Implications of Fragmentation and Dynamic Routing for Internet Datagram Authentication

Gene Tsudik

Networks and Distributed Systems Laboratory
Computer Science Department
University of Southern California

AND
IBM Los Angeles Scientific Center

[1] This work was supported in part by the grants from GTE, NCR and the National Science Foundation.

Abstract

With the recent growth of internets, large networks connecting heterogeneous entities, access control is an issue no longer confined to individual hosts. Recent literature suggests that authentication may have to be performed in gateways as well as on an end-to-end basis. In this paper we discuss the implications of fragmentation and dynamic routing on gateway-level authentication in packet-switched networks like DARPA Internet. Two algorithms are presented that permit fragmentation and dynamic routing to some extent, while allowing the gateways to authenticate successive packets belonging to authorized connections.

Introduction

Recently, there has been a surge, both in numbers and size, of internets -- large networks connecting administratively heterogeneous entities such as government agencies, academic and research institutions and commercial sector. The DARPA Internet is an example of one such internet. One of the problems incurred by these internets is that of access control across organizational boundaries.

Estrin [1], gives a thorough treatment of various access control issues in large internets. Recently, Estrin, Mogul and Tsudik [8] presented several variations of a Visa Scheme [2], a network-level mechanism for inter-organization access control in DARPA-like internets. What these discussions point to is the need for the low-level support that allows access control at the gateway level to coexist with certain properties of packet-switched (datagram) networks. Fragmentation and dynamic routing are two of the most problematic issues that at the first glance seem to discourage authentication in gateways.

It has been suggested that fragmentation is, at best, a necessary evil [2][3], and must be avoided if access controls are to be introduced at the gateway level. Similarly, routing is frequently restricted to a single route [6][4] or a set of routes to accommodate a specific access control mechanism, thus failing to take advantage of the ever changing topology and load balancing on gateways.

When an authentication mechanism is introduced at the packet level, an authenticating gateway has to be able to verify that each packet passing through it belongs to some authorized connection (or has originated at a trusted host) by checking some property of a packet, e.g. a data signature. If authentication is done in a straight-forward manner, the gateway must have access to each packet in its original form, i.e., the same as it was when it left the source host. If the source host knows in advance the Maximum Transmission Units, MTUs, of intermediate networks, it can pre-fragment the packet into small enough pieces so that each piece will reach the gateway unfragmented, thus allowing for immediate authentication. However, since hosts rarely possess such information, packets will usually reach the gateway in fragments. Moreover, fragments can arrive out of order. It is evident, therefore, that in order for a gateway to be able to authenticate individual packets it must either receive them

unfragmented or have some means to authenticate each packet on the basis of its fragments.

In this paper we attempt to reconcile to some degree fragmentation and dynamic routing with network-level authentication in gateways. We present two methods - one that allows fragmentation but requires state information in gateways and where authentication is "delayed", and one that curtails fragmentation by limiting packet sizes, while allowing for immediate authentication.

Terminology

The following abbreviations and symbols are used throughout:

- src - source network or organization
- dst - destination network or organization
- trans - transit network or organization
- GW - gateway
- H - host
- K_{priv} - private key
- F_{char} - characteristic function, e.g., checksum.
- P - packet
- PK_{id} - packet identification number.
- FP_i - i-th fragment of a packet.

Network Environment

The algorithms described below are designed to operate on IP networks such as DARPA Internet as well as other, privately operated internetworks. The general approaches should, nevertheless, be applicable to most network-layer datagram protocols; the implementation would be protocol dependent. The reader, unfamiliar with IP is referred to [4] for more background information.

Since IP comes in several different "flavors" we make some assumptions about its operation:

1. Packets (and packet fragments) can be fragmented more than once.
2. Fragments of the same packet, as well as successive packets between the same source-destination host pair, can take different routes.
3. Fragments of the same packet, as well as successive packets, may arrive out of order.
4. Packets between a particular pair of hosts can be uniquely identified by a sequence (identification) number in the IP header. [4]
5. The overall length of the packet can be inferred from its fragments. (See "Determining Packet Length" below).
6. Options can be introduced in the IP header[4]

The presence of an access control mechanism similar to the Visa Scheme described in [8] or [2] is assumed. In any one of these schemes each authenticating gateway shares a secret key with the source host by means of which successive packets are authenticated.

Delayed Authentication

This method is particularly applicable under the following set of conditions:

- Authentication is performed by either of the following methods:
  1. The source host (who possesses a secret key, K_{priv}) encrypts a packet with K_{priv} and computes the characteristic function over it producing the packet signature, $F_{char}(K_{priv}(P))$. The packet is then sent in cleartext format with the signature attached to it, i.e. $< F_{char}(K_{priv}(P)), P >$.
  2. The source host computes a characteristic function over the cleartext, producing $F_{char}(P)$, encrypts the packet with K_{priv} producing $K_{priv}(P)$ and sends $< F_{char}(P), K_{priv}(P) >$.

- The characteristic function used to compute packet signatures is TRANSITIVE, i.e.,

$$F_{char}(AB) = F_{char}(A) \text{ op } F_{char}(B)$$

where 'op' is some arithmetic or logical operator.

Dynamic routing, can be employed with the restriction that all fragments of the same packet travelling between H_{src} and H_{dst} must pass through GW_{src} and GW_{dst} at some point in the path. In other words, two fragments from H_{src} can reach H_{dst} by paths P1 and P2 as long as

$$\{GW_{src}, GW_{dst}\} \text{ is in intersection}(P1, P2)$$

The essence of this algorithm is that each of GW_{src}, GW_{dst} keep state information on a per packet[2] basis. Here packet is more of a logical entity since it may be fragmented before it reaches either gateway. The fragments of the packet are processed and forwarded but the F_{char} of each fragment is kept since there is no way to authenticate individual fragments. When the last fragment is received the gateway can finally verify the authenticity of the whole packet. If the composite F_{char} of all fragments matches the $F_{char}(P)$ the gateway forwards the last fragment, otherwise it is discarded.

The last step, in case of unsuccessful match, is the crucial one. By dropping one fragment of the packet, the gateway prevents the destination host, H_{dst}, from reassembly of the packet fragments, thus forcing it to drop the rest of the fragments [4][7].

One of the benefits of this method is that the hosts don't have to do anything special in order to implement it. IP itself will time out and discard the fragments on its reassembly queue in H_{dst} if one of the fragments doesn't arrive within a certain time.

In more detail, the algorithm is as follows:

1. When a gateway, say GW_{src}, receives a fragment, FP_i it first checks its table using H_{src},H_{dst} and PK_{ID}, to see if any other fragments of this packet have been processed. If not, it makes a new entry in the table.

2. The fragment, is then encrypted (or decrypted, see above) with K_{priv} and $F_{char}(FP_i)$ is computed.

3. If FP_i is the whole packet, $F_{char}(FP_i)$ is compared with $F_{char}(P)$ found in the header and if two values match, it is sent on and the entry in the table is freed. Otherwise the packet is discarded.

4. If FP_i is only a fragment of the original packet two cases arise:

   a. FP_i is the (not necessarily logical) last fragment of the original packet. Then, GW_{src} computes the combined F_{char} of all previously processed fragments and FP_i and compares it with the $F_{char}(P)$ (or $F_{char}(F(P))$) found in the header. Once again, if the values match, FP_i is sent on, otherwise it is discarded. In both cases the table entry is freed.
   b. FP_i is not the last fragment. GW_{src} stores the length, $F_{char}(FP_i)$ and the offset (which can be found in IP header) of FP_i in the original packet in the table. FP_i is then forwarded.

---

2 The term packet is used in an end-to-end sense, i.e., a collection of fragments that share an ID

The functions of GW_{src} and GW_{dst} are almost equivalent. The only difference is their operation is in that GW_{dst} can receive a larger number of fragments of the same packet as a result of some transit fragmentation.

Determining Packet Length

We consider two methods for determining overall packet length from its fragments. The first method consists of introducing original packet length in all fragments of the packet. In IP this would entail using a new Option in IP header which must be copied onto all fragments. This can be accomplished by setting a COPIED flag in the OPTION TYPE field. Any IP-speaking gateway would then copy the length onto all fragments. This approach is easy to implement but it has some obvious drawbacks -- increased (although, minimally) packet length and processing time in gateways. Its main advantage is that it provides the authenticating gateway with the overall packet length as soon as the first packet fragment is received.

The second approach requires no protocol changes but is somewhat less reliable. In it, a gateway finds out the packet length only when the last logical packet fragment is received. It does so by adding the FRAGMENT_OFFSET field (already present in IP header) with the DATA_LENGTH field of the last fragment. The gateway can detect the logical last fragment by checking the MORE_FRAGMENTS flag in the IP header. This approach is the easier of the two, since it requires no protocol changes and introduces no additional packet length, however, it is more prone to errors.

As an example, consider a situation when a gateway receives a fragment of length X. It processes this fragment and sends it on. Shortly thereafter it receives another fragment of the same packet, with length Y, which is the last logical fragment. The gateway calculates the overall packet length and discovers that it is less than X. At this point all it can do is drop the second fragment and hope that the either GW_{dst} or H_{dst} drop the previous fragment. If the first method is used such errors can be caught on the spot, i.e., the gateway would have realized immediately that the first fragment is too large and would have dropped it, thus, not littering the network with useless traffic.

Algorithm with Probe Packets

The second algorithm makes no assumptions about authentication method nor does it require any additional state information in gateways, but it curtails the use of fragmentation and involves some extra packet overhead.

The main idea behind this method is for the source host, H_{src}, to figure out the maximum packet size that can reach the destination host's gateway, GW_{dst} without being fragmented. There are several ways to accomplish this - one is discussed in a recent paper by Kent and Mogul [3]. It requires sending a special PROBE packet onto which each intervening gateway stamps the MTU (Maximum Transmission Unit) of its network interface. This has the unfortunate implication of having to change the protocols on all gateways in order to support the information gathering. Also, an overhead of the PROBE packet is incurred as well as the option processing time in all gateways.

We propose a modification of the above approach in the following manner. Instead of H_{src} sending a PROBE packet, it sends a normal, useful packet of minimal length, say 576 bytes (an IP minimum). However, before the packet is sent on, H_{src} pads it with white space upto the limit of its local

MTU (it can never send a packet whose length is greater than the local MTU without incurring fragmentation). The first authenticating gateway, GW_{src} drops all but the logical first fragment of this packet, $FP0_{src}$ (it determines that by checking for FRAGMENT_OFFSET = 0 condition). Note that, when the first logical fragment ($FP0_{dst}$) arrives to GW_{dst}, it is not always the case that

$$length(FP0_{src}) = length(FP0_{dst})$$

In fact, it is reasonable to assume that the length $FP0_{dst}$ would be less than that of $FP0_{src}$ due to transit fragmentation. GW_{dst} then sends $FP0_{dst}$ onto H_{dst} (resetting MORE_FRAGMENTS flag in advance) which receives the packet and can now use what little data is in it. At the same time GW_{dst} sends a special packet (possibly via ICMP [5]), which contains the length of $FP0_{dst}$, back to H_{src}. On receipt, H_{src} can send signed packets of the known length to H_{dst}. Both GW_{src} and GW_{dst} are able to check the authenticity of the packets on the spot since no intermediate fragmentation will occur along this path.

At this point one might wonder why this approach is better than having no fragmentation at all. First, some fragmentation can take place - namely beyond GW_{dst}, since GW_{dst} and H_{dst} may not be on the same network. Second, having no fragmentation at all implies either sending VERY SMALL packets thus congesting the network unnecessarily or taking chances and disallowing fragmentation (by setting DONT_FRAGMENT flag in IP header) which can result in denial of service, poor response time and a slew of ICMP [5] "FRAGMENTAION_NEEDED" packets [3].

Dynamic routing can be used with the same restriction as in the first scheme. Multiple routes can be used as long as all fragments of the same packet pass through same GW_{src},GW_{dst} pair. But, what happens if a route suddenly changes and the minimum MTU on the new route is less than the one used previously? In that case, the authenticating gateway, say, GW_{dst}, will receive a fragment of the packet, detect that fragmentation has taken place and inform the source host of the new maximum packet size. The unlucky fragmented packet is then discarded and will have to be retransmitted.

Analysis

The criteria for evaluating the above methods include:

- Increased packet size

  Packet size will only be increased in the first scheme if an IP option is introduced that will hold the signature of the original packet. Also if the length of each packet is to be present in all of its fragments, the overall data length will increase. The total increase will be on the order of 4 to 8 bytes per packet fragment, depending on the signature length.

- Control packet overhead

  Control packets need only be exchanged in the second scheme, i.e, the packets that inform the host of a "safe" packet size whenever a route changes and the minimum MTU decreases. Only one packet of minimal (25-30 bytes) length is needed to inform the source host each time such change occurs.

- Data packet overhead (in number of packets sent)

  As far as the overall number of packets generated, the second scheme is clearly inferior. This is because packet sizes are limited by the minimum MTU of all transit networks up to the

last authenticating gateway. In addition, sometimes packets will have to be retransmitted due to routing changes and decreased minimum MTU.

- Overhead in hosts and gateways

  Overhead is mainly due to the state information and processing time in hosts and gateways. The first scheme doesn't affect the hosts - thus, the only overhead is in gateways. However, this overhead is significant since state information is maintained on per-packet basis. It is safe to assume that at any one time the number of packets in 'transit'[3] through a gateway can be quite large.

  In the second scheme, the overhead is minimal, save for the fact that hosts must maintain state information on a per-connection basis.

- Complexity of implementation

  Both schemes are quite trivial to implement. If the number of authenticating gateways is small, as compared to the number of communicating hosts, the first scheme maybe more desirable since it only requires gateway code to modified. Alternatively, if the number of hosts is small, second scheme is preferable as the changes to protocol software are minimal.

  The preferred scheme is likely to be contextually determined. By way of example, suppose that majority of communication is via remote login. Since packets tend to be very short no fragmentation will occur and the second scheme is probably a better a choice. Alternatively, if file transfer is the dominant application packets tend to be fewer in number and larger in size. In that case, the first scheme will perform better since it allows fragmentation to take place.

Conclusions

We have described two methods that allow fragmentation and dynamic routing to coexist with gateway-level authentication in datagram networks. The first adapts do changing paths and fragmentation by keeping state information on a per packet basis, while the second restricts fragmentation but incurs no additional state overhead. The two methods vary in implementation complexity, overhead and number of extra packets sent. The two schemes are applicable under different conditions and since they are not mutually exclusive, both can be incorporated and used depending on the nature of communication.

Acknowledgements

The author wishes to thank Deborah Estrin, Kim Korner and Jeffrey Mogul for many insightful comments and discussions.

3 By 'transit' packet we mean a packet some (but not all) fragments of which have passed through the gateway, i.e., a packet for which an entry is maintained in the gateway's table

Appendix.

Maintaining state information in the first scheme

As indicated above, it is safe to assume that the number of packets for which a gateway may have to keep state information at any one time can be quite large. Thus, it is necessary to minimize the memory requirements of the data structures used to maintain state information on a per packet basis.

The following format is suggested:

#	*Field*	*Length*
1	Packet ID	32 bits
2	SRC address	32 bits
3	DST address[4]	32 bits
4	Data length	16 bits
5	Composite F_{char}	32-64 bits
6	Partial F_{char}	32-64 bits
7	List of "holes"	Variable

Fields (1) through (4) amount to 112 bits. Fields (5) and (6) sum up to 64-128 bits depending on the characteristic function(F_{char}) used. Field (7) is a list of "holes", i.e, intervals for which more packet fragments are expected. For each "hole" only a high and low offsets need to be kept; each one is 32 bits long. This field is of variable length in multiples of 64 bits. If packet fragments arrive in order, which is frequently the case, only one "hole" needs to be kept. Thus, the least amount of memory utilized per "transit" packet is 240 bits (30 bytes) - a rather insignificant amount, if the number of packets in "transit" is reasonably small, say, in hundreds of packets.

---

4 If a gateway is on the same network with either source or destination host, it is possible to cut the size of either SOURCE or DESTINATION address down to 8 bits, since the rest can be constructed from the gateway's own interface address.

Bibliography

[1] Deborah Estrin, *Inter-Organizational Networks: Implications of Access Control Requirements for Interconnection Protocols,* Proceedings of ACM SIGCOMM 1986 (August 1986) pp. 254-264.

[2] Deborah Estrin and Gene Tsudik, *VISA Scheme for Inter-Organization Network Security,* Proceedings of IEEE Symposium on Security and Privacy 1987 (April 1987) pp. 254-264.

[3] Christopher Kent and Jeffrey Mogul, *Fragmentation Considered Harmful,* Proceedings of ACM SIGCOMM 1987 (August 1987).

[4] Jon Postel, *Internet Protocol,* RFC 791 (SRI NIC, September 1981).

[5] Jon Postel, *Internet Control Message Protocol,* RFC 792 (SRI NIC, September 1981).

[6] Carl Sunshine, *Source Routing in Computer Networks,* Computer Communications Review **1, No.7** (January 1977) pp. 29-33.

[7] David Clark, *IP Datagram Reassembly Algorithms,* RFC 815 (SRI NIC, July 1982).

[8] Deborah Estrin, Jeffrey Mogul, Gene Tsudik, *Visa Protocols For Controlling Inter-Organizational Datagram Flow,* In submission to IEEE JSAC (Draft of January 88).