              Analysis of Potential Solutions for Revealing a
            Host Identifier (HOST_ID) in Shared Address Deployments

Abstract

   This document is a collection of potential solutions for revealing a
   host identifier (denoted as HOST_ID) when a Carrier Grade NAT (CGN)
   or application proxies are involved in the path. This host
   identifier could be used by a remote server to sort packets according
   to the sending host. The host identifier must be unique to each host
   under the same shared IP address.

   This document analyzes a set of potential solutions for revealing a
   host identifier and does not recommend a particular solution,
   although it does highlight the hazards of some approaches.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   As reported in [RFC6269], several issues are encountered when an IP
   address is shared among several subscribers. These issues are
   encountered in various deployment contexts, e.g., Carrier-Grade NAT
   (CGN), application proxies, or Address plus Port (A+P) [RFC6346].
   Examples of such issues are: implicit identification (Section 13.2 of
   [RFC6269]), spam (Section 13.3 of [RFC6269]), blacklisting a
   misbehaving host (Section 13.1 of [RFC6269]), or redirecting users
   with infected machines to a dedicated portal (Section 5.1 of
   [RFC6269]).

   In particular, some servers use the source IPv4 address as an
   identifier to treat some incoming connections differently. Due to
   the deployment of CGNs (e.g., NAT44 [RFC3022], NAT64 [RFC6146]), that
   address will be shared. In particular, when a server receives
   packets from the same source address, because this address is shared,
   the server does not know which host is the sending host [RFC6269].
   The sole use of the IPv4 address is not sufficient to uniquely
   distinguish a host. As a mitigation, it is tempting to investigate
   ways that would disclose information to be used by the remote server
   as a means of uniquely disambiguating packets sent from hosts using
   the same IPv4 address.

   The risk of not mitigating these issues include: OPEX (Operational
   Expenditure) increase for IP connectivity service providers (costs
   induced by calls to a hotline), revenue loss for content providers
   (loss of users' audience), and customers' dissatisfaction (low
   quality of experience, service segregation, etc.).

   The purpose of this document is to analyze a set of alternative
   channels to convey a host identifier and to assess to what extent the
   alternatives solve the problem described in Section 2. The
   evaluation is intended to be comprehensive, regardless of the
   maturity or validity of any currently known or proposed solution.
   The alternatives analyzed in the document are listed below:

   o Use the Identification field of the IP header (denoted as IP-ID,
      Section 4.1).
   o Define a new IP option (Section 4.2).
   o Define a new TCP option (Section 4.3).
   o Inject application headers (Section 4.4).
   o Enable Proxy Protocol (Section 4.5).
   o Assign port sets (Section 4.6).
   o Activate HIP (Host Identity Protocol) (Section 4.7).
   o Use a notification channel (Section 4.8).
   o Use an out-of-band mechanism (Section 4.9).

A synthesis is provided in Section 5, while the detailed analysis is elaborated in Section 4.

Section 3 discusses privacy issues common to all proposed solutions. It is out of scope of this document to elaborate on privacy issues specific to each solution.

This document does not include any recommendations because the working group felt that it was too premature to include one.

2. On HOST_ID

Policies that rely on source IP addresses and that are enforced by some servers will be applied to all hosts sharing the same IP address. For example, blacklisting the IP address of a spammer host will result in all other hosts that share that address having their access to the requested service restricted. [RFC6269] describes the issues in detail. Therefore, due to address sharing, servers need extra information beyond the source IP address to differentiate the sending host. We call this information the HOST_ID.

The HOST_ID identifies a host under a shared IP address. Privacy-related considerations are discussed in Section 3.

Within this document, a host can be any computer located behind a Home Gateway or directly connected to an address-sharing function located in the network provider's domain (typically this would be the Home Gateway itself).

Because the HOST_ID is used by a remote server to sort out the packets by sending host, the HOST_ID must be unique to each host under the same shared IP address, where possible. In the case where only the Home Gateway is revealed to the operator side of the translation function, the HOST_ID need only be unique to the Home Gateway. The HOST_ID does not need to be globally unique. Of course, the combination of the (public) IP source address and the identifier (i.e., HOST_ID) ends up being unique.

If the HOST_ID is conveyed at the IP level, all packets will have to bear the identifier. If it is conveyed at a higher connection-oriented level, the identifier is only needed once in the session establishment phase (for instance, a TCP three-way handshake), then all packets received in this session will be attributed to the HOST_ID designated during the session opening.

Within this document, we assume the operator-side address-sharing function injects the HOST_ID. Another deployment option to avoid potential performance degradation is to let the host or Home Gateway

   inject its HOST_ID, but the address-sharing function will check its
   content (just like an IP anti-spoofing function). For some
   proposals, the HOST_ID is retrieved using an out-of-band mechanism or
   signaled in a dedicated notification channel.

   For A+P [RFC6346] and its variants, port set announcements may be
   needed as discussed in Section 4.6.

   Security considerations are common to all analyzed solutions (see
   Section 6). Privacy-related aspects are discussed in Section 3.

   The HOST_ID can be ambiguous for hosts with multiple interfaces or
   multiple addresses assigned to a single interface. HOST_IDs that are
   the same may be used to imply or infer the same end system, but
   HOST_IDs that are different should not be used to imply or infer
   whether the end systems are the same or different.

3. HOST_ID and Privacy

   IP address sharing is motivated by a number of different factors.
   For years, many network operators have conserved public IPv4
   addresses by making use of Customer Premises Equipment (CPE) that
   assigns a single public IPv4 address to all hosts within the
   customer's local area network and uses NAT [RFC3022] to translate
   between locally unique private IPv4 addresses and the CPE's public
   address. With the exhaustion of IPv4 address space, address sharing
   between customers on a much larger scale is likely to become much
   more prevalent. While many individual users are unaware of and
   uninvolved in decisions about whether their unique IPv4 addresses get
   revealed when they send data via IP, some users realize privacy
   benefits associated with IP address sharing, and some may even take
   steps to ensure that NAT functionality sits between them and the
   public Internet. IP address sharing makes the actions of all users
   behind the NAT function unattributable to any single host, creating
   room for abuse but also providing some identity protection for
   non-abusive users who wish to transmit data with reduced risk of
   being uniquely identified.

   The proposals considered in this document help differentiate between
   hosts that share a public IP address. The extent of that
   differentiation depends on what information is included in the
   HOST_ID.

   The volatility of the HOST_ID information is similar to that of the
   internal IP address: a distinct HOST_ID may be used by the address-
   sharing function when the host reboots or gets a new internal IP
   address. As with persistent IP addresses, persistent HOST_IDs
   facilitate user tracking over time.

As a general matter, the HOST_ID proposals do not seek to make hosts
any more identifiable than they would be if they were using a public,
non-shared IP address. However, depending on the solution proposal,
the addition of HOST_ID information may allow a device to be
fingerprinted more easily than it otherwise would be. To prevent
this, the following design considerations are to be taken into
account:

o It is recommended that HOST_IDs be limited to providing local
   uniqueness rather than global uniqueness.

o The address-sharing function should not use permanent HOST_ID
   values.

Should multiple solutions be combined (e.g., TCP option and Forwarded
header) that include different pieces of information in the HOST_ID,
fingerprinting may become even easier. To prevent this, an address-
sharing function that is able to inject HOST_IDs in several layers
should reveal the same subsets of information at each layer. For
example, if one layer references the lower 16 bits of an IPv4
address, the other layer should reference these 16 bits too.

A HOST_ID can be spoofed, as this is also the case for spoofing an IP
address. Furthermore, users of network-based anonymity services
(like Tor [TOR]) may be capable of stripping HOST_ID information
before it reaches its destination.

In order to control the information revealed to external parties, an
address-sharing function should be able to strip, rewrite, and add
HOST_ID fields.

An address-sharing function may be configured to enforce different
end-user preferences with regards to HOST_ID injection. For example,
HOST_ID injection can be disabled for some users. This feature is
policy based and deployment specific.

HOST_ID specification document(s) should explain the privacy impact
of the solutions they specify, including the extent of HOST_ID
uniqueness and persistence, assumptions made about the lifetime of
the HOST_ID, whether and how the HOST_ID can be obfuscated or
recycled, whether location information can be exposed, and the impact
of the use of the HOST_ID on device or implementation fingerprinting.
[IAB-PRIVACY] provides further guidance.

For more discussion about privacy, refer to [RFC6462].

4. Detailed Solutions Analysis

4.1. Use the Identification Field of the IPv4 Header (IP-ID)

4.1.1. Description

The IPv4 ID (Identification field of IP header, i.e., IP-ID) can be
used to insert information that uniquely distinguishes a host among
those sharing the same IPv4 address. Use of the IP-ID as a channel
to convey the HOST_ID is a theoretical construct (i.e., it is an
undocumented proposal).

An address-sharing function can rewrite the IP-ID field to insert a
value that is unique to the host (16 bits are sufficient to uniquely
disambiguate hosts sharing the same IP address). The address-sharing
function injecting the HOST_ID must follow the rules defined in
[RFC6864]; in particular, the same HOST_ID is not reassigned to
another host sharing the same IP address during a given time
interval.

A variant of this approach relies upon the format of certain packets,
such as TCP SYN, where the IP-ID can be modified to contain a 16-bit
HOST_ID.

Address-sharing devices using this solution would be required to
indicate that they do so, possibly using a special DNS record.

4.1.2. Analysis

This usage is not consistent with the fragment reassembly use of the
Identification field [RFC0791] or the updated handling rules for the
Identification field [RFC6864].

Complications may arise if the packet is fragmented before reaching
the device that is injecting the HOST_ID. To appropriately handle
those packet fragments, the address-sharing function will need to
maintain a lot of state.

Another complication to be encountered is where translation is
balanced among several NATs; setting the appropriate HOST_ID by a
given NAT would alter the coordination between those NATs. Of
course, one can argue that this coordinated NAT scenario is not a
typical deployment scenario; regardless, using the IP-ID as a channel
to convey a HOST_ID is ill-advised.

4.2. Define an IP Option

4.2.1. Description

   An alternate way to convey the HOST_ID is to define an IP option
   [RFC0791]. A HOST_ID IP option can be inserted by the address-
   sharing function to uniquely distinguish a host among those sharing
   the same IP address. An example of such an option is documented in
   [REVEAL-IP]. This IP option allows the conveyance of an IPv4
   address, an IPv6 prefix, a Generic Routing Encapsulation (GRE) key,
   an IPv6 Flow Label, etc.

   An IP option may also be used as described in Section 4.6 of
   [RFC3022].

4.2.2. Analysis

   This proposal can apply to any transport protocol. However, it is
   widely known that routers and other middleboxes filter IP options
   (e.g., drop IP packets with unknown IP options, strip unknown IP
   options, etc.).

   Injecting the HOST_ID IP option introduces some implementation
   complexity in the following cases:

   o The packet is at or close to the MTU size.

   o The options space is exhausted.

   Previous studies demonstrated that "IP Options are not an option"
   (refer to [Not_An_Option] and [Options]).

   In conclusion, using an IP option to convey a HOST_ID is not viable.

4.3. Define a TCP Option

4.3.1. Description

   The HOST_ID may be conveyed in a dedicated TCP option. An example is
   specified in [REVEAL-TCP]. This option encloses the TCP client's
   identifier (e.g., the lower 16 bits of its IPv4 address, its VLAN ID,
   VRF ID, or subscriber ID). The address-sharing device inserts this
   TCP option into the TCP SYN packet.

4.3.2. Analysis

   Using a new TCP option to convey the HOST_ID does not require any
   modification to the applications, but it is applicable only for
   TCP-based applications. Applications relying on other transport
   protocols are therefore left unsolved.

   [REVEAL-TCP] discusses the interference with other TCP options.

   The risk of session failure due to handling a new TCP option is low
   as measured in [Options]. [REVEAL-TCP-EXP] provides a detailed
   implementation and experimentation report of a HOST_ID TCP option.
   This document provides an in-depth investigation of the impact of
   implementing HOST_ID on the host, the address-sharing function, and
   the enforcement of policies at the server side. It also reports a
   failure ratio of 0.103% among the top 100,000 websites.

   Some downsides have been identified with defining a TCP option to
   reveal a host identity:

   o Conveying an IP address in a TCP option may be seen as a violation
      of OSI layers, but since IP addresses are already used for the
      checksum computation, this is not seen as a blocking point.
      Moreover, the updated version of [REVEAL-TCP] no longer allows
      conveyance of a full IP address because the HOST_ID is encoded in
      16 bits.

   o TCP option space is limited and might be consumed by the TCP
      client. [REVEAL-TCP-EXP] discusses two approaches to sending the
      HOST_ID: sending the HOST_ID in the TCP SYN (which consumes more
      bytes in the TCP header of the TCP SYN) and sending the HOST_ID in
      a TCP ACK (which consumes only two bytes in the TCP SYN).

   o Content providers may find it more desirable to receive the
      HOST_ID in the TCP SYN, as that more closely preserves the HOST_ID
      received in the source IP address as per current practices.
      Moreover, sending the HOST_ID in the TCP SYN does not interfere
      with [FASTOPEN]. In the ACK mode, if the server is configured to
      deliver different data based on HOST_ID, then it would have to
      wait for the ACK before transmitting data.

   o HOST_ID mechanisms need to be aware of end-to-end (E2E) issues and
      avoid interfering with them. One example of such interference
      would be injecting or removing TCP options of transited packets;
      another such interference involves terminating and re-originating
      TCP connections not belonging to the transit device. The HOST_ID
      TCP option handled by the source node avoids this issue.

   o Injecting the HOST_ID TCP option introduces some implementation
      complexity if the options space is exhausted. Specification
      document(s) should specify the behavior of the address-sharing
      function in detail in such a case.

   o It is more complicated to implement sending the HOST_ID in a TCP
      ACK, as it can introduce MTU issues if the ACK packet also
      contains TCP data or if a TCP segment is lost. Note that MTU
      complications can be experienced if user data is included in a SYN
      packet (e.g., [FASTOPEN]).

   o When there are several NATs in the path, the original HOST_ID may
      be lost. The loss of the original HOST_ID may not be a problem,
      as the target usage is between proxies or between a CGN and
      server. Only the information leaked in the last communication leg
      (i.e., between the last address-sharing function and the server)
      is likely to be useful.

   o Interference with usages such as a Forwarded HTTP header (see
      Section 4.4) should be elaborated to specify the behavior of
      servers when both options are used; in particular, specify which
      information to use: the content of the TCP option or what is
      conveyed in the application headers.

   o When load balancers or proxies are in the path, this option does
      not allow the preservation of the original source IP address and
      source port. Preserving such information is required for logging
      purposes (e.g., [RFC6302]). [REVEAL-TCP-EXP] defines a TCP option
      that allows various combinations of source information (e.g.,
      source port, source port and source IP address, source IPv6
      prefix, etc.) to be revealed.

   More discussion about issues raised when extending TCP can be found
   at [ExtendTCP].

4.4. Inject Application Protocol Message Headers

4.4.1. Description

   Another option is to not require any change within the transport or
   the IP levels but to convey the required information that will be
   used to disambiguate hosts at the application payload. The format of
   the conveyed information and the related semantics depend on its
   application (e.g., HTTP, SIP, SMTP, etc.).

   Related mechanisms could be developed for other application-layer
   protocols, but the discussion in this document is limited to HTTP and
   similar protocols.

For HTTP, the Forwarded header [HTTP-FRWD] can be used to display the
original IP address when an address-sharing device is involved.
Service providers operating address-sharing devices can enable the
feature of injecting the Forwarded header, which will enclose the
original IPv4 address or the IPv6 prefix part (see the example shown
in Figure 1). The address-sharing device has to strip all included
Forwarded headers before injecting its own. Servers may rely on the
contents of this field to enforce some policies such as blacklisting
misbehaving users.

Note that [HTTP-FRWD] standardizes the Forwarded header field, to
replace the de facto (and not standard) X-Forwarded-For (XFF) header.

```
                 Forwarded: for=192.0.2.1,for=[2001:db8::1]
                 Forwarded: proto=https;by=192.0.2.15
```

                     Figure 1: Example of Forwarded-For

4.4.2. Analysis

Not all applications impacted by address sharing can support the
ability to disclose the original IP address. Only a subset of
protocols (e.g., HTTP) can rely on this solution.

For the HTTP case, to prevent users from injecting invalid HOST_IDs,
an initiative has been launched by Wikimedia to maintain a list of
trusted ISPs (Internet Service Providers) using XFF (see the list
available at [Trusted_ISPs]). If an address-sharing device is on the
list of trusted XFF ISPs, users editing Wikimedia located behind the
address-sharing device will appear to be editing from their
"original" IP address and not from the NATed IP address. If an
offending activity is detected, individual hosts can be blacklisted
instead of blacklisting all hosts sharing the same IP address.

XFF header injection is a common practice of load balancers. When a
load balancer is in the path, the original content of any included
XFF header should not be stripped. Otherwise, the information about
the "origin" IP address will be lost.

When several address-sharing devices are crossed, the Forwarded
header can convey the list of IP addresses (e.g., Figure 1). The
origin HOST_ID can be exposed to the target server.

Injecting the Forwarded header also introduces some implementation
complexity if the HTTP message is at or close to the MTU size.

It has been reported that some HTTP proxy implementations may
encounter parsing issues when injecting an XFF header.

Injecting the Forwarded header for all HTTPS traffic is infeasible.
This may be problematic given the current HTTPS usage trends.

4.5. PROXY Protocol

4.5.1. Description

The solution, referred to as the Proxy Protocol [Proxy], does not
require any application-specific knowledge. The proposed solution
(Proxy Protocol Version 1) would insert identification data directly
into the application-data stream prior to the actual protocol data
being sent, regardless of the protocol. Every application protocol
would begin with a textual string of "PROXY", followed by some
textual identification data, and with a CRLF; only then would the
application data be inserted. Figure 2 shows an example of a line of
data used for this purpose, in this case, for a TCP-over-IPv4
connection received from 192.0.2.1:56324 and destined to
192.0.2.15:443.

         PROXY TCP4 192.0.2.1 192.0.2.15 56324 443\r\n

              Figure 2: Example of PROXY Connection Report

Upon receipt of a message conveying this line, the server removes the
line from the incoming stream. The line is parsed to retrieve the
transported protocol. The content of this line is recorded in logs
and used to enforce policies.

Proxy Protocol Version 2 is designed to accommodate IPv4/IPv6 and
also non-TCP protocols (see [Proxy] for more details).

4.5.2. Analysis

This solution can be deployed in a controlled environment, but it
cannot be deployed to all access services available in the Internet.
If the remote server does not support the Proxy Protocol, the session
will fail. Other complications will arise due to the presence of
firewalls, for instance.

As a consequence, this solution is infeasible and cannot be
recommended.

4.6. Assign Port Sets

4.6.1. Description

This solution does not require any action from the address-sharing
function to disclose a host identifier. Instead of assuming that all
transport ports are associated with one single host, each host under
the same external IP address is assigned a restricted port set.
These port sets are then advertised to remote servers using offline
means. This announcement is not required for the delivery of
internal services (i.e., offered by the service provider deploying
the address-sharing function) relying on implicit identification.

Port sets assigned to hosts may be static or dynamic.

Port set announcements to remote servers are not required to reveal
the identity of individual hosts; they are used to advertise the
enforced policy to generate non-overlapping port sets (e.g., the
transport space associated with an IP address is fragmented to
contiguous blocks of 2048 port numbers).

Examples of such an approach are documented in [RFC6346] and
[DETERMCGN].

4.6.2. Analysis

The solution does not require defining new fields or options; it is
policy based.

The solution may contradict the port randomization [RFC6056] as
identified in [RFC6269]. A mitigation would be to avoid assigning
static port sets to individual hosts.

The method is convenient for the delivery of services offered by the
service provider that is also offering the Internet access service.

4.7. Host Identity Protocol (HIP)

4.7.1. Description

[RFC5201] specifies an architecture that introduces a new namespace
to convey identity information.

4.7.2. Analysis

This solution requires both the client and the server to support HIP
[RFC5201]. Additional architectural considerations are to be taken
into account, such as the key exchanges.

An alternative deployment model that does not require the client to
be HIP-enabled is having the address-sharing function behave as a
UDP/TCP-HIP relay. This model is also not viable as it assumes all
servers are HIP-enabled.

This solution is a theoretical construct (i.e., the proposal is not
documented).

4.8. Use of a Notification Channel (e.g., ICMP)

4.8.1. Description

Another alternative is to convey the HOST_ID using a separate
notification channel than the one the packets issued to invoke the
service.

This solution relies on a mechanism where the address-sharing
function encapsulates the necessary host-identifying information into
an ICMP Echo Request packet that it sends in parallel with the
initial session creation (e.g., SYN). The information included in
the ICMP Request Data portion describes the five-tuples as seen on
both sides of the address-sharing function. An implementation
example is defined in [REVEAL-ICMP].

4.8.2. Analysis

o This ICMP proposal is valid for any transport protocol that uses a
   port number. The address-sharing function may be configured with
   the transport protocols that will trigger issuing those ICMP
   messages.

o A hint should be provided to the ultimate server (or intermediate
   nodes) that the ICMP Echo Request conveys a HOST_ID. This may be
   implemented using magic numbers (a magic number is a self-selected
   codepoint whose primary value is its unlikely collision with
   values selected by others).

o Even if ICMP packets are blocked in the communication path, the
   user connection does not have to be impacted.

o Implementations requiring a session establishment to be delayed
   until receipt of the companion ICMP Echo Request may lead to some
   user-experience degradation.

o Because of the presence of load balancers in the path, the
   ultimate server receiving the SYN packet may not be the one that
   receives the ICMP message conveying the HOST_ID.

   o Because of the presence of load balancers in the path, the port
      number assigned by address sharing may be lost. Therefore, the
      mapping information conveyed in the ICMP may not be sufficient to
      associate a SYN packet with a received ICMP.

   o The proposal is not compatible with the presence of cascaded NAT.
      The main reason is that each NAT in the path will generate an ICMP
      message to reveal the internal host identifier. Because these
      messages will be translated by the downstream address-sharing
      devices, the remote server will receive multiple ICMP messages and
      will need to decide which host identifier to use.

   o The ICMP proposal will add traffic overhead for both the server
      and the address-sharing device.

   o The ICMP proposal is similar to other mechanisms (e.g., IPFIX
      [IPFIX-NAT] and Syslog [SYSLOG-NAT]) for reporting dynamic
      mappings to a mediation platform (mainly for legal traceability
      purposes). Performance degradation is likely to be experienced by
      address-sharing functions because ICMP messages are sent for each
      new instantiated mapping (even if the mapping exists).

   o In some scenarios (e.g., Section 3 of [REVEAL-PCP]), the HOST_ID
      should be interpreted by intermediate devices that embed Policy
      Enforcement Points (PEP) [RFC2753] responsible for granting access
      to some services. These PEPs need to inspect all received packets
      in order to find the companion (traffic) messages to be correlated
      with ICMP messages conveying HOST_IDs. This induces more
      complexity to these intermediate devices.

4.9. Use Out-of-Band Mechanisms (e.g., Ident)

4.9.1. Description

   Another alternative is to retrieve the HOST_ID using a dedicated
   query channel.

   An implementation example may rely on the Identification Protocol
   (Ident) [RFC1413]. This solution assumes that the address-sharing
   function implements the server part of IDENT, while remote servers
   implement the client part of the protocol. IDENT needs to be updated
   to be able to return a host identifier instead of the user-id as
   defined in [RFC1413]. The IDENT response syntax uses the same USERID
   field described in [RFC1413], but rather than returning a username, a
   host identifier (e.g., a 16-bit value) is returned. For any new
   incoming connection, the server contacts the IDENT server to retrieve
   the associated identifier. During that phase, the connection may be
   delayed.

4.9.2. Analysis

   o IDENT is specific to TCP. Alternative out-of-band mechanisms may
      be designed to cover other transport protocols such as UDP.

   o This solution requires the address-sharing function to embed an
      IDENT server.

   o A hint should be provided to the ultimate server (or intermediate
      nodes) that the address-sharing function implements the IDENT
      protocol, for example, publishing this capability using DNS (other
      solutions can be envisaged).

   o An out-of-band mechanism may require some administrative setup
      (e.g., contract agreement) between the entity managing the
      address-sharing function and the entity managing the remote
      server. Such a deployment is not feasible in the Internet at
      large because establishing and maintaining agreements between ISPs
      and all service actors is burdensome and not scalable.

   o Implementations requiring delay of the establishment of a session
      until receipt of the companion IDENT response may lead to some
      user-experience degradation.

   o The IDENT proposal will add traffic overhead for both the server
      and the address-sharing device.

   o Performance degradation is likely to be experienced by address-
      sharing functions embedding the IDENT server. This is further
      exacerbated if the address-sharing function has to handle an IDENT
      query for each new instantiated mapping (even if the mapping
      exists).

   o In some scenarios (e.g., Section 3 of [REVEAL-PCP]), the HOST_ID
      should be interpreted by intermediate devices that embed Policy
      Enforcement Points (PEP) [RFC2753] responsible for granting access
      to some services. These PEPs need to inspect all received packets
      in order to generate the companion IDENT queries. This may induce
      more complexity to these intermediate devices.

   o IDENT queries may be generated by illegitimate TCP servers. This
      would require the address-sharing function to enforce some
      policies (e.g., rate-limit queries, filter based on the source IP
      address, etc.).

5. Solutions Analysis: Synthesis

   Table 1 summarizes the approaches analyzed in this document.

	IP-ID	IP Option	TCP Option	HTTP Header	PROXY	Port Set	HIP	ICMP	IDENT
UDP	Yes	Yes	No	No	No	Yes		Yes	No
TCP	Yes	Yes	Yes	No	Yes	Yes		Yes	Yes
HTTP	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes
Encrypted Traffic	Yes	Yes	Yes	No	Yes	Yes		Yes	Yes
Success Ratio	High	Low	High	High	Low	100%	Low	High	High
Possible Perf Impact	Low to Med	High	Low to Med	Med to High	High	No	N/A	High	High
OS TCP/IP Modif	Yes	Yes	Yes	No	No	No		Yes	Yes
Deployable Today	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes
Notes	(1) (7)	(8)	(8)	(2)	(8)	(1) (3)	(4) (7)	(6) (8)	(1) (6) (8)

                  Table 1: Summary of Analyzed Solutions

   o "Encrypted Traffic" refers to TLS. The use of IPsec and its
      complications traversing NATs are discussed in Section 2.2 of
      [RFC6889]. Similar to what is suggested in Section 13.5 of
      [RFC6269], HOST_ID specification document(s) should analyze the
      compatibility of each IPsec mode in detail.

   o "Success ratio" indicates the ratio of successful communications
      with remote servers when the HOST_ID is injected using a proposed
      solution. More details are provided below to explain how the
      success ratio is computed for each proposed solution.

   o "Possible Perf Impact" indicates the level of expected performance
      degradation. The indicated degradation is an estimate based on
      the need for processing at the IP layer.

   o "OS TCP/IP Modif" indicates whether a modification of the OS
      TCP/IP stack is required at the server side.

   o "Deployable today" indicates if the solution can be generalized
      without any constraint on current architectures and practices.

   Notes:
   (1) Requires mechanism to advertise that NAT is participating in
        this scheme (e.g., DNS PTR record).
   (2) This solution is widely deployed (e.g., HTTP severs, load
        balancers, etc.).
   (3) When the port set is not advertised, the solution is less
        efficient for third-party services.
   (4) Requires that the client and the server to be HIP-compliant and
        that HIP infrastructure be deployed. If the client and the
        server are HIP-enabled, the address-sharing function does not
        need to insert an identifier. If the client is not HIP-enabled,
        designing the device that performs address sharing to act as a
        UDP/TCP-HIP relay is not viable.
   (6) The solution is inefficient in some scenarios (see Section 5).
   (7) The solution is a theoretical construct (i.e., the solution is
        not documented).
   (8) The solution is a documented proposal.

   Provided success ratio figures for TCP and IP options are based on
   the results documented in [Options] and [REVEAL-TCP-EXP].

   The provided success ratio for the IP-ID is theoretical; it assumes
   the address-sharing function follows the rules (see [RFC6864])to
   rewrite the IP Identification field.

   Since PROXY and HIP are not widely deployed, the success ratio for
   establishing communication with remote servers using these protocols
   is low.

   The success ratio for the ICMP-based solution is implementation
   specific, but it is likely to be close to 100%. The success ratio
   depends on how efficiently the solution is implemented on the server
   side. A remote server that does not support the ICMP-based solution
   will ignore received companion ICMP messages. An upgraded server
   will need to delay the acceptance of a session until it receives the
   companion ICMP message.

The success ratio for the IDENT solution is implementation specific
but it is likely to be close to 100%. The success ratio depends on
how efficient the solution is implemented on the server side. A
remote server that does not support IDENT will accept a session
establishment request following its normal operation. An upgraded
server will need to delay the acceptance of a session until it
receives a response to the IDENT request it will send to the host.

The provided success ratio for the Port Set and HTTP header solutions
is 100% because no additional Layer 3 or Layer 4 field is needed to
convey HOST_ID for these solutions.

6. Security Considerations

If the server trusts the content of the HOST_ID field, a third-party
user can be impacted by a misbehaving user revealing a "faked"
HOST_ID (e.g., original IP address). This same security concern
applies for the injection of an IP option, TCP option, and
application-related content (e.g., the Forwarded HTTP header) by the
address-sharing device.

The HOST_ID may be used to leak information about the internal
structure of a network behind an address-sharing function. If this
behavior is undesired for the network administrator, the address-
sharing function can be configured to strip any existing HOST_ID in
received packets from internal hosts.

HOST_ID specification documents should elaborate further on threats
inherent to each individual solution used to convey the HOST_ID
(e.g., use of the IP-ID field to count hosts behind a NAT [Count]).

For more discussion of privacy issues related to the HOST_ID, see
Section 3.

7. Acknowledgments

Many thanks to D. Wing, C. Jacquenet, J. Halpern, B. Haberman, and
P. Yee for their review, comments, and inputs.

Thanks also to P McCann, T. Tsou, Z. Dong, B. Briscoe, T. Taylor, M.
Blanchet, D. Wing, and A. Yourtchenko for the discussions in Prague.

Some of the issues related to defining a new TCP option have been
raised by L. Eggert.

The privacy text was provided by A. Cooper.

8. References

8.1. Normative References

[RFC0791]   Postel, J., "Internet Protocol", STD 5, RFC 791,
            September 1981.

[RFC3022]   Srisuresh, P. and K. Egevang, "Traditional IP Network
            Address Translator (Traditional NAT)", RFC 3022, January
            2001.

[RFC6056]   Larsen, M. and F. Gont, "Recommendations for Transport-
            Protocol Port Randomization", BCP 156, RFC 6056, January
            2011.

8.2. Informative References

[Count]   Belloven, S., "A Technique for Counting NATted Hosts",
            <http://www.cs.columbia.edu/~smb/papers/fnat.pdf>.

[DETERMCGN] Donley, C., Grundemann, C., Sarawat, V., Sundaresan, K.,
            and O. Vautrin, "Deterministic Address Mapping to Reduce
            Logging in Carrier Grade NAT Deployments", Work in
            Progress, January 2013.

[ExtendTCP] Honda, M., Nishida, Y., Raiciu, C., Greenhalgh, A.,
            Handley, M. and H. Tokuda,, "Is It Still Possible to
            Extend TCP?", November 2011,
            <http://nrg.cs.ucl.ac.uk/mjh/tmp/mboxes.pdf>.

[FASTOPEN] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP
            Fast Open", Work in Progress, February 2013.

[HTTP-FRWD] Petersson, A. and M. Nilsson, "Forwarded HTTP
            Extension", Work in Progress, October 2012.

[IAB-PRIVACY]
            Cooper, A., Tschofenig, H., Aboba, B., Peterson, J.,
            Morris, J., Hansen, M., and R. Smith, "Privacy
            Considerations for Internet Protocols", Work in
            Progress, July 2012.

[IPFIX-NAT] Sivakumar, S. and R. Penno, "IPFIX Information Elements
            for Logging NAT Events", Work in Progress, March 2013.

[Not_An_Option]
          R. Fonseca, G. Porter, R. Katz, S. Shenker, and I.
          Stoica,, "IP Options Are Not An Option", 2005,
          <http://www.eecs.berkeley.edu/Pubs/TechRpts/2005/
          EECS-2005-24.html>.

[Options]   Medina, A, Allman, M. and S. Floyd, "Measuring
          Interactions Between Transport Protocols and
          Middleboxes", 2005,
          <http://conferences.sigcomm.org/imc/2004/papers/
          p336-medina.pdf>.

[Proxy]   Tarreau, W., "The PROXY protocol", November 2010,
          <http://haproxy.1wt.eu/download/1.5/doc/
          proxy-protocol.txt>.

[REVEAL-ICMP]
          Yourtchenko, A., "Revealing Hosts Sharing An IP Address
          Using ICMP Echo Request", Work in Progress, March 2012.

[REVEAL-IP] Wu, Y., Ji, H., Chen, Q., and T. ZOU), "IPv4 Header
          Option For User Identification In CGN Scenario", Work in
          Progress, March 2011.

[REVEAL-PCP] Boucadair, M., Reddy, T., Patil, P., and D. Wing, "Using
          PCP to Reveal a Host behind NAT", Work in Progress,
          November 2012.

[REVEAL-TCP-EXP]
          Abdo, E., Boucadair, M., and J. Queiroz, "HOST_ID TCP
          Options: Implementation & Preliminary Test Results",
          Work in Progress, July 2012.

[REVEAL-TCP] Yourtchenko, A. and D. Wing, "Revealing Hosts Sharing An
          IP Address Using TCP Option", Work in Progress, December
          2011.

[RFC1413]   St. Johns, M., "Identification Protocol", RFC 1413,
          February 1993.

[RFC2753]   Yavatkar, R., Pendarakis, D., and R. Guerin, "A
          Framework for Policy-based Admission Control", RFC 2753,
          January 2000.

[RFC5201]   Moskowitz, R., Nikander, P., Jokela, P., and T.
          Henderson, "Host Identity Protocol", RFC 5201, April
          2008.

   [RFC6146]   Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful
                NAT64: Network Address and Protocol Translation from
                IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

   [RFC6269]   Ford, M., Boucadair, M., Durand, A., Levis, P., and P.
                Roberts, "Issues with IP Address Sharing", RFC 6269,
                June 2011.

   [RFC6302]   Durand, A., Gashinsky, I., Lee, D., and S. Sheppard,
                "Logging Recommendations for Internet-Facing Servers",
                BCP 162, RFC 6302, June 2011.

   [RFC6346]   Bush, R., "The Address plus Port (A+P) Approach to the
                IPv4 Address Shortage", RFC 6346, August 2011.

   [RFC6462]   Cooper, A., "Report from the Internet Privacy Workshop",
                RFC 6462, January 2012.

   [RFC6864]   Touch, J., "Updated Specification of the IPv4 ID Field",
                RFC 6864, February 2013.

   [RFC6889]   Penno, R., Saxena, T., Boucadair, M., and S. Sivakumar,
                "Analysis of Stateful 64 Translation", RFC 6889, April
                2013.

   [SYSLOG-NAT] Chen, Z., Zhou, C., Tsou, T., and T. Taylor, "Syslog
                Format for NAT Logging", Work in Progress, May 2013.

   [TOR]   Dingledine, R., Mathewson, N., and P. Syverson, "Tor:
                The secondgeneration onion router", In Proceedings of
                the 13th USENIX Security Symposium, August 2004.

   [Trusted_ISPs]
                Wikimedia, "Trusted XFF List", May 2013,
                <http://meta.wikimedia.org/w/
                index.php?title=XFF_project&oldid=5507690>.

Authors' Addresses

   Mohamed Boucadair
   France Telecom
   Rennes 35000
   France

   EMail: mohamed.boucadair@orange.com

   Joe Touch
   USC/ISI
   4676 Admiralty Way
   Marina del Rey, CA 90292-6695
   United States

   Phone: +1 (310) 448-9151
   EMail: touch@isi.edu

   Pierre Levis
   France Telecom
   Caen 14000
   France

   EMail: pierre.levis@orange.com

   Reinaldo Penno
   Cisco
   United States

   EMail: repenno@cisco.com