# The Automatic Peer-to-Peer Signature for Source Address Validation

Yan Shen, Jun Bi, Jianping Wu, and Qiang Liu

Network Research Center, Tsinghua University,
Beijing 100084, China

**Abstract.** SPM (Spoofing Prevention Method) proposed a peer-to-peer anti-spoofing method which could effectively filter spoofed packets and support incremental deployment. However, mechanism of SPM key updating has some serious problems: (1) heavy management cost; (2) risk of becoming the target of DOS/DDOS attacks during the key updating; (3) limitation of the scale of the SPM union; (4) impossibility of tackling the threat of real-time packet sniffing. This paper proposes an Automatic Peer-to-Peer Anti-spoofing Method (APPA). APPA resolves the key management problem and could effectively prevent source address spoofing.

**Keywords:** Source Address Spoofing, Spoofing Prevention, Key Management.

## 1 Introduction

Spoofing of source IP address is often used by malefactors to launch DOS/DDOS (Denial of Service) attacks [1] [2] [3]. Attackers take advantages of spoofing to make it hard for the victim destination to categorize malicious traffic by source. Nowadays economic and trade services are in fast development in internet, such as e-commerce, e-bank, IP spoofing are much more commonly used by criminals. Spoofing gives rise to a number of challenges for law enforcement. If not deterred, spoofing would make much more serious threat to the internet and even the social life in the foreseeable future.

Great efforts have been made to prevent IP address spoofing. uRPF (Unicast Reverse Path Forwarding) [4] and Ingress filtering [6] are famous of them. Ingress filtering filters out packets containing unexpected source IP address at the source of the packets' route, while uRPF filters out packets with source IP address which is not expected to exist in a route. Both of them are effective and fast. IETF OPSEC (Operation Security) work group suggests that the combination of Ingress filtering and uRPF can solve the spoofing problem. However, Ingress filtering and uRPF are not designed for stepwise deployment and won't work well if it is not deployed in large scale. Ingress filtering also can't recognize spoofing in the same AS (Autonomous System) and uRPF can't filter packet spoofing in the same route. SPM (Spoofing Prevention Method) [5] presents an anti-spoofing method which is effective and also supports stepwise deployment. However, SPM is not DOS resilient in the key-update process. It also suffers from sniffing and can't be used for anti-spoofing of smaller granularity.

This paper present a new anti-spoofing method called Automatic Peer-to-Peer Anti-spoofing Method (APPA), which derives from and improves upon SPM. APPA

B. Apolloni et al. (Eds.): KES 2007/WIRN 2007, Part I, LNAI 4692, pp. 855–863, 2007.

uses the same tagging and verifying key mechanism as SPM and makes the key-update process automatically. It is effective, DOS resilient, applicable in large scale, anti-sniffing and designed for stepwise deployment.

The rest of this paper is as follows. Section 2 introduces the related work of anti-spoofing. Section 3 introduces the basic rationale and the implementation choices of APPA. In section 4, we proposed a detailed solution of APPA used for inter-AS anti-spoofing. Section 5 is the comparison of APPA and SPM. Section 6 is a brief introduction of future work of APPA. Section 7 concludes the whole paper.

## 2 Related Work

Methods for IP address anti-spoofing could be categorized into 3 types: traceback, filter in route, and filter at destination. Traceback is used to trace the real source of the malicious packets when attack happens. It is of forensic value of law enforcement and will help catch and prosecute the attackers. According the way of tracing, Traceback has three different types. (1) The routers on the route add information into the packets. [7][8][9][15] are typical traceback of this type. (2) The routers on the packets' route send ICMP messages to the destinations of the packets. [10] [11] belong to this type. (3) The routers on the packets' route deposit the digest of the packets. [12][13] are of this type. Traceback does not work well against spoofing. Even it could exactly trace back to the attacker it will not work well, since many hosts in the internet are comprised by attacks, traceback may find out the comprised host instead of the real attackers.

Filtering in the route, the second category of anti-spoofing methods is more effective. The typical two are Ingress filtering [6][14] and uRPF[4]. Ingress filtering brings benefits to the network that deploys it only if it is deployed in large scale, hence it is lack of incentive for stepwise deployment. uRPF is based on that a packet destined to a given AS would traverse a specific route. uRPF permits attackers to spoof any units on the same route to a victim. It also suffers from asymmetry route. Methods mentioned in [16] to solve the problem of uRPF in asymmetry route is rough and would even weaken the original effect of uRPF. uRPF also does not support stepwise deployment.

Hop count [16] and SPM [5] do filter at the destination. Hop count uses TTL value as a key for each (source, destination) pair. Hop count could filter out 90% of the forged packet if the attackers do nothing to the TTL value. However, the TTL value is small and the attackers could easily learn the right TTL value he should put in packets. It does not work to prevent sophisticated attacks.

SPM proposed a novel peer-to-peer anti-spoofing method. It is effective and light-weighted, also brings great incentive for internet owner to deploy it and directly benefits them by conquering the deficiency of trace back and route filtering. If adequately deployed, SPM could filter out 99.999999% of the forged packet and it is much safer than Hop count. Compared with AH [17] of IPSec, it is light-weighted and would not become a target of DOS/DDOS attack.

In SPM, a packet is tagged with a key at the border of the source AS, and the key is verified and removed at the border of the destination AS. The key is associated with the (source, destination) pair and is kept secret from any AS else. SPM suggests to change

keys every two 2 hours for safety. It provides two ways to update keys (one active way and one passive way), both needs safe interactions between every pairs. As SPM thought it is difficult to sniff in bone network, keys are used in plain text.

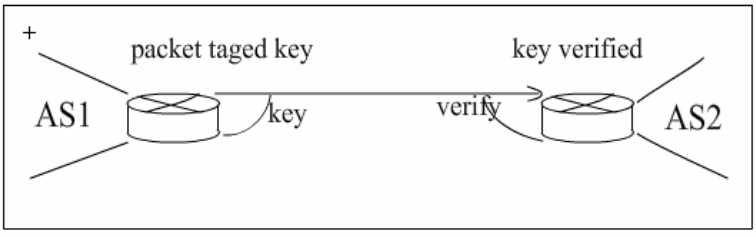


**Fig. 1.** SPM, packet is tagged with a key and the key is verified at the destination

We developed an implement of SPM and found out SPM have the flowing flaws:

(1) SPM could not be used in large scale. Keys need be changed to keep safe. SPM AS needs to interact with every ASes else in the SPM league. When the network is busy, it is hard to finish all the interactions in half an hour when 2000 ASes take part in the SPM league.

(2)The interaction could easily become the target of DOS/DDOS attack. This makes the key updating became much more difficult.

(3) SPM has no idea to deal with sniffing so could not be used where sniffing is not so hard.

## 3 APPA

### 3.1 Basic Rationale

As a signature-and-verification method, APPA tags a key into each packet at the source and verifies the key at the destination. Each key is used only in one packet and will change in the next packet. This makes eavesdropping and replaying the key or the whole packet useless. The key updating is finished automatically by special mechanism rather than interaction. This improves its safety and efficiency.

### 3.2 Updating Keys

The most important thing for APPA is updating key safely and efficiently without interaction, in spite of any states of the network. To implement APPA, the following issues must be satisfied. 1. The key change can not be inferred from the known serial keys by anyone except the source and destination. The only opportunity left for attackers is brute-force guess. 2. Updating key should be very fast. 3. The choice of algorithms should be pretty much, so APPA would keep robust and safe.

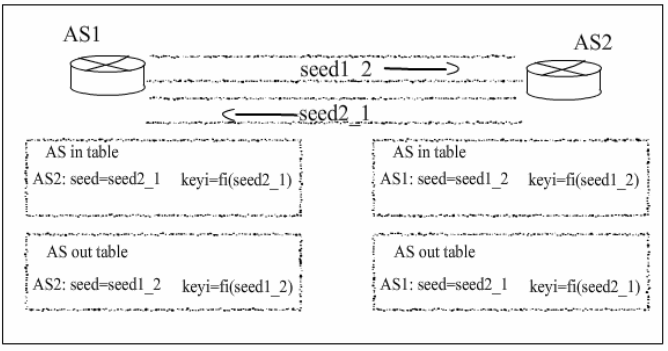


**Fig. 2.** The seed exchange process uses special method (e.g. Diffle Hellman protocol) to exchange seeds between peers. Keys are generated from the seed.

Random Number Generator(RNG) fits all the request of APPA. RNG generates a series of numbers that satisfies several tests of statistics in mathematics. Good RNGs have several characteristics: 1. Fast. 2. Well distributed, pass all statistical tests known. 3. Independent. 4. Portable and reproducible. 5. Long periods. 6. Large seed space. 7. Unpredictable and irreproducible.

The last one is an additional request for cryptology. [18][19] describe RNGs in detail.

APPA needs the request of 1,6,7 of RNGs, that is fast, large seed spaces and unpredictable .There are many RNGs fit for APPA such as [22][24]. Every source and destination pair only needs to exchange a seed for RNG, then all the sources and destinations could easily update and synchronism keys.

To enhance safety, we could make the result of f(seed_a) $\oplus$ f(seed_b) the key, where f is a RNG. This is similar to the One-Time-Pad (OTP) cipher mechanism [18] in cryptology.

Countless good RNGs are fit for APPA such as [20] [21] [22].The progress in RNG theory may cause some RNGs unsafe but we still have choices. The long period of RNGs omits any more interactions in pairs except the first seed exchange.

## 4 Inter-AS Solution of APPA

### 4.1 Overall

In this section we give one Inter-ASes anti-spoofing solution in detail.

The phase of tagging keys and verifying keys in packet is similar as SPM. Packet is tagged with a key at the border of the AS and the key is verified at the border of destination AS. Keys are changed periodically to enhance safety. APPA updates keys automatically rather than by interaction in SPM, it can generate millions of keys in 1 second.

### 4.2 Initialization

The initialization is very important to APPA because it decides all the keys. In our solution we choose The KISS Generator as the key generator. The task of initialization

is to exchange seeds between pairs. Suppose there are N ASes in the APPA league and 1 new AS now takes part in. The fresh APPA AS needs to exchange seeds with all the N ASes. This is called the Initialization.

To accomplish the initialization of seeds exchange safely, the SA (Security Association) process of IPSec and the Diffie Hellman exchange are practical ways. We use a combination of TCP Interception and Diffle Hellmann protocol [26] to exchange seeds. The KISS Generator needs a seed of 128 bits and we use two seeds to generate one key, so the seed's length is 256 bits:

$$key = KISS(seed\_1) \oplus KISS(seed\_2)$$

The KISS Generator is safe, fast and has a period of $2^{124}$ .No more interactions is needed after the initialization of seeds exchange, unless one AS thinks it necessary to change the seed.256 bits seed make it safe to brute force guess. The KISS can generate millions of keys per second with normal hardware nowadays, so keys can change as fast as possible and the APPA league can be very large.

### 4.3 The Length, Position, and Period

The length of the key is directly associated with the safety while under attack. SPM suggested 32bits-length key. In IPV6, the length is not the problem because it can be put in expansion head, in [24] the key is even up to hundreds of bytes .In IPV4, the length is a problem. Considering the efficiency of padding and verifying, the key should be placed in the IP head. The IP option field is 32 bits and enough in length, but based on the report of [23], more than 70% of the connection requests would be failed and 44% of the connection would break if the packets contain unknown IP option field. The IP identification could work but is only 16bits long. So in IPV6 the key could be placed in expansion head, while in IPV4 in expansion head but keys need faster change.

Obviously, the faster the key changes, the safer is the key. Shannon proved one-time key is the most safe in [25]. SPM suggested the 32-bit key changes every two 2 hours that is not safe under brute-force attack, we'll give an analysis for that in the next section. APPA suggests the 32-bit key period 10 minutes and could provide even one-time key for each packet.

### 4.4 Failure Recovery

When a router reboots by spontaneity or accident, its keys for other ASes is out of time and it also does not know what keys other ASes would use, unless it knows how much time it has shut down. An AS often has more that one border router and the rebooted router can easily get the AS-in-table and AS-out-table from other routers. Only if all the routers which have the tables reboot at the same time period, we have to learn the tables from other ways. One possible way to learn from the in-bound packets, compute keys use the table before rebooting to match the keys gets in the packets. That is fast and easy .Another way is to use series number as part of the key, or use the same keep-alive scheme in TCP. The failure recovery is not difficult for APPA.

### 4.5 Anti-sniffing

SPM mentioned that it is hard to sniff in backbone network so the key can be used in plain text. We agree that it is feasible in inter-AS solution and also think it is useful to develop ways for anti-sniffing, as sniffing possibly exists in backbone network and anti-spoofing should finally be deployed for end-to-end systems.

The main threat of sniffing is that attackers could launch replay attack, which makes many secure mechanisms brittle. If attacker could get the key by sniffing, he could launch spoofing attack easily. The key point is anti-replay for anti-sniffing. Many methods have been proposed for anti-replay, such as time-stamp, sequence number, bloom filters [27]. Time-stamp needs global clock synchronization, while bloom filters need large cache spaces. IPSec AH [17] uses sequence number and a 32-size slide window to performance anti-replay, it is feasible because AH is end-system oriented, while the packets between two end systems' session are less possible in disorder. The slider window needed in inter-AS solution is large, and every peer has to keep n-1 windows while the league size is n. Nowadays the flux of packets between ASes is very huge and fast. For the reasons above, anti-replay is a difficult task in inter-AS solution, unless we don't care the huge cost which even becomes a bottle-neck.

APPA could achieve anti-replay by two ways. 1) To change keys as fast as possible, the limit of that is to change key for each packet and every key is used only in one packet. If most of the packets arrive the destination in order as they leave the source, keep a small slide-window would achieve anti-replay effectively as APPA generates keys very fast. 2) To use a combination of sequence number and Chaos Theory[28], and also take advantage of APPA's fast changing key. One possible solution is as following steps:

1) APPA generates a key, called *k0*.
2) Each packet from *S* to *D* contains an increasing sequence number, which are 1,2,3,4...
3) The key tagged in the packet is calculated from *L(k0, sequence number)*, where *L* is a Chaos-based function, e.g., *L* is a four-times call of Logistic mapping, that is,

$$X_{n+1} = ax_n(1-x_n), 0 < a \le 4, 0 < x_n < 1, \text{(Logistic mapping of Chaos Theory)}$$

Logistic mapping has the butterfly effect after 3 or 4 times iteration. Hence *L(k0, sequence number)* produces safe and unpredictable keys. The source and destination could calculate and verify the keys easily but the attackers not. The *k0* is changing fast so it is safe. A bit map is kept to denote which sequence number has been used. If *k0* is updated, the sequence is reset to 0 and the bit map too.

Anti-replay is not necessary in inter-AS solution and could be an option function. However, APPA is easy to against replay attacks.

## 5 Analysis and Comparison

The main purpose of this section is to give analysis and comparison to evaluate the advantage that APPA over SPM and other relative methods. The management cost,

safety and expansibility is evaluated in relative-comparative manner with SPM, and the efficiency and safety with other methods.

The first and foremost deficiency of SPM is the high-frequency interaction for update keys, which is not DOS resilient and even not busy-network resilient. That is, the interaction would become hard while DOS attack exists or the network is in busy state. In contrast, APPA only needs the initial interaction to exchange the seeds, the trait of long period and unpredictable makes the key safe, and the large seed space makes it safe under brute force attack.

The total number of key-update interactions during a short key period in SPM is increased by $O(n^2)$, which is zero in APPA after the initial interaction. Attackers could take advantage of this to launch attacks to block part of key-update process or DOS at the router/server. On the other hand, though each participant only does $O(n)$ interactions but it is still difficult to finish in short time. Our experiments show that a SPM league with 2,000 participants is hard to update keys in half an hour while the network is just normally busy. That means neither could SPM update keys fast nor could SPM have many participants.

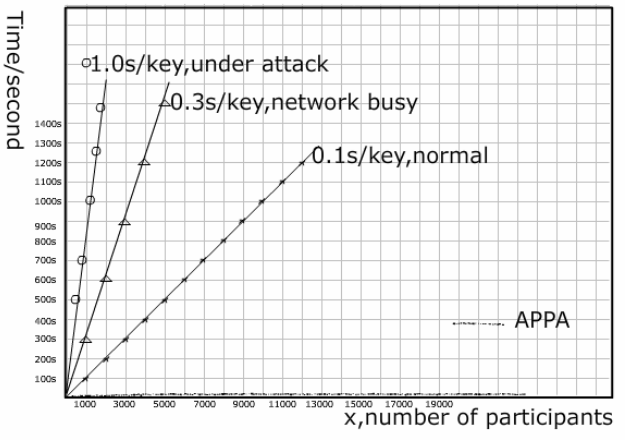


**Fig. 3.** This figure shows the time each participant spends to update keys with other participants. The x axis denotes the size of the league, while the y axis denotes the time needed to update keys. SPM needs much more time to update keys, while APPA needs only few time. APPA could update millions of keys per second, even billions of participants won't bring problem.

Another aspect should be considered is the length of the key.The key should be as short as possible to improve the effective rate payloads. This is contradicting with the safety request. If the attackers control 1GB bandwidth, a 32-bit key would be doped out in $\frac{2^{32}\times 64\times 8}{2^{30}}$ seconds. Updating the key faster provide more safety and reduce the

length of the key. APPA could easily make the key period a few seconds even reach one key per packet, which provide perfect safety for anti-spoofing.

## 6 Future Work

As described in section 3, anti-replay is difficult in inter-AS solution, because the inter-AS flux of packets in huge and fast, any known anti-replay methods would bring great loss of network performance. The APPA could change keys very fast, that brings a glimpse of hope to solve anti-replay in inter-AS solution. That is, each key is used only in one packet. This makes it impossible to launch replay attack because each key is used only once. However, because of the packet would arrive in disorder or lost in the route, a slide window is needed for each peer, to keep memory that which key has been used. When the league size is large, this needs large memory spaces to deposit information. The usage of Logistic mapping also needs a bit map for checking used sequence number for each peer, which also needs much space. One of our future works is to perfect one-time key.

APPA's low management cost makes it easy to be used in smaller granularity anti-spoofing, the limit of which is spoofing prevention between end-systems. That is another aspect of our future work.

## 7 Conclusion

This paper proposed a peer-to-peer based anti-spoofing method, APPA, which solves the deficiency of a former peer-to-peer based method (SPM). APPA requests low management cost and is safe enough, also has all the merits those SPM has. It could be used in large scale and smaller granularity.

The APPA inter-AS solution also provide a feasible way to achieve anti-replay. APPA brings great improvement on SPM and provide an effective way to solve anti-spoofing in smaller granularity.

## References

[1] Moore, D., Voelker, G., Savage, S.: Inferring internet Denial-of-Service activity. In: in Proc. USENIX Security Symposium, pp. 9–22 (2001)
[2] CERT Advisory CA-, 01:Denial-of-service Development (January 2000), http://www.cert.org/advisories/CA-2000-01.html
[3] Mazu, Networks. Enforcer, 2002, http://www.mazunetworks.comproducts
[4] Cisco, I.O.S.: Unicast reverse path forwarding (1999)
[5] Anat, B.B., Hanoch, L.: Spoofing Prevention Method. In: Proceedings IEEE Infocomm (2005)
[6] Baker, F., Savola, P.: Ingress Filtering for Multihomed Networks. RFC 3704 (March 2004)
[7] Dawn, X.iao,D.S., Adrian, P.: Advanced and authenticated marking schemes for IP traceback. In: Proceedings IEEE Infocomm 2001 (April 2001)

[8] Kihong, P., Heejo, L.: On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack. Tech. Rep. CSD-00-013, Department of Computer Sciences. Purdue University (June 2000)
[9] Micah, A.: Tradeoffs in probabilistic packet marking for IP traceback. In: Proceedings of 34th ACM Symposium on Theory of Computing (STOC), New York (2002)
[10] Belenky, A., Ansari, N.: On IP Traceback. IEEE Communications Magazine 41(7) (July 2003)
[11] Bellovin, S., Leech, M., Taylor, T.: Icmp traceback messages (February 2003), http://www.ietf.org/internet-drafts/draft-ietf-itrace-04.txt
[12] Alex, C.S., Craig, P., Luis, A.S., Christine, E.J., Fabrice, T., Beverly, S., Stephen, K., Strayer, W.: Single-packet IP traceback. ACM/IEEE Transactions on Networking (December 2002)
[13] Timothy, W.S., Christine, E.J., Fabrice, T., Regina, R.H.: SPIE-IPv6: Single IPv6 Packet Traceback, Local Computer Networks, 29th Annual, IEEE International Conference on (16-18 November 2004), pp. 118 – 125 (2004)
[14] Ferguson, P., Senie, D.: Network ingress filtering: Defeating denial of service attacks which employ ip source address spoofing (May 2000), RFC (2827)
[15] Amin, S.O., Kang, M.S., Hong, C.S.: A Lightweight IP Traceback Mechanism on IPv6. In: Zhou, X., Sokolsky, O., Yan, L., Jung, E.-S., Shao, Z., Mu, Y., Lee, D.C., Kim, D., Jeong, Y.-S., Xu, C.-Z. (eds.) Emerging Directions in Embedded and Ubiquitous Computing. LNCS, vol. 4097, pp. 671–680. Springer, Heidelberg (2006)
[16] Cheng, J., Wang, H.N., Kang, G.S.: Hop-count filtering: An effective defense against spoofed DDoS traffic. In: Proceedings of ACM CCS ' (2003)
[17] Kent, S.: IP Authentication Header. RFC 4302 (December 2005)
[18] Mathew, S., Jacob, K.P.: A New Fast Stream Cipher: MAJE4. IEEE INDICON 2005.
[19] Stephen, K.P., Keith, W.M.: Random number generators: good ones are hard to find. Communications of the ACM archive 31(10) (October 1988)
[20] Matsumoto, M., Nishimura, T.: Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. 8(1), 3–30 (1998)
[21] George, M., Wai, W.T.: The 64-bit universal RNG, Letters in Statistics and Probability. 6(2), 183–187 (January 2004)
[22] George, M.: The KISS generator. http://oldmill.uchicago.edu/~wilder/Code/random/Papers/Marsaglia_2003.html
[23] Alberto, M., Mark, A., Sally, F.: Measuring the Evolution of Transport Protocols in the Internet. ACM Computer Communication Review, 35(2), April (2005), http://www.icir.org/mallman/papers/tcp-evo-ccr05.ps
[24] Li, X., Yang, X.W.: Efficient and Secure Source Authentication with Packet Passports. In: Proc. of USENIX SRUTI, San Jose, CA (2006)
[25] Shannon, C.E.: Communication Theory of Secure Systems. Bell System Technical journal 28(4), 656–715 (1949)
[26] Whitfield, D., Martin, E.H.: New directions in cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
[27] Li, F., Pei, C., Almeida, J., Broder, A.Z.: A scalable wide-area web cache sharing protocol. Technical Report 1361, Department of Computer Science. University of Wisconsin-Madison (1998)
[28] Baptista, M.S.: Cryptography with chaos[J]. Physics Letters A, 50–54 (1998)