A Light-weighted Source Address Validation Method in IPv4/IPv6 Translation

Yu Zhu, Jun Bi and Yayuan Sun
Network Research Center, Tsinghua University,
Department of Computer Science, Tsinghua University,
Tsinghua National Laboratory for Information Science and Technology (TNList),
Beijing, China
zhuyu05@mails.tsinghua.edu.cn, junbi@tsinghua.edu.cn, sunyayuan@cernet.edu.cn

***Abstract*— since global IPv4 address has already exhausted in 2011, IPv6 is going to be deployed more widely in the next years. Both IPv4 and IPv6 would coexist in Internet for many years. Some transition technologies can help IPv4 to work with IPv6, but most of them are vulnerable to IP address spoofing attack. This paper proposes a source address validation method which works with IPv4/IPv6 translation. Only one change is required in DNS translation, based on current translation technology. Currently, an IPv4 server's address in DNS reply would be translated to an IPv4-mapped IPv6 address by DNS translator. In this paper, we proposed a method called "gateway identify code" (GIC) that the translator gateway embeds authentication information in IPv4-mapped IPv6 address in translated DNS reply. A host who receives this DNS reply would use this GIC embedded address to start communication. When packets reach translator gateway, validation is performed to check whether the GIC is correct. This technology can work with both stateful translation method and stateless translation method, including NAT-PT, NAT64 and IVI. This method will protect the address pool and filter the IP address spoofing attack.**

Keywords- IPv4/IPv6 translation; Anti-spoofing; Source address validation; Packet filtering; Internet Security; Access control.

I. INTRODUCTION

A. IPv4 address exhaustion and IPv6 development

In Feb 2011, ICANN (The Internet Corporation for Assigned Names and Numbers) declared that it had handed over the last IPv4 blocks to RIRs (Regional Internet Register). This handoff means the global IPv4 Internet has run out of addresses. There are not many addresses in RIRs address pool too. In April 2011, APNIC (Asia-Pacific Network Information Centre) reached the last /8 block of IPv4 address [1]. Asia Pacific is going to be the first region unable to meet the IPv4 Address demand due to the unprecedented fixed and mobile network growth in this area. After the exhaustion of IPv4 addresses, there would be no new address for new device such as cellphones, wireless sensors to visit Internet. IPv4 address exhaustion may cause the growth of Internet to speed down.

Noticing IPv4 cannot provide enough addresses to meet the requirement of the development of Internet, IETF proposed IPv6 in 1990. IPv6 has 128bit address space which could provide far more address than current IPv4 network. In October 2010, 243 (83%) of the 294 top-level domains in the Internet supported IPv6 to access their domain name servers, and 203 (69%) zones contained IPv6 glue records, and approximately 1.4 million domains (1%) had IPv6 address records in their zones [2]. However, IPv6 address family is not compatible with IPv4. An IPv6 host cannot directly connect to an IPv4 host. Moving billions of users and millions of sites from IPv4 to IPv6 will cost many years. A long period is needed for IPv4 users to transit to IPv6. Two categories of transition methods have been proposed to provide a way to connect IPv4 and IPv6 network, namely, translation and tunnel. Both of them are important scenarios in IPv6 deployment.

B. IP spoofing attack in IPv6/IPv4 translation

IP address spoofing attack uses forged source address in packets. Since the address is forged, the reply of the attack packet will not reach the attacker. This kind of attack is widely used in a DoS (Denial of Service) attack [3]. Although many ISPs have enforced spoofing prevention functions on their network infrastructures over these years, no mitigation improvement has been achieved over four years [4]. The MIT ANA Spoofer project [5] shows that 17.2% of the IP addresses, 15.2% of the net blocks and 24.4% of the ASes are still spoofable across Internet.

Many kinds of anti-spoofing methods have been proposed by researchers, including SAVI [6], uRPF [7], Packet Passports [8], DPF [9], SPM [10], etc. SAVI switches sniffer packets of IP address allocation protocol, uses information in the packet to create binding entry on the switch. However, it needs to replace all access switches change access switch to enable SAVI function. uRPF reversely utilizes the forwarding table on routers to check packets' source addresses, but may drop valid packets in case of asymmetric routing. Packet Passports inserts "AS passports" which are checked by the ASes along the packets' paths into packets. It fails when packets are delivered through different AS paths due to routing dynamics. DPF is a distributed route-based packet filtering framework. With partial deployment, DPF can decrease spoofing packets significantly. The major omission from the DPF research is the method for routers to learn the incoming direction information, which is very critical and hard in practice. SPM is an AS-level source address validation system. A unique temporal key is associated with each ordered pair of source destination ASes in SPM. Edge routers will add the key to outbound packets and verify the key in the inbound packets.

 ISBN: 978-1-61208-201-1

All of these anti-spoofing methods are not designed for the IPv6/IPv4 transition scenario. So a new method is needed to apply anti-spoofing in IPv6/IPv4 transition.

Address spoofing attack is a tremendous threat to stateful IPv6/IPv4 translation method. In stateful translation method, translator allocates temporary IPv4 address from address pool for IPv6 hosts. The address pool is an important resource of translator gateway. Attacker can exhaust the address pool by forged address attack. Each packet with a different source address can obtain an address from the address pool. One reason that NAT-PT [11] is moved to history status is "Creation of a DoS (Denial of Service) threat relating to exhaustion of memory and address/port pool resources on the translator."[12].

This paper proposes a novel anti-spoofing method "Gateway embed and verify" (GEAV) method to prevent IP spoofing attack in IPv6/IPv4 translation. DNS translator will translate an IPv4 address to an IPv6 address in DNS records. In GEAV, the translator gateway sends "gateway identify code" information to hosts in the IPv4-mapped address in DNS reply packet. This special IPv4-mapped address will be carried by every packet a host sends out which needs translation. The translator gateway checks GIC in packets to apply anti-spoofing when packets reach. GEAV does not need any host change. The detail of this anti-spoofing method will be discussed in next chapters.

II. GATEWAY EMBED AND VERIFY ANTI-SPOOFING METHOD IN IPV6/IPV4 TRANSLATION

A. *IPv4/IPv6 translation technologies*

Transition technologies could be mainly classified in two classes: tunnel and translation. Tunnel enables IPv6 (IPv4) communications to pass through IPv4 (IPv6) network, while translation enables communication between IPv6 and IPv4 hosts, as Figure 1 shows. Both technologies are used widely in IPv4/IPv6 transition.

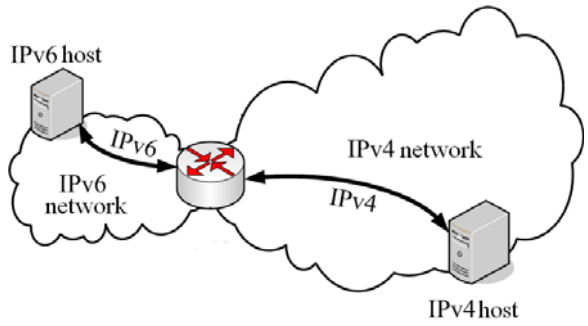

Figure 1. IPv6 / IPv4 translation.

Classified by working layers, translation technologies are sorted into three categories: application layer, transport layer and network layer translation. Network layer translation methods translate IP packets between different address families and typical solutions are NAT-PT, NAT64 [13] and IVI [14].

NAT-PT assigns a temporary IPv4 address for the IPv6 client and translates each packet on the translator gateway according to the mapping table of IPv6 client address and IPv4 temporary address. NAT64 conquers NAT-PT's flaw, separates the translate DNS with the gateway. IVI uses special IP address allocation strategy for hosts. As a stateless method, IVI do not save the mapping information of IPv4 and IPv6 addresses. IVI calculates IPv4 address/port for a host using its IPv6 address, replacing the mapping table with a global mapping rule.

Let us take NAT-PT as an example to see how IPv4/IPv6 translation works. Step1, an IPv6 host (client) wants to use the service provided by an IPv4 server, so it sends a DNS request to look for the server's address. Step2, An A record, which contains an IPv4 address, would be carried in the DNS reply to response the request. The translator gateway would hijack the DNS reply and transform the A record in the DNS reply into an AAAA record, which contains an IPv4-mapped IPv6 address. Figure 2 shows an A record of 1.2.3.4 is translated to a AAAA record with 2001::[1.2.3.4] by a translate device.

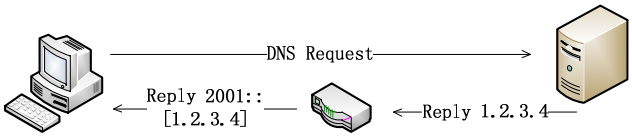

Figure 2. DNS interaction in NAT-PT translation.

Step3, after the DNS interaction, the IPv6 client will start to send packets to the address in AAAA record to acquire service. When the packet reaches the gateway, the translator gateway recognizes the packet's destination address represents an IPv4 host. The gateway allocates an IPv4 address from address pool for client, and translates the packet to IPv4 address family. When the server responses the request, the translator gateway translates server's reply and forward it to the IPv6 client. In Figure 3, the gateway allocates a temporary IPv4 address 4.3.2.1 for the IPv6 host 2001:da8:10::4. As Figure 3 shows, in the view point of the client, it communicate with host 2001:[1.2.3.4]. In the view point of the server, it has a IPv4 session with host 4.3.2.1 .

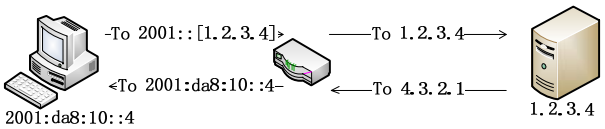

Figure 3. IPv6 client visit IPv4 server using translation.

In all these steps, the packets pass through translator gateway. The communication between the clients and translator gateway looks like a three-way handshake. In step2, if the translator gateway embeds some information in the DNS reply, this information would only be acknowledged by the owner of the source address of DNS request and the gateway. When there is an attack using spoofed address, if the owner of the source address of the

 ISBN: 978-1-61208-201-1

DNS request did not send the request, the reply would be discarded by the receiver. Unless the attacker uses Man-in-the-middle attack, he cannot receive the information embedded by the translator. The translator gateway embeds GIC (Gateway Identify Code) information to the DNS reply, and verify it when receive packets from hosts as shown in Figure 4. This method is called “Gateway embed and verify” method.

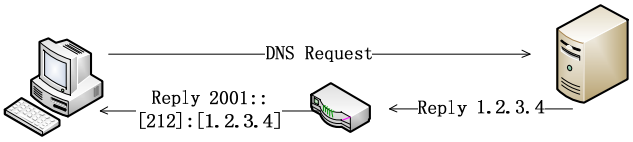

Figure 4. GIC embedded DNS reply.

To deal with man-in-the-middle attack, a proper method is encrypting all the packets which cost extra resources. This is not a light-weighted method that we are looking for.

B. Gateway embed and verify method

Without any DNS extension, the only information stored in DNS record andused by the sender is the address. In IPv4/ IPv6 translation, the IPv4 address after translation would be an IPv6 address with the original IPv4 address embedded. In NAT-PT address format, as shown in Figure 5, 64bit will be used for the prefix; 32bits will be used for the original IPv4 address; the rest 32bits are set to 0.

0 63	64	96	127
Local ISP prefix	Zero	IPv4 addr	

Figure 5. Normal IPv4-mapped address format.

These unused bits can be used to store information. An identify code named “gateway identify code” can take some bits. For example, as shown in Figure 6, 8 bits GIC is placed in 88-95bits.

0 63	64	88	96	127
Local ISP prefix	Zero	GIC	IPv4 addr	

Figure 6. Improved IPv4-mapped address format.

When a packet arrives at the translator gateway, the embedded GIC would be checked. Gateway forwards a packet if the GIC is correct, or discards it if it is wrong, as shown in Figure 7. If the GIC in the packet is 212, it would be translated. If not, it will be blocked.

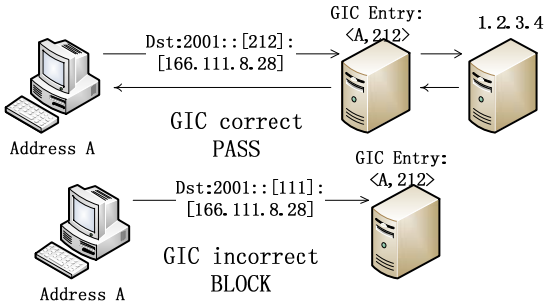

Figure 7. Gateway validation with GIC.

The translator gateway keeps a table for hosts who sent out DNS requests. Gateway maintains the table that stores the IPv6 source addresses of the DNS requests and GICs for verifying. The GIC entry has a very short lifetime. Typically, it is 5 minutes, assuming a host start to send a request to the server address after it receives the address in 5 minutes. When a client start to use the address to connect to an IPv4 server and an allocation of temporary address happens in the address pool, this GIC entry changes into a static entry until the resource is reclaimed.

The improved process of a session using IPv4/IPv6 translation would look like this.

- Step1, IPv6 client sends request to an DNS server.
- Step2, translator gateway receives the A record of IPv4 server.
- Step3, translator gateway tries to find the source address of the DNS request in the GIC table. If this entry could be found, then use the GIC in this entry. If it is not found, then a new entry would be created, add assign a GIC for this host.
- Step4, translator gateway sends the GIC-embedded IPv4-mapped address in AAAA record to the IPv6 client.

The GIC entry will be inactive and be removed when the IPv4 translation address is reclaimed by the address pool.

C. Infomatin sharing issue between gateway and DNS

In some translation method, DNS-ALG [15] will provide translation of DNS replies. In the previous part of this chapter, we assume that it is the gateway device that sends the IPv4-mapped DNS reply to the host. If a DNS reply a host receives was sent out by an independent DNS server, the assumption that GIC information is shared only by the gateway and the host is violated. Two solutions can be used in this scenario.

The first is to launch a communication between DNS and gateway to share information with each other. By sharing information, DNS and gateway could be treated as one device. In the next chapters, we would treat the translate DNS and the translator gateway as one gateway.

The second is to deploy the independent DNS-ALG server out of the translator gateway. That means every message from the DNS to the host would pass the gateway. The DNS sends out IPv4-mapped IPv6 address in DNS reply,

 ISBN: 978-1-61208-201-1

the gateway hijacks the DNS reply and embeds GIC information in it.

III. CORRECTNESS ANALYSIS

A. *Correctness in stateful method*

Now check some scenario in this system when an address spoofing attack happens.

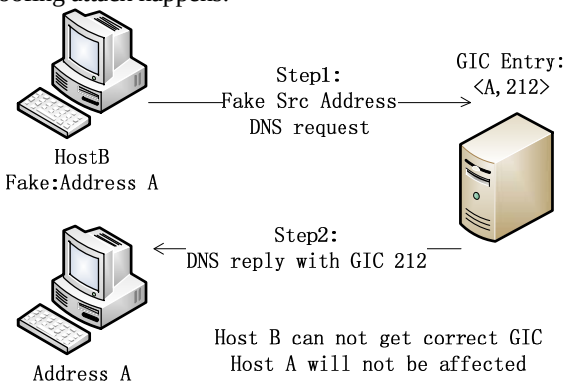

Figure 8. Scenario1

Scenario1, as shown in Figure 8, IPv6 host A keeps a session with an IPv4 server, and has already gotten an IPv4 address from the address pool. Host B tries to fake host A to start an attack. If host B sends a DNS request using host A's address, then the gateway would send to GIC to A but not B. If B skipped the DNS step and try to send a packet to an IPv4-mapped address directly without a correct GIC, this attack packet cannot pass the gateway. If an attacker tries to enumerate GIC, only a small part of attack stream would pass.

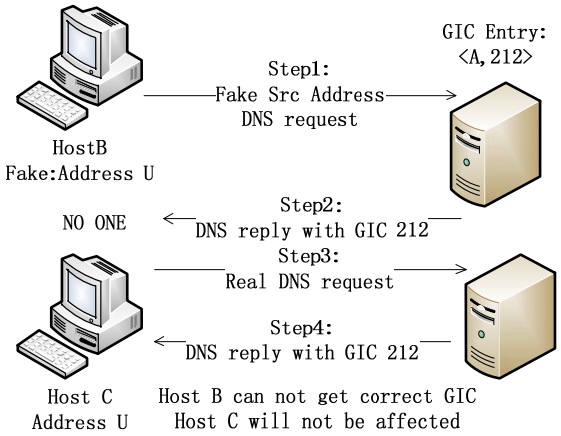

Figure 9. Scenarios 2&3

Scenario2, Host B wants to fake an unused IPv6 address U. B uses U as source address and starts to send packets to an IPv4-mapped address. The translator gateway finds no entry of U in GIC table when receives a packet using U as source address, and discard the packet. When host B sends a DNS request using U, the gateway would add an entry in GIC table, and send the GIC to the address U which cannot be picked by host B. Host B do not know the correct GIC because it cannot receive any packets sent to U unless Man-in-the-middle situation.

Scenario3 is a transition of Scenario2. Host B fakes a host with an unused IPv6 address U. Then the legal owner of U, called host C, start to use U. Host B may have sent some DNS request, so the gateway may have already got an entry of U in GIC table. When C sends DNS request, gateway uses the entry already exists in GIC table and sends the GIC to C. Any attack performed by B would not affect host C to use U to communicate to IPv4 servers. Figure 9 shows the GEAV method could work properly in scenarios 2 and 3.

From the analysis of the above scenarios, it can be concluded that a spoofing address attack will only form an entry in the GIC table. Most attacks would be filtered at the gateway. Only a small number of attacks can get out of the gateway when the attacker happens to guess correctly.

B. *Security issues of GIC table*

In GEAV, before a request gets resource from the address pool, a GIC entry will be set up for verification. If the verification fails, the allocation will not happen. But the GIC table might be a new target of DoS attack. An attacker might forge different addresses to form a lot of GIC entries. When gateway detect the size of GIC table is growing very fast, it can reduce the lifetime of a GIC entry to a very short time, for example 5 seconds. The speed rate of attack stream multiple the lifetime of GIC entry is the amount of forged entries in the GIC table. By reducing the lifetime of GIC entry, the pressure from DoS attacks to the GIC table could be reduced.

C. *Correctness in stateless method*

There is no address spool to protect in stateless method. Though this verifying method is stateful, it can also be used in stateless IPv6/IPv4 translate method to reduce the address spoofing attack. But additional verification cost is required.

The size of the GIC table would grow larger than that in stateful method. In stateful method, the GIC table is as large as the scale of the address pool. In stateless translation, there are no allocation in the address pool and no reclaim of address, so the size of GIC table would reach the size of the address space of the subnet since GIC for each address should be assigned in advance. This will cause terrible scalability problem. We will solve it in the next Chapter.

IV. SCALABILITY ANALYSIS

The protection is done on the translator gateway. The scale of the GIC table could be a problem as gateway needs to store the whole GIC table. Though the GIC entry lifetime could be extremely short in worst cases, the GIC table will grow to approximate the size of address space. Unlike the address pool, GIC table will be visited more frequently. Each packet using IPv6/IPv4 translation will visit GIC table. If the scalability problem is not handled properly, this mechanism is fragile.

 ISBN: 978-1-61208-201-1

A. *Scalability in stateful method*

There are many ways to solve this scalability problem. *First, multiple gateway devices could be used for load balance.* Each gateway would be in charge of a part of the prefixes of the IPv6 subnet and store the GIC table of its own part. Using multiple gateway devices and using load balance technology could enhance the support of a large number of users. By separating the hosts in prefixes, the scale of GIC of every device could be reduced.

Second, a GIC entry could be shared by a group of hosts. Hosts could be sorted into small groups first. We can change the GIC allocate strategy from allocate a GIC for each host to allocate a GIC for each group. When the first client in one group sends a DNS request, a new entry is created for this group. GIC of one group should change from time to time to avoid guess attacks. The gateway keeps the former GIC for a period of time after the GIC changes because many hosts are still using the former GIC. Some former GICs and the current GIC should be considered as correct when verifying. Use multiple GICs for one prefix may reduce the performance, but dynamic GIC could provide more protection than fixed GIC. Assuming a method of separating address space by prefix is used, a final version would look like this. When gateway allocate an IPv4 address from the address pool to an IPv6 host, the gateway stores the IPV6 address A6, the IPv4 address A4 and the GIC G. Gateway keeps a group-shared GIC usage table, which logs the count of GICs hosts in one group. If one IPv4 address is leased from translator gateway, a GIC is assigned for the host of this address. The count of according entry will plus 1. When A4 is reclaimed, the count will minus 1. If a former GIC entry has a count of zero, this entry will be deleted because no host is using this GIC.

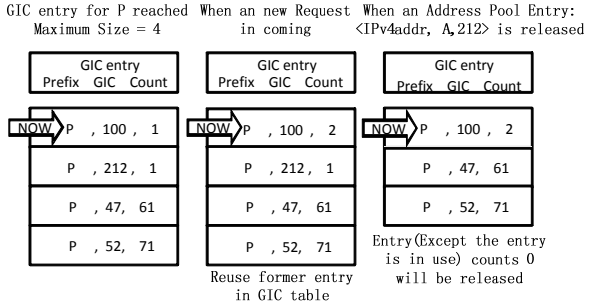

Figure 10. Prefix-shared GIC using count with reuse strategy

Below are the choices to cope with the case when the GIC set for this prefix has grown to the maximum size:

1. *Stop changing GIC temporarily.* This may cause the changing of GIC stops. But it may increase the risk of guess attacks.

2. *Delete the oldest item.* This method keeps the GIC changing, but may block connections which established in the past. For example, if the lifetime of GIC is 2days and a set contains 8 GICs, connections established 16 days ago maybe interrupted.

3. *Reuse the GIC in the oldest item for new requests.* The number of GIC in every set and the lifetime of every GIC could be adjusted to keep GIC changes for better filtering performance and sufficient security. Figure 10 shows this process.

B. *Scalability in stateless method*

In stateless translation, there are no allocation and reclamation of address in the address pool. One possible solution could be like this:

Set up a GIC table for the every prefix after separating. For each specific prefix, a list of GICs is maintained. The GICs in the list change periodically. Once a GIC is changed, the new one would be added to the end of the list. The new GIC is applied as soon as it is created. A variable indicates the visit number of the GIC in last small period would be added into the GIC entry. If one entry's visit number is zero, we may assume no one is using this GIC anymore. Then this entry would be removed from the table, as shown in Figure 11. The GIC entry of 100 is removed as the visit number is decreased to zero, but the entry of 52 is kept as it is a new entry just in use.

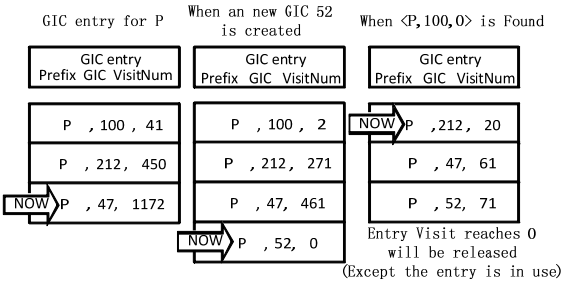

Figure 11. Prefix-shared GIC using visitnum

Deleting an entry with a non-zero visit number will pose risk to the system. If this list reaches the maximum length, we could use the reuse old-item method to maintain as shown in Figure 12. The old GIC entry of 100 is reused as the list reaches maximum length of 4. In this case, most users' interest will be protected. Special policy could be applied to delete old entries.

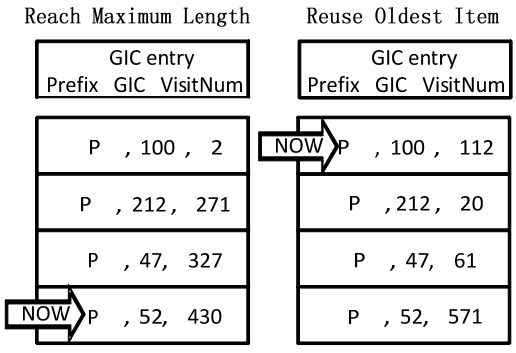

Figure 12. Prefix-shared GIC using visitnum with reuse strategy

 ISBN: 978-1-61208-201-1

C. Using real-time calculating in stateless method

When a packet reaches the gateway, the gateway should recognize the correct GIC. The gateway should have the mapping information of source address and GIC. In the previous part of this paper, we use a table to save GIC for verifying. In this part, we will introduce another approach.

A real-time calculating method could be used to replace GIC table. For security reasons, GIC should be private. For different addresses, they should have different GICs. So source address should be an input of the algorithm. For each incoming packet, the gateway uses its source address to calculate a GIC and compare with the one carried in the packet. To prevent guess attacks, the GIC should be dynamic. The algorithm needs a dynamic input besides address. This dynamic input for generating algorithm is called "seed", which should be unique and used for all hosts in the gateway. A set of seeds would be kept at the same time as multiple GIC is used.

Since the seed is unique, the safety of seed should be considered. The algorithm is usually public, if a seed can be inferred by one's address and GIC, the safety of this mechanism is challenged. So the generating algorithm should be irreversible.

Now this process would look like this: GIC is calculated with an input of seed and the address of source. GIC sent to the client in a DNS reply. When a packet comes from the client reaches the gateway, it calculates the GIC for this packet and check to see if they are equal. Each packet would trigger one calculating. The algorithm is the critical part of this method. It should be irreversible and fast. Comparing to store the whole GIC table, using real-time calculating is still an alternative choice when the scalability becomes a critical problem.

V. PERFORMANCE ANALYSIS

There are many parameters used in this mechanism, which includes the number of bits used in IPv4-mapped address N, the number of gateway devices used for load balance G.

Block Number (BN) indicates the number of small blocks which the whole address space has been separated to. Lifetime (LT) of GIC entry in stateful method would be from the temporary IPv4 is allocated from the address pool to the resource is reclaimed by the gateway. The number of entries in the block-GIC table is L.

Now, we will define key measurement of this method:

Protection: the percentage of address spoofing packets can affect the victim. Smaller value represents a better effect.

Lookup cost: the cost depends on the lookup algorithm and the number of GIC entry. Smaller value represents a better effect.

Minimum long connection time: the time a connection would possibly be reset by the gateway since it established. Bigger value represents a better effect.

Assuming the gateway has an IPv6 address space size: S_6 addresses. And an IPv4 address pool size: S_4 addresses.

Firstly, we will analysis how the bits that GIC take in IPv4-mapped address would influence the performance. There are 32 unused bits in IPv4-mapped address in total. When an attacker uses brutal force, it enumerates all 2^N combinations in GIC field. There are only L valid GICs at one time. The protection could be enhanced by more bits used for GICs. But the bits may be limited by longer prefix.

The GIC table size in stateful method is the size of IPv4 address space. In stateless method, it equals to the size of IPv6 address space. Both of them could be reduced by separating the address space to small blocks.

The GIC changes periodically. One GIC is valid until it is removed from the list. As the length of list is L, the minimum long connection time would be LT*T. If the reuse strategy in last section is applied, no connection would be interrupted by the GIC method. Table I and Table II show the performance of GEAV method in different scenarios.

TABLE I. PERFORMACE IN STATEFUL METHOD

	Stateful GIC	Stateful with separating blocks	NO GEAV
Protection	$1-1/2^N$	$1-L/2^N$	0
GIC entry number	S_4	S_4 /BN	N/A
Min long conn time	∞	LT*L (∞ if entry reuses)	∞

TABLE II. PERFORMACE IN STATELESS METHOD

	Stateless GIC	Stateless with separating blocks	NO GEAV
Protection	$1-1/2^N$	$1-L/2^N$	0
GIC entry number	S_6	S_6 /BN	N/A
Min long conn time	LT*L	LT*L (∞ if entry reuses)	∞

A typical campus network has an IPv4 address space of about one /16 and an IPv6 address space of some /48. This means the address pool size in IPv6/IPv4 translation would between tens of thousands to hundreds of thousands. The GIC table entry number would be around 100,000. The GIC table would take 100,000*(128+32+N*L). Setting N to 16 and L to 8, the GIC table would take 36MByte storage. If 100,000 entries lookup is a too heavy load for gateway devices, the IPv4 addresses pool could be spilt into small pools. Using 5 gateways, only about 20,000 entries would be loaded on one device. A /48 IPv6 prefix could be break into /60 blocks, only around 4,000 entries are needed. The deployment of GIC system would filter 65528/65536 attack packets, which is a significant benefit for the network security and performance.

We have a prototypical system in development and tested it with Tsinghua University IVI experimental deployment. This environment has an IPv4 prefix of 58.200.228.0/24, and

 ISBN: 978-1-61208-201-1

an IPv6 prefix of 2001:da8:ff3a:c8e4::/64 . There are one IVI router and 250 IVI-enabled hosts in this experimental subnet. Our GEAV device can support 64K GIC entries and bi-directional 10Gbps traffic in tests. After the development is finished, we will try to test it in real network.

VI. CONCLUSION AND FUTURE WORK

Applying with IPv4/IPv6 translation, "gateway embed and verify" (GEAV) use translated DNS reply to carry information to hosts. To map an IPv4 address to an IPv6 address, there are many unused bits in IPv4-mapped address. GEAV method uses these blank bits in the IPv4-mapped address of the DNS record to pass GIC information to client. The client uses this GIC-embedded IPv6 address for communication. The GIC will be carried in the destination address of all packets. Verification is performed on the translator gateway when a packet needs translation reaches. By implementing function on translator gateway and DNS translator, no change in host is required. GEAV is not transparent to hosts, but it will not affect communications between hosts. The risk of address pool being exhausted would be reduced by using this technology. Applying this technology with stateless translation method can also help reduce a lot of address spoofing packets. In the paper, we analyses the correctness of GEAV method, and solve the scalability problem when applied to both stateless and stateful translation method.

But, GEAV method still needs a lot of improvement. Unfortunately, this method can only work with DNS-based IPv4/IPv6 translation for now. GEAV method cannot provide protection when facing middle-man attack. Until now the tests are done in experimental network, which only has a small number of hosts. We are still improving the prototype system and try to deploy it in the campus network which has more users than the experimental subnet. Also, the algorithm, which can be used in the real time calculation, needs future discussion. An irreversible hash algorithm is needed, and there is a tradeoff between performance and security.

VII. ACKNOWLEDGEMENT

This paper is supported by National Science Foundation of China under Grant 61073172, Program for New Century Excellent Talents in University, and Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant 20090002110026.

REFERENCES

[1] http://www.apnic.net/__data/assets/pdf_file/0018/33246/Key-Turning-Point-in-Asia-Pacific-IPv4-Exhaustion_English.pdf [retrieved: May, 2012]

[2] M. Leber, "Global IPv6 Deployment Progress Report". Hurricane Electric. 2010. http://bgp.he.net/ipv6-progress-report.cgi [retrieved: May, 2012]

[3] C. Labovitz, "Botnets, DDoS and Ground-Truth -- A Look at 5,000 Confirmed Attacks", NANOG 50, October, 2010.

[4] R. Beverly, A. Berger, Y. Hyun, and K. Claffy, "Understanding the efficacy of deployed internet source address validation filtering", Proc. 9th ACM SIGCOMM conference on Internet measurement conference, pp. 356–369, 2009.

[5] The MIT ANA Spoofer Project, http://spoofer.csail.mit.edu/ [retrieved: May, 2012]

[6] J. Bi, G. Yao, J. Wu and F. Baker, "SAVI Solution for DHCPv4/v6", draft-bi-savi-cps-02.txt, October, 2009

[7] Unicast Reverse Path Forwarding, Cisco IOS, http://www.csm.ornl.gov/~dunigan/oci/uni_rpf.pdf. [retrieved: May, 2012]

[8] X. Liu, X. Yang, D. Wetherall, and T. Anderson, "Efficient and secure source authentication with packet passports", Proc. of SRUTI '06, pp. 2-2, July, 2006.

[9] K. Park, and H. Lee, "On the effectiveness of Route-Based packet filtering for distributed DoS attack prevention in Power-Law Internets", Proc. 2001 SIGCOMM, volume 31, issue 4, pp. 15-26, Oct 2001.

[10] A. B. Barr, and H. Levy, "Spoofing prevention method". Proc. IEEE INFOCOM, 2005.

[11] G. Tsirtsis, P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)", RFC 2766, February, 2000

[12] C. Aoun, E. Davies. "Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status", RFC4966, Jul.2007.

[13] M. Bagnulo ,P. Matthews and I. van Beijnum," Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", April 2011.

[14] X. Li. C. Bao,F. Baker and K. Yin," IVI Update to SIIT and NAT-PT draft-baker-behave-ivi-01", September 16, 2008.

[15] P. Srisuresh, G. Tsirtsis and P. Akkiraju,A. Heffernan, "DNS extensions to Network Address Translators (DNS_ALG)", September 1999

 ISBN: 978-1-61208-201-1