White paper (draft)

Revere

Contents

Chapter 1 Introduction -- Motivation and challenges

MOTIVATION

- Malicious code
- Virus
- Purpose of Revere

CHALLENGES

Chapter 2 Goals

SECURITY

FAST AND ROBUST DISSEMINATION

- Transmission of virus signature data
- Transmission of control information

SCALABILITY

- Scalability in terms of information storage
- Scalability in terms of performance
- Scalability for security and robustness consideration

HETEROGENEITY

LOW-OVERHEAD

MARKETABILITY

Chapter 3 RBone

RBONE PROPERTIES IN TERMS OF DISSEMINATION

- Why not single path?
- Why adaptivity?

Bandwidth aware but not frugal
Latency important but okay at human level
Latency metrics

HOW SHOULD RBONE WORK?

Session
RBone organization
RBone Autonomy

Multipath
Choke point

IP multicasting as a building block?
Active Network support for Revere
Achieving optimal performance
Reliability

Chapter 4 Security of Revere

DATA MESSAGES

Security of data messages
Ensure the security of data messages

CONTROL MESSAGES

Security of control messages

PROTECTION AND INTRUSION DETECTION IN A REVERE NODE

PROTECTION AND INTRUSION DETECTION IN A VIRUS DISSEMINATION CENTER

Protection
Intrusion detection
Revocation

KEY MANAGEMENT

Chapter 1

Introduction –- Motivation and challenges

MOTIVATION

Malicious code -> virus -> Revere

Malicious code

Malicious code residing in a computer system brings unpredictable threats to the information handling capability of a computer by attacking the system internally or externally.

Malicious code is becoming increasingly common and can tremendously cause serious economic and military damages.

As the rapid development of network, malicious code is also able to propagate more quickly than ever before and in a much easier way than through a floppy disk. It can be introduced into a system through many different ways, such as by embedding itself in a downloadable executables, attaching itself in an email, actively breaking into some machines, and even some unknown ways.

Virus

Virus, as one important category of malicious code, can replicate itself and infect other programs by modifying them or their environment. A virus normally attach itself to a program which still can perform its normal functions.

virus detection

There are many ways to detect a virus, including checking the changes of file characteristics such as size or checksum, expert system analysis, monitoring system activity for suspicious behaviors, and scanning the signatures of viruses.

Checking the changes of file characteristics can not detect all the viruses since some viruses can hide themselves in a very clever way such that no suspicious changes of an infected file can be noticed. Expert system analysis or system monitoring usually bring a heavy overhead, which may not be suitable to some systems with low capability such as a small memory size.

Virus signature scanning is easy and effective. However, only the viruses known to the scanner can be detected. The number of viruses has been always growing and there is no reason to believe this trend won't continue. Particularly as more and more computers are connected into the Internet, new viruses can be propagated through the network very quickly, which could be earlier before a virus scanner knows their signature.

Purpose of Revere

Revere aims to provide a virus signature dissemination solution over the Internet. So, a machine connected to the Internet can know in the earliest time the signature of a newly detected virus if they join Revere, and thus be able to detect the new virus when the machine is infected by that virus.

Currently after a new virus is detected, what normally is done is to update the anti-virus software so that the new virus can be detected. This involves manual intervention and may cause mistake. Furthermore if a user or a system administrator can not be kept updated in a prompt manner, the update of the software can easily fall behind. The automatic notification mechanism provided by Revere addresses these problems.

CHALLENGES

There are some challenges that a successful Revere solution must face:

1) While used for the protection against the viruses, the Revere working mechanism itself must be robust and secure. Revere itself can be an extremely attractive target of security attack because of its purpose. If the security of Revere is broken, the machines intended to be protected by Revere will lose all of the protective powers that Revere provided. Worse than that, Revere is a system that can alter the behavior of large number of machines connected to the Internet if Revere is widely deployed, so a broken Revere may be utilized by hostile forces to break into large number of machines.

2) The dissemination of the virus signature must be very fast, adaptive, and resilient to the failure or temporary disconnection of some Revere nodes, caused by either physical or security reasons. The dissemination of virus information by Revere has to compete with the propagation of viruses. Some Revere nodes may be disconnected for some time. Some Revere nodes may only have limited communication capability. Some Revere nodes may already be corrupted and even try to corrupt other machines and break the dissemination of virus information. Moreover, the whole Revere system is dynamic. Some new machines may join and some existing machines may leave, and some may even crash or be corrupted. Revere needs to adapt itself to all these situations quickly through the distributed cooperation among all elements.

3) There are currently(1998) tens of millions of machines connected into Internet, and each machine is a potential node to run Revere. The scale of Internet is still growing. Thus for a widely used Revere, it's very hard for a single machine or even dozens of machines to store all the needed information, such as per-node or per-domain data structures. Even if this is feasible by having some extremely powerful machines, to keep the stored information up-to-date is daunting. This makes a centralized management of Revere impossible. Moreover, running over a huge number of

machines, the security of Revere has to be still very strong, and the performance has to be still acceptable with low overhead.

4) A good Revere solution running over heterogeneous platforms of different hardware, operating system and communication protocols ought to be easily portable and interoperable.

5) While facing so many challenges, Revere needs to be low overhead and light-weight while still solving these problems, because Revere is used for abnormal behavior prevention, and under normal conditions, it is purely overhead which a user only would be happy to pay as a security cost.

Chapter 2

Goals

Let's call the system composed of all Revere nodes and links between them ***Revere-bone***, or ***RBone***. There are two kinds of messages communicated within the RBone: data messages and control messages. A data message is the virus signature originated from a virus signature dissemination center. And the control message, which can originate from anywhere, is used for RBone management.

The design and implementation of Revere should meet the following goals, that is, security, fast and robust dissemination, scalability, heterogeneity, low overhead, and marketability.

SECURITY

There are two aspects of security in designing Revere: Security of networked machines, and security of Revere itself.

The security of networked machines is exactly the purpose of Revere. Revere provides the signatures of newly detected viruses, so that these machines can be protected against newly emerging viruses. It's this goal that leads to the following goals such as scalability, heterogeneity, fast and robust dissemination, and so on.

The security of Revere itself is of vital importance because of its above purpose, which necessitates the following capabilities:

- Each Revere node must be able to tell whether or not a received message containing a newly detected virus signature is genuine.
- Verifying the authenticity of an incoming control message is harder than that of a data message containing virus information, because a control message can from any point. It would be good if the verification of a control message is doable. Otherwise, Revere needs to minimize the impact of malicious control messages and not let them cause serious corruption.
- Revere must have strong mechanism to protect the virus signature dissemination centers. And Revere must still be robust if a virus dissemination center happens to be corrupted. A corrupted center may try to flood fake warnings of some viruses or even try to cause Revere nodes to do something harmful.
- Each Revere node should have intrusion detection capability
- Revere is an open system. Any machine can choose to join RBone if it wants. This can certainly bring a malicious machine into RBone. This machine may

normally behave exactly like a regular Revere node, but it may also try to corrupt Revere, such as flooding the RBone with replayed Revere messages, blocking the transmission of a message containing a newly detected virus, reporting wrong information about the topology to other Revere nodes, trapping some Revere nodes to hear fake messages from itself, etc. Revere must be ready for these threats.

- Dissemination throughout the whole Revere system needs to be smooth. This will be addressed in "fast and robust dissemination".

FAST AND ROBUST DISSEMINATION

Data and control messages have somewhat different dissemination requirements.

Transmission of virus signature data

Transmission of virus signature messages must be *secure, fast, resilient* and *adaptive*.

The **security** of a data message is not the secrecy of the contained information, because by allowing any machine to be eligible to join RBone, we already imply that any machine has the right to take a look at a data message. However, nobody is allowed to modify the contained virus information in a data message, block the transmission of a data message, or replay previous data messages maliciously. Thus, if a data message is corrupted or blocked, a Revere node must be able to detect the problems. < This paragraph overlaps with the *security* goal section >

The **fast** dissemination of a data message is important because virus signature is urgent information. The sooner that a machine knows the signature of a new virus, the less possibility that the machine may be infected by the virus.

The **resiliency** of a data message is necessary because (1) the Internet and RBone are both *dynamic*, for instance, a Revere node which is responsible for forwarding a Revere message may quit from RBone or crash; (2) Both the Internet and RBone are also maybe *hostile* environment, such that some machines may be corrupted. If a Revere node detects one of its incoming path is corrupted, it should be able to find another incoming path for data messages.

This multipath capability can be implemented in two ways: (1) Once a Revere node detects that a data message is corrupted or blocked, it immediately reacts to this by relying on another available Revere node to be his new designated node to send him a genuine version of the message; or (2) the virus signature dissemination mechanism is essentially *n*-way, that is, each node has more than one fan-ins and can potentially receive more than one copy of the same message declaring a new virus signature from different incoming paths, and only when all the incoming paths are broken can the Revere node be prevented from receiving genuine information of virus signature.

The **adaptivity** of data message dissemination is also necessary.

Revere needs to be able to adjust the RBone topology for dissemination purpose. For instance, if a new Revere node joins RBone, or, an existing Revere node quits from

RBone, or a Revere node is believed to be corrupted or crashed, or a Revere node has stronger routing capability of Revere messages because of its geographical location or its platform and configuration, Revere should be able to tune the system to achieve best possible performance in terms of doing dissemination.

An optional goal for the data message dissemination is the ability of **monitoring the progress** of dissemination. The feedback information needs to be collected for this purpose so that the center can know the percentage of Revere node covered in a dissemination session, monitor the status of a Revere node, and adapt its dissemination strategies.

This problem is hard, because the large scale of RBone makes it impossible for any one single machine (or dozens of machines) to know the overall knowledge of the whole RBone. Moreover, in RBone only centers have authentication information available so that messages declared from them can be verified. No machine can have all the authentication information for all the regular Revere nodes thus can not verify all the feedback messages.

Transmission of control information

For transmission of Revere control information, which can be out-of-band or piggybacked in an online data message, it must be secure, resilient and adaptive too.

However, unlike virus signature messages which are all originally from virus signature centers, the initiator of a control information transmitted over RBone can be any Revere node in the RBone. How to prevent the replay of control information by malicious node is an important security issue of the Revere message transmission mechanism.

The resiliency and adaptivity of control message may have different mechanism from those of data message.

Moreover, depending on the purpose of different control messages, the speed requirement of some out-of-band control message may not have to be as fast as that of data message. For instance, there is normally no strict timing requirement on a “keep alive” message to a Revere node from some other nodes on RBone.

The control message needs to be light-weight, too.

SCALABILITY

Each machine on Internet is a potential node to run Revere. With the increasing of the scale of the Internet, the scalability of Revere can only become more and more significant.

Scalability in terms of information storage

The large number of machines on the Internet implies that there shall not be a centralized management at all, since it’s impossible for a single machine to keep track of changes of concerning information in order to do a central management. It’s difficult to fulfill the work based on stale data, even with very complex algorithms.

Thus, each Revere node only stores and manages incomplete knowledge of the whole Revere world. Based on the partial information, each Revere node should be able to cooperate with other Revere nodes to do the virus signature reception and dissemination in an both effective and secure way.

So does the virus signature dissemination centers. Each center needs to be able to disseminate virus signatures into RBone without the whole knowledge of Revere world.

Scalability in terms of performance

The number of machines on Internet running Revere is expected to be small at the beginning phase. But we also expect when more and more machines join and run Revere, the performance of the existing Revere nodes won't be severely degraded. Conversely, as more machines start to run Revere, the dissemination speed to one machine from another might be improved instead, if a better path can be formed by including some newly joined machines on the path.

It's possible that as the number of Revere nodes grows, the percentage of nodes having received a new virus signature within a certain fixed period is less than before. But the design of Revere should minimize this reduction.

Scalability for security and robustness consideration

This distributed nature of Revere is also necessary in terms of security and robustness. A centralized management of all Revere information is much more susceptible to security attack than a distributed scheme. A central site for Revere management can easily become a bottleneck and a single point of failure and any kinds of failure of this central site is very disastrous.

Other considerations

HETEROGENEITY

RBone includes all kinds of potential platforms in terms of hardware, operating system, or communication medium and protocols.

While we design Revere to run on a wide variety of hardware and operating systems, a Revere version running on Windows platform is essential, since this platform is the most popular platform and most viruses have been found in this platform.

However, fewer viruses in other platforms also imply much less caution against viruses in those user communities. So, Revere shouldn't limit its design within the scope of Windows platform.

Furthermore, different machines may have different security capabilities. Some machine may be protected by a firewall or an anti-virus software, some machines may have

security library installed to facilitate a security implementation, while some may have nothing. Different machine may also have different security impacts if corruption or failure occurs -- the impact of a broken firewall is different from that of an ordinary PC.

Different machine may also have different communication configuration and capabilities. A machine connected to Internet through a Modem has much less communication capability than a machine on a Ethernet, which has much less communication capability than that of an machine on Internet backbone.

Thus, Revere needs to exploit the features of its platform and configuration. For example, the authentication of Revere messages on a firewall machine can be configured to be stronger than that of a leaf machine (a machine doesn't forward Revere data message) ; or, a machine with broadcast or multicast capability may use different mechanism to send out control message or forward data messages than a machine only having a point to point connection.

LOW-OVERHEAD

Revere is meant to provide protection against extraordinary events. In normal operation, Revere will be pure overhead for most Revere nodes. (Overhead they will be glad to have paid when disaster strikes, admittedly.) Thus, in ordinary circumstances Revere must go about its business so efficiently that its users do not notice its operations having any impact on their daily activities. If Revere fails too badly in this respect, many users will disable or otherwise limit Revere activities.

In many cases, some hard problems Revere must solve can be simplified merely by spending more cycles computing them. For example, substantial resources can be devoted to intrusion detection and key distribution. The success of Revere depends on deciding wisely and carefully when to use extra system resources to solve a problem. Generally, we must design all Revere algorithm to work as efficiently as they possibly can, and must seek to make use of otherwise wasted resources on Revere nodes, such as CPU cycles during idle periods.

MARKETABILITY

Revere aims to become a highly marketable product, although a prototype has to be implemented first. For this purpose, besides achieving the above goals, Revere must also have following characteristics:

- Easy to use. Revere must be easy to use. For example, try to save the number of command options, use a graphical interface for configuration and running, simple to register a machine or restart the Revere functionality, etc;
- Bug-free;
- Support major existing commercial virus protection systems, and be able to expand the support to future systems.
- Logging. A logging capability should be provided for monitoring and analysis.

There are also some political and industrial factors.

In the following chapters, the design of security and the design of dissemination of Revere are discussed.

Chapter

3

RBone

RBONE PROPERTIES IN TERMS OF DISSEMINATION

The main purpose of RBone is to disseminate virus information to all nodes on RBone. In terms of the dissemination of Revere messages, two properties must be provided over RBone: *multipath* and *adaptivity*.

Why not single path?

If there is only one single path from a virus dissemination center to a Revere node, a malicious node which "happens" to be on this path can make the data transmission insecure. Particularly, it may modify a Revere data message, replay a previous Revere message, drop some or all Revere data messages en route, or route the messages in a wrong way. The result can be either the Revere node receives corrupted messages, or the Revere node is isolated from hearing Revere messages about some new viruses.

By checking the signature of a received message about virus and/or other information, a Revere node can tell whether the message is corrupted or not (such as a replay), and then request a retransmission. However, since there is only one single path, the retransmission can not be guaranteed to succeed next time.

If a Revere node can not hear genuine Revere data messages about viruses, the node can not get Revere service, which is exactly what Revere wants to do.

So, a single-path dissemination mechanism over RBone is subject to a denial of service attack. Therefore, Revere needs to be multipath when doing message dissemination over RBone.

A multipath mechanism can provide more than one incoming path to a Revere node, so that if one path is corrupted, the node can resort to another path.

Why adaptivity?

The dissemination of Revere messages needs to be adaptive, because

- In general, any message routing framework should be adaptive to the changing of network physical conditions, such as using another route because of congestion or disconnection, shrinking the data window at sender side to do flow control, and so on.

- Revere has its own specialties in terms of security. A previously uncorrupted node may be detected corrupted later, and in this case Revere needs to

disseminate message using a different path. Or, with Revere running, if some feedback about the dissemination is able to be collected, the dissemination for later sessions should be improved by adapting to a better strategy in doing dissemination.

On the other hand, while some information sources may respond to congestion or some other situations by tuning its parameters related to sending, Revere may not want to do so, because its information is light-weight and important. So the adaptivity of Revere is different from that of normal data transmission.

Bandwidth aware but not frugal

Revere is a system aiming at providing important information about viruses. Moreover, these emergent information are mostly light-weighted. A data message containing a virus signature and related information is normally expected to be less than 1 KB.

They are also not disseminated frequently, and it is only when a new virus is found that a dissemination session will be started by a center to notify Revere nodes the virus information.

This also makes multipath dissemination of viruses fairly acceptable. Though multipath dissemination seems to use more bandwidth mainly as a security cost (also improving reliability and speed), the light-weight of the messages does not make this cost a burden.

Latency important but okay at human level

After a virus is detected, and probably the remedy is also already found, a Revere session will be started to disseminate the virus information over RBone to Revere nodes. Some nodes which missed the information may pull the information of this session later. Since the virus is emergent information, fast dissemination is very important.

At the same time, the virus propagates at the level of human time, such as several minutes. A virus is normally activated by specific user behaviors. The detection of a new virus and the determination of its remedy also take time at human level. So, while it's certainly great to be able to disseminate the virus information as quickly as possible, Revere can still be successful if it can disseminate the information quickly enough at human time level.

Latency metrics

The latency metrics for a Revere session can be the time for a data message to reach all Revere nodes. However, considering that at any time there maybe some disconnected nodes, the time to reach 100% Revere nodes may be impossible. The metrics can also be the time to reach certain amount or some fraction of Revere nodes. To be more realtime, the metrics can also be the percentage of covered Revere nodes by the Revere message of a specific session.

HOW SHOULD RBONE WORK?

Session

Each Revere session is defined as an activity started by a Revere virus dissemination center to disseminate information about one (or several in a compound way) specific virus over RBone to those Revere nodes. The information about virus is contained in a signed data message, which includes virus signature and also maybe virus remedy.

Each session has a session ID. A session ID consists of the identification of the virus dissemination center which starts the session, and a sequence number. The sequence number is generated by the center. The tuple found by < virus dissemination center identification, sequence no > is unique.

RBone organization

Before Revere tries to organize Revere nodes in some fashion, one fact to recognize is that all Revere nodes already has some organizational properties. For example, a Revere node may already be located at a LAN, either protected by a firewall or not. Or some Revere node might be multicasting capable and they can cluster themselves together as a group unit.

No matter how RBone is organized, it needs to achieve the goals discussed in chapter 2.

< More discussion needed here>

RBone Autonomy

Some administrative domain may enforce their own Revere policies. For example, they want to double check a report of a new virus signature and probably sign the messages containing the virus information before the machines in the domain can hear the messages or act upon the received information. A firewall can be this kind of place enforcing local Revere policies.

Multipath

To implement a multipath dissemination mechanism for Revere, we need to firstly define what kind of multipath dissemination that Revere wants to have.

A multipath mechanism can have two different styles. One is to have many paths at the same time. By doing so, a Revere node can receive more than one copies of a data message of the same session. Another alternative of multipath is to let each Revere node be able to dynamically locate a new path only if an existing path is detected corrupted. To be more accurate, this latter method is potentially multipath, but still single path if taking a snapshot.

Choke point

Because of the randomness of Internet physical topology, given a virus dissemination center and a Revere node, there can be some choke points that any message from the

center to the node must pass through. Sometimes, no matter wherever a center is located, any messages from outside may have to pass through a choke point.

If a LAN only has one router, this router is a choke point for any nodes in the LAN as long as the message is from outside. For example, a firewall is a choke point to all nodes protected by it. A choke point can also be far away from a Revere node. For instance, if all Revere messages originated from a center have to pass another machine, typically a router, this machine is a choke point of any Revere messages initiated by this center.

These choke points are known ones. There can also be unknown choke points, which can be temporary for routing reasons. If special handling is needed, the detection of these unknown points may need to be done first.

The above choke points are unavoidable because of the physical topology. We can call them *hard* choke point. Sometimes a *soft* choke point may be formed because of the multipath algorithm of Revere which results in two or more paths sharing a single point, i.e. a soft choke point. The design of Revere should try to avoid the formation of soft choke points.

Choke point is of vital importance in terms of security. Even if there is more than one path to reach a Revere node from a center, if they share a common choke point, the choke point becomes a single point of failure, and thus an obvious target for malicious activity: if a choke point is broken, then every channel to a Revere node is broken.

A choke point can either be a Revere node or not. Regardless, a corrupted choke point should not cause serious impact on the rest of the world, except the domain downstream.

IP multicasting as a building block?

IP Multicasting is becoming more and more popular and many commercial router products will incorporate multicasting functionality. There are many reasons to believe that multicasting capability is universal in tomorrow's Internet, which may be evolved from today's MBone.

IP multicasting can cause a message to reach many receivers from a sender just once, instead of "unicasting" them one by one. This can save both time and bandwidth. Revere has similar requests in terms of disseminating virus information from a center to many Revere nodes, particularly when pushing. The problem is that IP multicasting works by having a sender-based tree or a shared tree, over which the messages are routed. So, there is only one single path from a sender to a receiver. To implement a multipath mechanism for Revere, simply multicasting is not satisfactory. Another problem of multicasting is that the multicasting functionality is still not universal over Internet, although multicast deployment has been expanding, such as MBone.

Reliable multicasting focuses on the reliability of multicasting in terms of message loss or error. It still can not provide security needed by Revere such as the resiliency by multipath if the reliable multicasting is still single path.

Also, no matter whether multicasting is used or not, a disconnected machine needs to pull missed messages anyway.

Active network support for Revere

Because of the special routing requests of Revere messages, and the existing Internet routing facility can not satisfy these requests, active network may be a good option. If Revere can tell a router or a machine(which is active network capable) how to route a Revere message specifically for Revere purpose, such as multipath requirement, a Revere machine can be liberated partly or completely from doing Revere messages routing themselves.

The active element in a packet, normally a piece of code, can either be executable or be a pointer to a routine resided in an active node. An active node can either be a Revere node to receive Revere service, or just a router being able to following Revere instructions embedded in a Revere message. When a Revere center disseminates some information about a virus which even has some particular dissemination requests, the center can similarly *code* the packets containing the virus information to satisfy the specific requests.

On the other hand, Revere also has the potential to protect active networks from being infected by viruses embedded in an active packet.

Achieving optimal performance

In doing the routing of Revere messages to Revere nodes, no matter by what mechanisms, the following points must be met:

- Loop prevention.

  Path vector may be used.

- No traverse over the same link too many times when doing pushing in one session or pulling a Revere message.

Reliability

A Revere node or some RBone link may fail during a session. Such a failure should be limited to its local space and not crash the whole system.

Worse than that, a virus dissemination center may also fail or be detached from the network.

Chapter

4

Security of Revere

DATA MESSAGES

A data message is initiated by virus dissemination center. It contains virus information and may be also accompanied by information on how to deal with the virus.

Security of data messages

1) Revere is an open system. It does not exclude any machine on Internet from viewing the data messages which contains virus information. Conversely, it aims to let every machine on Internet be prevented from virus attack by notifying them virus signatures. So, each machine is allowed to view Revere data messages, unless forbidden by local administrative policies.

   So, since Revere is open to every machine to join and any joined machine has the right to view the data messages, there is no secrecy of the virus information contained in data messages.

2) However, data messages must contain authentic virus information. Although each machine can view the messages, any machine can not modify the virus information in a data message in a manner undetectable by others.

3) Each Revere data message can not be a replay of a previous data message, except when multiple copies are received when a Revere node has more than one incoming paths for Revere messages, or, when a Revere node requests some missed virus information through a "pulling" session.

4) Furthermore, data messages can only be created by truly virus dissemination centers. Any other machines shall not be able to counterfeit fake data messages.

5) The data messages can also not be blocked by any malicious node. Blocking data messages transmission can cause denial of service.

Ensure the security of data messages

Corresponding to the above points of security requests for a Revere data message,

1) *No encryption of virus information in a data message is necessary for secrecy/privacy purposes.*

2) *Modification prevention*. Revere uses public key cryptography. Each data message must be signed by the virus dissemination center which initiates this data message.

Given that the public key of the center is already known by all Revere nodes, each Revere node can verify the signature of the data message to know the authenticity, so that a modification of data message can be detected.

The birthday attack prevention is an issue here.

3) *Replay prevention.* (This subsection needs more work.)

In a point to point communication, the replay is normally prevented by including a random number or a sequence number in a message. But the exchange of such a number for a Revere session can be very different from that of a point to point communication. Furthermore, in a Revere session, a replayed message, legal or not, is somehow helpful in transmitting more copies as long as the virus information is still authentic, such as when receiving multiple copies of virus signature from just one session using multipath technology. What we want to prevent is the flooding by the replayed Revere message.

** PULLING*

In a point to point communication when a receiver wants to **pull** a message from a sender, a random number can be firstly sent from the receiver to the sender, and then the sender can include the random number in the message, so that the receiver can check whether a received message is a replay or not by comparing the random number.

If a Revere node wants to pull a message directly from a center, this strategy may be used. But, a pulled data message is essentially also a repetition of a previous data message in most times, although it's a "*legal replay*". Then for those machines on the way from the center to the requesting Revere node, they have to decide whether a received replay message is legal (a pulled message) or illegal. Without the capability, RBone may be flooded by illegal replay messages.

A pulled message may be sent from a center, a neighboring node, or some other Revere nodes. Differentiating a pulled message from a replay is a challenging issue.

** PUSHING*

In a point to point communication when a sender **initiates** (**pushes**) a message which the receiver does not know at all before arrival, in order to protect against the replay attack, the sender can add a sequence number to each message so that each time a receiver can only receive a differently numbered message. Or, a receiver can deposit a random number in the sender during a session, and the sender includes this random number in the following session.

During each Revere session of communication when a center pushes a data message to huge number of Revere nodes, each Revere node needs to check whether a received data message is a replay or not.

If a random number is used to deal with replay attack, a random number associated with each dissemination session needs to be generated. Whoever generates a random number, whoever needs to disseminate the random number to all other Revere nodes and the center (if the center is not the random number generator). This reduces to

another dissemination problem (but of a random number, not a virus signature) – which means we haven't solved the problem!

If assigning each data message a sequence number, and each Revere node keeps a record of the most recent sequence number, then, upon receiving a data message, the node can check the sequence number of the message (given the signature has been verified). (It's possible that the sequence number of a received data message is not continuous, for example, when a data message of a following session arrives before that of a current session.)

It's also possible that a received data message with correct sequence number is a replayed message. But this does not matter as long as the signature can be verified. In this situation, such a replay message can only help our Revere dissemination session, although the later true message is judged as a repetition and discarded.

4) *Counterfeit center prevention.* After knowing the public key of a virus dissemination center, the origination verification of the data message becomes easy, since each message is signed by the center itself.

   The question here is that how to distribute and revoke the public keys of a virus dissemination center, which can happen

   - When a new center is formed;
   - When a center wants to retire;
   - When the public key of an existing center needs to be changed;

   This will be discussed at the security of the virus dissemination center in detail.

5) *Dissemination blocking prevention.* As long as a malicious node wants, it can block any message which is routed by this malicious node. So, if each Revere node is also a router on RBone for Revere data messages, nothing can be done for a specific malicious node technically (maybe administratively). For this reason, there must be an ingenious design of Revere dissemination mechanism.

CONTROL MESSAGES

Control messages, either online or offline, is initiated by any node on RBone, which is used for the management of RBone. Some examples of control messages include joining request, quitting notification, data pulling request, etc.

Security of control messages

The difficulty of the security of the control messages is that the messages can be from anywhere. There is no easy way to tell whether a control message is authentic or not. We may achieve the security of Revere by

- Limiting the transmission scope of a control message;

- Limiting the impact of receiving one more control message on the existing dissemination structure.

- Adding some local security policies if possible.

Essentially the security of control message depends on what a dissemination solution will come out.

PROTECTION AND INTRUSION DETECTION IN A REVERE NODE

Each Revere node, while having Revere service, may independently have other facilities to protect itself from many kinds potential attacks. Particularly it may have an intrusion detection facility installed already before Revere is installed. However, Revere can also have its own intrusion detection mechanism so that each Revere node is resilient to attacks, particularly those Revere specific.

The Revere intrusion detection facility running on a Revere node will work by monitoring Revere activities of the Revere node itself and its neighbors, the inbound and outbound traffic, and some other behavior patterns. Intrusion detection can help prevent and report the flooding by illegal control messages (those messages such as joining Revere or a pull request) and help to pinpoint which machine is causing trouble.

The multipath mechanism of Revere also provides a good chance for doing intrusion detection. A Revere node having more than one incoming paths expects to receive more than one copies of a Revere message of a session. For instance, if the number of received copies is less than expected, Revere can know that something is wrong with those incoming paths from which no Revere messages can get in. Certainly we need to make sure here that the messages reporting problems are authentic or at least in some degree they are authentic.

PROTECTION AND INTRUSION DETECTION IN A VIRUS DISSEMINATION CENTER

Compromise of a virus dissemination center is very disastrous. If a center is broken, or some machine can masquerade as a center machine, all kinds of trouble can be resulted. The compromised center can disseminate some malicious updates, flood networks easily, cause Revere nodes to respond to non-existing viruses such as turn itself off because of some detected “serious” problems, and so on.

Protection

Public key of centers are known. And the private key, on the other hand, is of vital importance and must be kept extremely secret.

Multiple-signatures of a Revere data message may be needed, depending on the impact if a Revere node responds according to the message. If the probability that a center is corrupted is a, the probability that all m centers are corrupted is $a \; exp \; m$, which is much

less then a , if these m centers are independent to each other and have same probability (i.e. a) of being corrupted.

Intrusion detection

An intrusion detection mechanism can be independently applied to a Revere virus dissemination center. At the same time, a center can be specifically tailored to the needs of disseminating Revere messages about viruses, which measn the behavior of the center can be of some particular pattern if it works correctly. Thus, observing the behavior of the center carefully can also provide intrusion detection

Revocation

If a virus dissemination center is found corrupted, a revoking mechanism may need to be started to revoke the messages just sent out.

Whether the revocation can succeed or not depends on how a Revere node responds to received messages about viruses, particularly the speed.

KEY MANAGEMENT

While the number of virus dissemination centers is much less than that of all Revere nodes, the public key of these centers should be distributed to each Revere node in some special way.

A public key can be expired or compromised. If so, a new public key needs to be redistributed.

In case of key compromise, the revocation of the compromised public key is an issue. It's not easy to make a revocation message faster than that of a Revere message with a compromised key.

Multiple signature may be used to overcome this problem. So only when all signatures are not authentic is the message corrupted. But this may slow down the dissemination speed since each message has to be signed more than once independently.

< More to be continued >