The ISP Column
A monthly column on things Internet

August 2014

Geoff Huston

Where is Metadata Anyway?

> *A police officer on his beat late at night sees a drunken man intently searching the ground near a lamppost and asks him the goal of his quest. The inebriate replies that he is looking for his car keys, and the officer helps for a few minutes without success then he asks whether the man is certain that he dropped the keys near the lamppost.*
>
> *"No," is the reply, "I lost the keys somewhere across the street." "Why look here?" asks the surprised and irritated officer. "The light is much better here," the intoxicated man responds with aplomb.*

I can't help but think that the situation in this rather old joke applies very precisely to the current Australian efforts to compel network operators, through some contemplated regulatory instrument, to record and retain network-collected data about their customers' online activities.

What I'd like to examine here the emerging picture that while networks, and network operators, make convenient targets for such surveillance efforts, the reality of today's IP network's are far more complex, and Internet networks are increasingly ignorant about what their customers do. The result is that its now quite common for Internet networks not to have the information that these agencies are after. Not only can moderately well-informed users hide their activities from their local network, but increasingly this has been taken out of the hands of users, as the applications we have on our smartphones, tablets and other devices are increasingly making use of the network in ways that are completely opaque to the network provider. Looking to the network and network operators for this stream of data about connected users and the IP addresses that they use is increasingly an exercise that appears to fall into the category of "security pantomime".

> "Security pantomime" is a term I've seen applied in a number of security-related exercises where, like a pantomime, there is the superficial appearance of security, whereas in fact it's a parody of the real thing, and it's obviously ineffectual in achieving it's supposed objectives.

Let's take a step back and consider the question: Why are some Internet users are apparently so concerned about measures to capture and retain network "metadata" on the Internet? It appears that they believe that the network is in a position to be privy to all our communications, both in terms of who we contact and what we do and say online, and were this information to be circulated in an uncontrolled manner, or even if there is a risk of uncontrolled disclosure, this level of data capture and retention comes uncomfortably close to sensitive aspects of the erosion of personal privacy. There are related concerns about agencies acting outside of their conventional powers and without due regard for legal process.

The starting position of this metadata metaconversation is that it appears to be a commonly accepted truth that the network, and the network operator, is privy to the complete details of all our communications activities.

Why do we think that? And is it really the case?

It's often useful to compare the Internet to traditional telephony. Not only are folk more confident that they understand the basic concepts of telephony, but also its often the case that when we talk about the Internet, we unconciously borrow terms and concepts from telephony.

So I'll take a quick excursion into the operation of the traditional telephone network. Feel free to move on if you've known all this for the past forty years or so!

> When you make a "call" in a traditional switched telephone network there is an initial exchange of data within what is known as the "control plane". The network takes the dialled number and maps that into a terminating location. A call request control message is sent to this location and the ensuing exchange of messages sets up a virtual circuit in the data plane of the network that links the two telephone endpoints. Termination of the call generates a further exchange of control messages to tear down this virtual circuit.
>
> If we could look at a telephone network's control data relating to call establishment and teardown we would have a form of "metadata" about telephony: a record of who is dialling whom and when, whether the call was answered and, if so, how long the call lasted. But of course this control "metadata" is all about telephone numbers. If we combine this data with current telephone directory information then the combination of the two data sets can be transformed into a "metadata" log of who is talking to whom, when, and for how long. This per call control data does not track what what said, nor can it. It just tracks who is talking to whom. Even in mobile telephony networks the network's ability to track the location of handsets through triangulation of base station data can be coerced to provide the same metadata.
>
> Now telephone networks have been collecting this per-call "metadata" for decades. Most telephone tariffs were based on a rental fee and a usage component, where the usage component was based on who you called, and how long each call lasted. Subscribers could get a copy of this call record data if they had a dispute over their bill, and more recently the telephone companies turned this call log information into a "feature" by generating itemiZed logs on your bill. Not only does it providing you with a comprehensive log of all the telephone calls you made in each billing period, it also illustrates that the telephone company is privy to all of your call behaviour on the telephone network. The public telephone directory maps my name to a telephone number, and the telephone company's call records detail all of the calls I made or received. The two combined form this rich stream of metadata information about who is talking to whom, where and when. Who else is privy to this metadata? The directory is public. The telephone company retains the call information, and no doubt under the terms of an appropriate regulatory instrument certain third parties can also gain access to this information.
>
> Now this is not a new development for telephony. Its been the case for decades. So if the collection and retention of per-call data in the telephone network has been a matter of deep and abiding concern on the part of the users of the telephone network, then they've not exactly been highly vocal on the topic. I suspect that we've all largely grown up with a telephone network that we know generates and retains this data, and we appear to accept that.
>
> Now let's switch context to the Internet.

Given that the traditional telephone network of yesteryear generated a rich stream of metadata about each individual call that is made across the network, why can't we just look for the same information in the Internet. Aren't these both instances of large scale public communications systems with very similar properties? Indeed, many of the old telephone companies now also operate an Internet service, so they can't be that different. My mobile phone is also a mobile Internet platform. It must be the same under the hood. So if we are on the hunt for Internet metadata, why can't we use the same information, the same procedures, the same regulatory framework on the Internet that's we've used on the telephone

network?

Or it this line of thought a case of looking for Internet metadata under a convenient lamppost, when in actual fact this missing metadata is elsewhere?

There are many manifestations of a response to this question, but there is a common architectural theme behind all of these responses. This common theme is that, architecturally, the Internet is not telephony. It's almost precisely the reverse. The Internet inverted the telephone model.

The telephone network is constructed of a "smart" network and dumb devices. Without the telephone network, a traditional telephone handset is a lump of useless plastic and metal. In contrast, the Internet is populated with computers and various incarnations of "smart devices". These connected devices make no critical demands of the network's functionality. The Internet network can be "dumb" and the connected devices will use this dumb network without any problem whatsoever. In the Internet model, these end devices generate datagrams and hand them into the network. The network can deliver these datagrams to their intended destination, or it may drop them. It can re-order them, and in IPv4 at any rate, it can slice and dice them in order to squeeze through narrow network crevices. In this datagram delivery network every packet is an adventure. Normally such a level of unreliability and variability in the network service model would be inadequate to support a useful set of applications and services. But its the responsibility of the network protocol software in the connected end devices to take these arriving datagrams and reassemble the original information stream.

This reversal of roles between network and attached device has a profound consequence. Within the network there is no control data plane that establishes and tears down individual "calls". Within the network there is no "call" at all. When one computer establishes a connection to another computer, then within the framework of the original architecture of the Internet, this connection is a shared state between the two edge computers, and the network not only is not privy to this shared state, it has no need to be privy to this shared state. From this perspective networks do not collect call logs because there is no network-level control plane that causes the network to establish and tear down "calls" in the first place.

Oddly enough, this role removal has not resulted in simpler networks. The result has been more complex networks and even more complex application behaviours. This is now reaching the stage when we can question what exactly is the role of an IP address in today's Internet.

Historically, an IP address was a relatively stable endpoint identifier token: "Everybody who wants to converse with me pushes packets into the network that are addressed to my IP address." These days we are seeing networks whose use of an IP address is limited to an ephemeral conversation endpoint token without any connotations of permanence or even uniqueness, and in response we are seeing applications whose use of an address is even more ephemeral, and IP addresses are redefined as an ephemeral conversational fragment endpoint token, whose lifetime does not even extend across the lifetime of the conversation.

If the hunt for Internet metadata is a hunt for the stable associations of end users with IP addresses, then as we head down this path of redefining what is an IP address, then wherever this metadata might be found, looking inside the network would be a poor place to start.

How did we get to this rather curious situation? Lets look at a few technologies that illustrate this situation.

Carrier Grade NATs

For many years the issue of forthcoming IP address exhaustion was been something that Internet Service Providers (ISPs) pushed over the wall to become a customer problem rather than address this

as a network problem. ISPs assigned their customers a single public IP address, and left it to the customer to incorporate an edge NAT in their modem or similar to permit multiple devices in the home or office to share this single public address. But as the IP address shortfall pressures increased in intensity, there were situations where even one unique IP address per connected customer would require more IP addresses than the ISP actually had. The response has been to deploy NATs in the interior of the network. This allows number of customers (who themselves may have NATs as well) to share a smaller pool of unique public addresses.

Each time an interior device initiates an external connection attempt, the outbound packet is intercepted by the CGN, and an available external port number and IP address pair is pulled from the CGN's managed pool and assigned to this connection. In terms of IP address use, a single public IP address may be used by many users simultaneously. The efficiency of the CGN in terms of maximizing the efficiency of use each public IP address is improved by increasing the pool of interior customers behind each CGN. In other words, there are advantages to the network operator to configure the service network to use a small number of large CGNs, allowing the CGN operator to maximize the efficiency of this form of address sharing. The implication of this form of configuration is that an arbitrarily large number of end users would be using the same IP address within a given time window.

From the outside, users positioned behind a CGN assume the IP address of the CGN itself. And because the same address may be used by multiple users' connections simultaneously, you need to use additional lookup keys to disambiguate one user from the other. For low efficiency CGNs the source port and protocol field is sufficient. But if really high levels of address efficiency are required, CGN binding software may be configurable to operate in a "5-tuple binding" mode, where simultaneous connections to different external services can use identical source-side IP addresses and port numbers. In this situation the only way to disambiguate individual connections in the CGN log is to log both source and destination IP addresses.

What CGNs illustrate is that the Internet, unlike the telephone network, is asymmetric. One class of users have a stable well known IP address. This class of users can initiate connections and receive connection requests. These users are often not human users at all, but are more typically used by servers. Web servers need a stable IP address, as do mail servers, cloud servers, and similar. The other class of users live behind NATs. These users no not have a permanent IP address and cannot receive incoming connection requests. When they initiate connections, the IP address they use to represent their own identity to the network depends on the local configuration of the network. If may be that the IP address is a stable address that is used across all connections from this user. With CGN on the path the picture changes radically, and the IP address that is used to identify the client is in fact just part of a vector that is used to identify the appropriate CGN-binding table entry. In this context the IP address does not relate to the user, and as the user makes further connections there is no assurance that the user will be assigned the same IP address.

Tunnelling

Tunnelling uses a different form of packet transform, where, in its simplest form, a datagram becomes the payload of an enclosing datagram. This wrapping of one IP datagram in another is performed upon tunnel ingress, and the complementary unwrapping is performed at tunnel egress. When outside the tunnel the packet exposes its source and destination address to the network, but within the tunnel the source and destination addresses of the IP packet are the addresses of the tunnel ingress and egress points. The actual IP packet is carried as a payload of the tunnel packet. This hiding of the inner IP packet from the network can be further strengthened by encrypting the tunnel packet payload within the tunnel, using a cryptographic secret that is shared between the tunnel ingress and egress points. None of the activity within the tunnel is visible to the network that carries the tunnel traffic.

Tunnels can assume arbitrary forms. HTTPS proxy tunnels embed IP packets int he payload of a secure transport session that appears to be a secure web transaction. The rationale for this form of

tunnelling is that in certain firewall configurations about the only packets that can be passed through the firewall without being hopelessly mangled are packets that are part of a secure web flow. Because the payload of these packets are encrypted, placing the original IP packet into this stream is a payload is a useful technique. I've heard of IP buried in the DNS, and no doubt its possible to embed IP into many application level protools, but at some point the exercise becomes one of flag planting to prove that such convolutions are possible as distinct from providing a solution to a real world problem.

VPNs

Tunnelling is used by many forms of host based Virtual Private Network (VPN) services, where one end of the tunnel is the user's device, and all the device's traffic is passed into the tunnel within the device. The device's interaction with the local network is limited to passing encrypted traffic to and from the tunnel egress point. The visible local IP address of the end device doesn't communicate with anyone other than the nominated tunnel egress. At the other end of a connection, the remote service receives a packet whose source address is associated with the tunnel egress point. In this form of tunnel configuration there is no point where the local IP address of the device and the remote address of the server are exposed together in the same packet header.

VPN + NATs

The VPN and NAT functionality can be combined by using a NAT at the common tunnel egress. The VPN client is provided with a private address by the VPN provider, and a secured tunnel is set up between the client device and the VPN provider. When an outbound packet arrives at the tunnel egress point, a local NAT is used on the inner IP packet to transform the packet's source address to the local IP address of the NAT. Inbound packets arrive at the NAT, and a comparable transforms applied to the destination address fields, and the packet is then passed into the tunnel for transmission to the VPN client. Here again there is no point where there is an IP packet whose packet header contains the address of the VPN client and the address of the remote service.

TOR

Into this combination of IP-in-IP tunnelling, payload encryption, VPNs and NATs we can add relaying, and the result is the TOR network. TOR is a form of relayed tunnelling where a packet is passed from relay to relay through tunnels, and at no point is the a cleartext IP packet that contains a packet header with the actual IP addresses of the two parties who are communicating via TOR. TOR is an asymmetric protocol, in that both sides of a TOR conversation do not need to be TOR-aware, allowing a TOR client to connect to any conventional IP service.The service receives an IP packet whose source address is that of the TOR exit router, and its response is passed to this TOR point, which then applies a TOR tunnel wrapper and sends it back along a tunnel relay path to the original TOR client. One can go further with TOR and wrap it in WebSocket format and bounce the traffic though short-lived javaScript proxies on browsers to further disguise the TOR traffic pattern so as to emulate conventional secure web transactions, for example, so that even traffic profiling scanners would be unable to distinguish a TOR tunnel from other unrelated traffic.

V6 Transition

Interestingly, the IPv6 transition has also become enmeshed into this story of increasing network complexity in their treatment of IP addresses in packet headers. As originally envisaged, network operators would progressively deploy IPv6 across their infrastructure in addition to IPv4, in a so-called dual stack configuration. End users would need to wait for their service provider before they could connect using IPv6. To some extent impatience took over, and it was not long before we saw various tunnel approaches used to connect "islands" of IPv6 across the oceans of an IPv4 substrate. Some of

these used IP-in-IP (such as 6to4) while others used IP-in-UDP-in-IP (such as Teredo) in order to traverse NATs. These approaches have been taken up by network service providers who are attempting to address the twin issues of IPv4 address depletion and IPv6 deployment simultaneously.

Some approaches use a single protocol IPv4 network, and tunnel IPv6 over this infrastructure, such as 6RD, while other approaches use an IPv6 common substrate and tunnel IPv4 across it. The tunneling can take the form of one of the various permutations of IP-in-IP, or if the substrate is IPv6, the approach can use protocol translation and embed the original IPv4 addresses into the interface identifier part of the IPv6 packet headers. This translation can be mapped, so the transform is stateless, or it can be performed using a NAT-like function, using a stateful translation. There are now many approaches to IPv6 transition, and ISPs appear to be customizing their choices from this selection rather than adopting a uniform approach.

So in a network that performs one of these transition mechanisms, what does an IP address signify? Is it an endpoint identity or an address of a translating unit? Is the address a synthesised compound of one or more IPv4 addresses embedded into the IPv6 address (such as Teredo or 6to4 and their form of embedding V4 addresses into a V6 address) or are the IPv4 addresses contained in the payload of the outer packet?

In this hybrid environment there are no clear and consistent outcomes. IP addresses can take on many forms and the address itself typically provides no clue as to its role. It may be a stable endpoint identity, or it may be an address which has significance only within a particular scope or context. It may not be a stable address, and may only have significance in conjunction with other fields in the IP packet header. It may be an ephemeral conversation token, or it may be just a one-off translation table index. Its only when you understand the context of the address, and understand the form and function of the units that have to manipulate the address can you place any meaningful interpretation on what an IP address signifies.

V6

There is a point of view that many of these transitional complexities are due to the combination of the difficulties in this transition to IPv6 and the exhaustion of IPv4 addresses. The claim is that if we were operating an all-ipv6 network it would all be far simpler, we could dispose of these cumbersome and complex NATs, remove all this transitional complexity and IP addresses would once more become stable endpoint identifiers.

But that's not going to happen.

IPv6 now has taken on so-called "privacy addresses" with a passion. Devices that use this function periodically generate a new 64-bit interface identifier based on some form of random number identification, and generate a new local synthetic address combining a common 64 bits of the network identifier part with a self-selected random lower 64 bits. In this world of privacy addressing IPv6 addresses have a dual personality. The upper 64 bits are a stable endpoint identifier that identifies a local network. But the lower 64 bits are again ephemeral, and have no permanent existence. IPv6 hosts can use these privacy addresses for outbound connections, but cannot use them as stable connection points for inbound connection attempts.

In IPv4 the concept of private addresses and NATs has been around for a very long time. IPv6 has never been clear about its position on these concepts. Initially, IPv6 had no such concept. Local addresses were simply global addresses that were exclusively used in some local scope and were unreachable from elsewhere, akin to the original concept in IPv4. However, we subsequently saw the reservation of a very large block of IPv6 addresses to be used exclusively in local addressing scoped contexts, where the remainder of the network part of the address was locally defined. These Unique Local Address prefixes (ULAs) are now being used in the same manner as private addresses in the IPv4

realm. NATs also exist in IPv6, and they appear to come in a couple of flavours: NAT66 performs a NAT translation on the entire 128 bit address field, in a manner entirely analogous to a NAT in IPv4. NPTv6 is a variant of this function that performs an address transform only on a the address prefix, leaving the remainder of the IPv6 address field unaltered.

Other forms of address transforms are encompassed by the SHIM6 work. This leverages the observation that in IPv6 its possible for a device to have a number of IP addresses simultaneously. There are numerous motivations for this, including explicit address scoping as part of a site security framework, but the explicit area SHIM6 was addressing was the practice of edge networks "multi-homing" with multiple upstream transit service providers for greater resiliency. In IPv4 this is undertaken by using so-called provider independent space and asserting a unique entry in the global routing table. While this is possible in IPv6, there was the desire to be a bit kinder to the routing table, and see if there was a solution which used provider-based addressing from each of the upstream providers and allowed individual TCP sessions (indeed allowed any IP end-to-end context to be address agile without disturbing the upper level end-to-end transport sessions. One way to look at SHIM6 is that it embeds a NPTv6 function right into the host's IPv6 protocol stack, and via a dialogue with its counterpart at the other end of the connection, allows the local host to switch provider prefixes at will, and thereby move the transport session from one provider to another without disturbing the transport session. At a network level this would produce some peculiar outcomes. TCP sessions could simply appear within a network without any opening context of a SYN packet exchange, and equally they could disappear without any visible shutdown. Pretty clearly, a combination of SHIM6 and CGNs in IPv6 would be mutually incompatible. In the case of SHIM6, the IP address visible to the network is not necessarily useable without the previous establishment of a SHIM6 context.

There are other aspect of IPv6 address management with a similar flavour. What they all illustrate is that in IPv6 an IP address is not necessarily a stable end point identifier either. Whatever simplicity is being regained in an all-IPv6 network, a consistent and simple semantic interpretation of all IPv6 addresses is not part of the package being offered.

MultiPath TCP

As a final illustration of the level of complexity that we see in the Internet today I'd like to highlight MultiPath TCP. In devices with two or more interfaces the protocol in the local device uses a local rule to determine which interface to use to send an outbound packet, and this choice is "sticky" at least for the level of granularity of a TCP connection. TCP cannot move from one connection to another, or even use multiple connections in parallel. Until MultiPath came along. With MultiPath TCP the local protocol stack is able to open up multiple TCP connections to the remote endpoint, and then distribute the application payload data stream across these multiple connections.

All this sounds a bit esoteric until you look closely at the device in your pocket. Often it has a WiFi interface and a cellular data interface. Normally there is a local configuration rule that says "when the WiFi interface is useable stop using cellular data". When cellular data was a highly priced limited service and WiFi was an abundant service with no marginal use tariff, this kind of local preference rule made a lot of sense. But increasingly we see "unlimited" market offering for cellular data, and the speeds of cellular data are rising to a level that is comparable to many WiFI services. If we had MultiPath TCP we could open up a connection on each interface and use both at once. The faster of the two would deliver more data, and we would optimize the speed of the overall transaction as a result. If the device in your pocket is an Apple device with a recent version of iOS, then chances are that when you use the Siri application the application will attempt to use MultiPath TCP if it can. All this may sound esoteric, but there are hundreds of millions of these particular devices out there, and as a result MultiPath TCP is an esoteric technology with a user base that numbers in the hundreds of millions! Any other technology would call itself "mainstream" based on those deployment metrics!

What does MultiPath TCP mean for the network? On any individual network, the network sees only

fragments of the full data exchange. The complete content flow is being passed across multiple networks at once, using multiple IP address pairs. Its possible that no single network carries the complete conversation flow, and even if you could look at the packet flows within each of these networks, the binding glue that identifies each sub-stream as part of a common TCP MultiPath stream is not necessarily visible to the network: that essential context information is held as state on the two end devices.

Where's the Metadata?

There is no doubt that Internet service networks generate large volumes of data about the network, or "metadata". This data is used to operate the network, and used for network planning. It often drives the network's security subsystems. It's used for authentication and accounting of the network's customers. It's used to enforce various network policies and operational practices.

But does this torrent of metadata about how the network operates provide an insightful window on the identity and actions of its end customers? Do IP networks have the ability to generate the equivalent of the telephone call log?

Increasingly, the answer is "no", and the more we are provided with information on various surveillance efforts to use the network to extract such information, the response from end user systems and applications is even more in the direction where the users' actions are kept hidden from the network.

If you really want to understand what is happening in an end-to-end network, the best place to do so it at either end, and the best way to do so is within the application. But we have a rich regulatory history, gathered in the postal, telegraph and telephone industries of using the network as the point of regulatory control. We have various legislative frameworks that govern the way in which public telecommunications services can operate. We recycled this entire legacy framework when the Internet came along, and these days the locus of attention in terms of regulation remains the network. But the Internet does not use a network-centric architecture. As we've explored here there are many ways in which users, and the applications that users run, can still function perfectly normally, yet still hide their essential aspects of their communications from the network itself. But applications and servers are not conventionally subject to public regulation. What happens on a web site is the business of the web site's owner and operator and there are few, if any, regulatory constraints in the way in which the web site operates. Our response has been through generic measures in terms of consumer protection and privacy protective measures, but these measures are generic and often poorly applied in this context.

Looking within the network to try and piece together exactly what is happening end-to-end is a guessing game that is increasingly becoming a rather expensive and futile exercise. And forcing network operators to collect and retain their data for arbitrarily long period of time does not restore any form of rationality to the exercise. In the end all that is being collected here in the form of network metadata is just a truly massive pile of useless bits.

So when visible action is called for, and various agencies whose charter includes aspects of national security are called to stand up and be seen to be engaged with the national security agenda, then it is perhaps no surprise that the network is the focus of their attention. Not because there is a rich vein of readily accessible information lurking there. Far from it. It's just that the network is a convenient, well lit and well established lamppost. But the keys truly are elsewhere.

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.