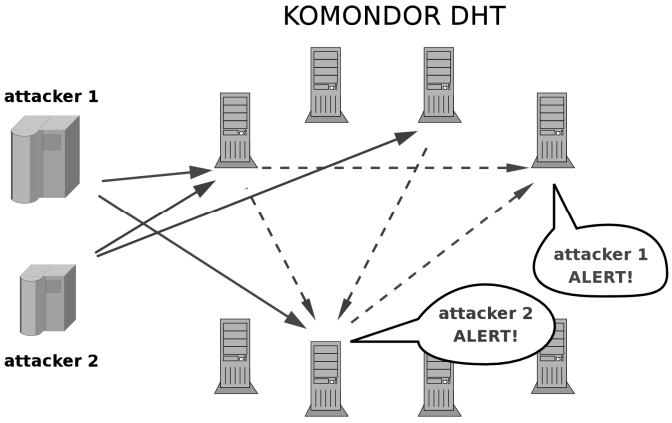

Figure 1.
Attacks detected by the Komondor system

Collecting information of possible attacks and alerts require two entirely different types of messaging in the overlay network. Having detected a suspicious event, the node that detected it uses the hash function to select which node will store the report, and then sends it. This is the same as the usual STORE message used in Kademlia [1]. The collector node, according to the reports sent to it, may respond automatically – this is unusual in normal DHT networks. The response is a broadcast alert message.

Broadcast messaging is unusual in P2P networks; however it is a requirement of the Komondor system. All nodes must be alerted as soon as possible if an attacker is detected. For these two unusual messaging method we had to modify the overlay, therefore the Komondor system uses its own, unique Kademlia implementation.

RELIABILITY OF STORING DATA IN THE KADEMLIA NETWORK

In the Kademlia overlay, every node has its unique 160 bit network identifier (NodeID), which is used for routing in the overlay network. The routing table used by the nodes is actually a ta-

ble containing application level network address and IP address assignments, enabling nodes finding a node with a given NodeID. The literature of Kademlia refers to these tables as *k-buckets*. Each k-bucket contains at most *k* assignments. All bits of the 160 bit NodeID has a k-bucket; each node stores IP addresses of other nodes, which have their NodeIDs sharing 0, 1, 2, … 159 bits in common with its own. This gives the scalability of the network: every node has to store at most *160*k* IP addresses, and this is independent of the number of nodes in the network.

In contrast to other DHT networks, Kademlia nodes do not route messages from one node to another. If a node wishes to communicate with another one, it looks up the IP address of the destination – for this process it asks for k-buckets of other nodes in the network. If the address of the destination is known, the two nodes can communicate directly, using UDP packets. For our Komondor system, this is a feasible property of the overlay; if, having detected a suspicious event, a node has to send such messages, the IP address of the collector node is only required to be looked up once. Usually it is expected, that a probe will detect more than one suspicious event in a short time – for every successive event, the IP address of the collector node is already known, and the report can be sent directly and immediately.

If we denote the probability of a single lookup being successful with P', then 1-P' is the probability that the given lookup will fail. However, this does not necessarily mean that the node cannot lookup a value, as it has other neighbors as well to send the lookup request to. By sending the lookup request to k neighbors, the probability of the lookup failing decreases to $(1-P')^k$. Thus, the probability of the whole lookup procedure being successful is $1-(1-P')^k$. If P' is known, we can solve this equation for k to get the required replication level for any Kademlia network – this is described in our article [5] in greater detail.

This direct communication between nodes has certainly an impact on reliability. If the destination node cannot be accessed, the reports of suspicious events cannot be sent to it. If the collector node – due to any network error – can be reached by only some of the nodes, suspicious event reports will not be collected at a single spot in the network, thus the accuracy of intrusion detection is decreased. This problem can be solved by using replication. Data is not sent to a single node (which is closest to the hashed key), but *k* nodes in the network, the probability of finding an appropriate node to collect data increases rapidly. However an overly high value for *k* is also unfeasible, as it increases network load unnecessarily [2].

BROADCAST (ONE TO ALL) MESSAGES IN KADEMLIA

The use of broadcast messages is not common in P2P networks. When designing these networks, researchers usually assume, that tens or hundreds of thousands of nodes will be in the overlay; one to all messages are neither required, nor feasible due to the large amount of network traffic generated. In some cases, for example arbitrary or partial keyword searches can make use of this kind of messages [5]. Also in the Komondor system, when the collector node, having processed reports of possible suspicious events, detects an attacker, initiates a broadcast message over the application level network.

This type of messaging can be efficiently built upon the built in topology of the overlay. The Kademlia protocol requires, that all peers know at least one peer from every neighboring subtree (if there are any) [1]. It is a fast and efficient method to build the broadcast subsystem upon these k-buckets, as this way it does not rely on any supplementary broadcast routing tables to be created.

Our broadcast messaging works as follows. The node initiating the broadcast sends the message to the largest neighboring (half) subtree; it sends it to the second largest (quarter) neighboring subtree, the eighth subtree and so on. Every node receiving the broadcast sent this way is responsible to forward it in its own subtree (see Figure 2). The message thus is delivered in exponential time. Replication can be used to enhance speed and reliability in the network (and requires messages to be tagged with a semi-unique identifier, as it can result in duplicate deliveries).

Replication is needed even more for broadcast messages than for single STORE messages in Kademlia [5]. If a data packet, which is sent to a node responsible for forwarding it in half of the overlay, at least 50% of the nodes will not get that message. We studied this phenomenon in our own simulator application [5]; the level of replication needed here for reliable functioning of the broadcast subsystem is however lower, than the one for data store and lookup requests.

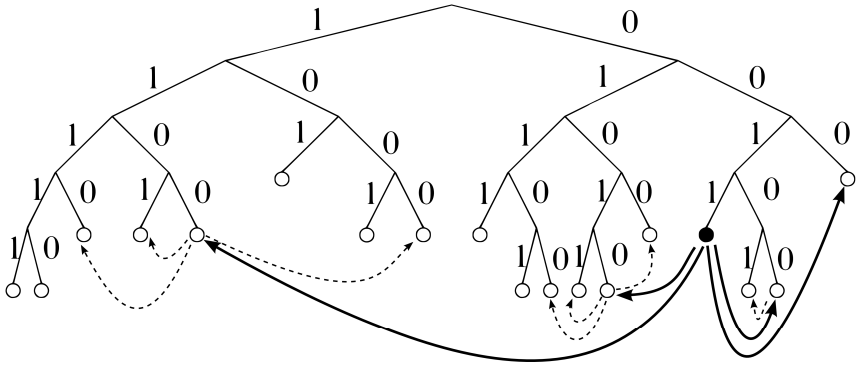

Figure 2.
Broadcast messages in the Kademlia overlay

CONCLUSION

The first, proof of concept implementation of our Komondor system has undergone a successful three year test. During this period it collected enough data to estimate the efficiency of this intrusion detection method. The detection and protection based on IP addresses is particularly useful against many kinds of attacks; and can also be ineffective for others.

(A) A common scenario is that an attacker may try to guess weak passwords of users simply by the brute force method. One may not think, but this is still a very common type of attack on the Internet – our Komondor system detects many of these every single day.

(B) Other common suspicious events involve virus attacks. However, these are not deliberate, planned attacks against a single host or a network. Viruses usually choose IP addresses randomly to spread themselves through remotely exploitable security holes. The 32 bit range of all IP addresses in the world is extremely large for a single infected machine to come up randomly with more than one IP from a subnet protected by the overlay. The Komondor intrusion detection method is therefore not efficient against viruses.

Grouping reports by IP addresses is not the ultimate solution against attacks originating from multiple sites. Attacks can however not only grouped by that. The property of events selected to correlate them can be anything in later versions of Komondor. The only requirement for the DHT is that for events assumed to be correlated, a common key must be generated. This ensures, that any place events are generated at, they will be collected at a common node for processing. Events can even be correlated by multiple keys, of which the IP address of the attackers is only one possibility.

REFERENCES

[1] P. Maymounkov, D. Mazieres. *Kademlia: A Peer-to-peer Information System Based on the XOR Metric.* In Proc. Of IPTPS02, Cambridge, USA, March 2002.

[2] S. Rhea, D. Geels, T. Roscoe, J. Kubiatowicz. *Handling Churn in a DHT.* In Proc. Of USENIX Technical Conf., June 2004.

[3] Vasileios Vlachos, Diomidis Spinellis. *A Proactive Malware Identification System based ont he Computer Hygiene Principles.* Information Management of Computer Security, 15(4):295-312, 2007.

[4] Feng Zhou, Li Zhuand, Ben Y. Zhao, Ling Huang, Anthony D., Joseph, John Kubiatowicz. *Approximate Object Location and Spam Filtering on Peer-to-peer Systems.* In ACM Middleware 2003.

[5] Czirkos Z., Hosszú G. *Peer-to-peer based Intrusion Detection.* INFOCOMMUNICATIONS JOURNAL LXIV:(I) pp. 3-10. (2009)

[6] Sameh El-Ansary, Luc Onana Alima, Per Brand, Seif Haridi. *Efficient Broadcast in Structured P2P Networks.* Lecture Notes in Computer Science, pp. 304-314, 2735/2003.