NIPAA-21 - 2nd Workshop on New Internetworking Protocols, Architecture and Algorithms,
ICNP-2021 - 29th IEEE International Conference on Network Protocols.
Virtual Event, 1-5 Nov 2021

End-to-End Privacy for Identity & Location with IP

Saleem N. Bhatti, Gregor Haywood, Ryo Yanagida
School of Computer Science, University of St Andrews, UK
{saleem,gh66,ry6}@st-andrews.ac.uk

***Abstract*—We describe protocol features to provide both Identity Privacy and Location Privacy at the network layer that are truly end-to-end, strengthening the trust model by constraining the boundary of trust to only the communicating parties. We show that Identity Privacy and Location Privacy can be provided by changing only the addressing model, whilst still remaining compatible with IPv6. Using the Identifier-Locator Network Protocol (ILNP), it is possible to use ephemeral end-system ILNP Node Identity (NID) values to improve identity privacy. Using the ILNP Locator values with dynamic bindings, it is possible to use multiple IPv6 routing prefixes as network Locator (L64) values to provide (topological) location privacy. This is achieved: (a) whilst maintaining end-to-end state for transport protocols, without proxies, tunnels, or gateways at the transport layer or application layer; and (b) without the use of cryptographic techniques, so performance is not impacted.**

I. Introduction

The Internet community has long had best practice for considering security issues in protocols [1], but it was not until the Snowden leaks, a decade later, that recommendations for privacy issues were documented [2]. However, privacy issues have been part of protocol design for some time before this. Today, privacy issues are considered at design time for new protocols, while work continues for improving privacy for existing protocols.

For application layer protocols, privacy concerns arise when sensitive and non-sensitive personally identifiable information (PII) is visible in communication exchanges, or via the user interface. For lower-layer protocols, there are different *name* types: a *name*, in the most general sense, being a set of bits with specific semantics and bindings to objects at a specific protocol layer. Such a lower-layer name might not provide a direct link to PII, but, with additional information, could allow an attacker to implement privacy-invasive analytics. For example, an IP address for an end-system could be used to detect communication flows, and help to identify users, even if the IP payload is protected from inspection.

A. Contribution and paper structure

In this paper, we show how the Identifier-Locator Network Protocol (ILNP) increases the effort required for an attacker to perform privacy-invasive analytics by inspection of IP address values in packet headers. We demonstrate that our approach:

- Does not need to use proxies or tunnels, so does not require sharing of privacy-sensitive addressing information with a third party.
- Does not require the use of cryptographic techniques, so will have negligible impact on performance.
- Can be used for *any* transport layer protocols, including existing TCP and UDP implementations.
- Can be used with new and existing IPv6 applications without those applications needing to be modified.

We make critical analysis of the ILNP architecture in comparison to other systems currently in use, and show that ILNP is the only approach to offer such capability for IP.

In Section II we provide the problem statement for our work, with reviews of current popular mechanisms in Section III. We show the efficacy of our approach in Section IV, with analysis in Section V, finishing with a summary in Section VI.

978-1-6654-4131-5/21/$31.00 ©2021 IEEE

II. Use and visibility of addressing information

Addressing information exists in end-system protocol state, and is visible in the wire image of a protocol packet [3]. Addresses need to be visible in packet headers for correct forwarding of those packets. However, an IP address contains, implicitly, both identity and (topological) location —- a long-standing problem, recognised in the Internet community [4], and impacting network operation beyond the privacy issues that are the focus of this paper.

A. Identity Privacy

The default mechanism for generating a 64-bit Interface ID (IID) for an IPv6 address was to use directly the 48-bit Extended Unique Identifier (EUI-48) value assigned to a communication interface on a device. The EUI-48 value was designed to be globally unique, is typically encoded into hardware devices, and is converted algorithmically to a 64-bit EUI (EUI-64) value for the IPv6 IID. This allows a device to be tracked for the *lifetime* of that interface, *globally*.

This problem was recognised early for IPv6, so modern implementations typically include one or more of several schemes that allow local generation of EUI-64/IID values [5], e.g. opaque stateless address auto-configuration (SLAAC). However, implementation and deployment of these mechanisms is variable across operating systems and devices. More significantly, even with these mechanisms employed, multiple flows could still be correlated, as the same IID is used across all the flows using the same IPv6 address.

B. Location Privacy

Network prefixes are used for routing – they have topological significance. Many, widely-accessible geolocation look-up services allow some level of geographic resolution based on IP address prefixes, at least identifying the ISP to which a prefix is routed to. However, even relatively coarse identification of topological or geographical location may lead to privacy issues, e.g. if communication can be tracked to regional areas, or a specific organisation, this could be coupled with additional information to localise tracking of an individual device or user.

Overall, the current model for Internet routing and forwarding requires direct visibility of routing prefixes in packets. Unless this model changes, then prefix values need to remain visible. So, potentially, a packet always has embedded within it information about its (topological) location in terms of destination and source networks, which is directly visible.

C. Default mechanisms in ILNP

The Identifier-Locator Network Protocol (ILNP) is currently being implemented as a superset of IPv6 [6]–[9]. ILNP replaces the use of the IP addresses with *Node Identifier (NID)* values and network *Locator (L64)* values, as two distinct name types in the communication stack.

ILNP packets carry an Identifier-Locator Vector (IL-V) in place of an IPv6 address, as shown in Figure 1. The L64 value occupies the same bits as the IPv6 unicast routing prefix, and the NID takes the place of the IPv6 IID bits.

```
IPv6 (RFC8200(S)) - general IPv6 global address format:

 | 3 |     45 bits         |  16 bits  |              64 bits               |
 +---+---------------------+-----------+------------------------------------+
 |001|global routing prefix| subnet ID |    Interface Identifier (IID)      |
 +---+---------------------+-----------+------------------------------------+


ILNP (RFC6741(E)) - Identifier Locator Vector (IL-V):

 |                64 bits                |               64 bits              |
 +---+---------------------+-----------+------------------------------------+
 |                Locator                |       Node Identifier (NID)        |
 +---+---------------------+-----------+------------------------------------+
```

Figure 1. An ILNP Identifier-Locator Vector (I-LV) value has the same structure as an IPv6 address. An IPv6 routing prefix is used as a L64 value, with the same syntax and semantics. The NID has the same *syntax* as an IPv6 Interface Identifier (IID), but has different semantics: the NID is bound to a *node*, not an *interface*.

ILNP packets carry an IPv6 Nonce Destination Option (‘nonce header’) [9]. This holds an ephemeral value of 4 bytes (default) or 12 bytes, which provides lightweight protection against off-path packet spoofing.

In its default configuration, ILNP has no better (and no worse) privacy for identity and location compared to IPv6.

D. Visibility of end-system state

The end-system transport protocol state needs to remain invariant for the duration of a transport protocol flow. However, visibility of those invariants, especially across multiple flows and across different timescales, allow privacy-invasive analysis to become more viable and effective. For example, in tuple expressions (1) and (2) is shown the TCP and UDP protocol end-system state, respectively, for two nodes, X and Y, communicating with addresses, A, and port numbers P. Whilst port numbers have purely local significance (though might still indicate application usage), the addresses have global significance, even if Network Address Translation (NAT) is used for the site.

$$\langle tcp : P_X, A_X, P_Y, A_Y \rangle \langle ip : A_X, A_Y \rangle \tag{1}$$

$$\langle udp : P_X, A_X, P_Y, A_Y \rangle \langle ip : A_X, A_Y \rangle \tag{2}$$

The visibility of IP addresses in an IP packet header allow the progress of an individual flow, and multiple flows, to be identified and correlated. So, the two aims of our approach are for a packet’s wire image to:

A1 change the *exposure* of state invariants; and
A2 change the *correlation potential* of state invariants.

III. Current popular approaches to privacy

To achieve A1 and A2 today, solutions are based around obfuscating or hiding the value of the network addresses in the packets headers, for example, by changing those values completely at one or more hops along the network path. An overlay, proxy, address translation, or gateway service, might be used, which holds state to preserve a mapping between the ‘real IP address(es)’ used in packets to/from an end-system and the ones that are visible on the Internet.

A. Virtual Private Network (VPN) services

Virtual Private Network (VPN) services are popular for this kind of capability. They successfully hide source addresses, and so provide utility for users with client systems trying to access services and maintain privacy. At least some (most, perhaps) of the popular use of VPN services by consumers is to circumvent geolocation-based service access rather than for privacy, e.g. for access to geographically-constrained video-on-demand (VoD) services.

However, VPN services use secure tunnels from a user to the VPN service provider, increasing computational cost, and latency for communication. Also, the VPN service provider must be *completely trusted by the user*, as that provider sees all the original user traffic, and would be able to infer all the privacy information which the user is trying to hide by employing that VPN service in the first place.

B. Tor

The *Tor* network provides a popular approach, and does not have the same, single trusted-party approach required for VPNs. It is also a free service, relying on a community of users to run the Tor relays and provide a global service. Especially for web-based services, this is a convenient and easily-accessible mechanism.

However, the key disadvantage with Tor is, again, the additional processing overhead and latency, as tunnelling and multiple relays are used. Also, applications typically need to be modified to operate with Tor, and not all existing applications work correctly when Tor is in use. The main use of Tor is via the Tor browser (https://www.torproject.org) for privacy in accessing WWW-based services.

C. *The trust domain boundary*

In the two popular approaches described above, as well as performance issues, the following issues exist:

- A NAT-like function might be necessary, which might not work for all applications.
- An application-level gateway function might be required, e.g. Tor has a specific gateway function for HTTP.
- The availability of the systems(s) will vary, depending on local service deployment, and permissible access to the service(s), e.g. site, local, regional, or national policy might restrict deployment and access.
- *The boundary of the trust domain is not confined to the communicating parties*: one or more third-party service providers needs to be trusted implicitly, especially if they provide cryptographic protection of packets.

So, in the cases of both VPNs and Tor, a completely general solution is not offered, though each solution has been shown to have utility for popular applications, including access to WWW-based services and VoD, for example. However, the last item listed above is a key point: is it possible to constrain the trust domain to the communicating parties, in manner which is more generally applicable, whilst still obtaining some level of identity privacy and location privacy?

IV. End-to-end privacy in addressing with ILNP

As ILNP can be seen as a super-set of IPv6, our description focuses on how ILNP packet handling and state information *differs from IPv6 in the end-systems*, but remains *backwards compatible with IPv6 packet handling in core network devices*, to fulfil aims A1 and A2 from Section II-D.

We are concerned with the identity privacy and location privacy of *individual* users or devices. So, our context is addressing for the individual user, who is usually involved in an interaction with a service or service provider.

A. *Basic mechanism*

The tuple expressions (3) and (4) give transport flow-state for IP and ILNP, respectively. The suffixes X and Y are, respectively, for two communicating systems engaged in the communication session. A is an IP address, P is a port number, N is a NID value, and L is a L64 value. At the interface (if), an IP address is bound semi-permanently to the interface. In ILNP, there is only a dynamic binding to a Locator (L64) value, the latter itself having a dynamic binding to an interface. We extend this basic approach, exploiting the dynamic bindings for identity privacy and location privacy.

$$\langle tcp : P_X, A_X, P_Y, A_Y\rangle\langle ip : A_X, A_Y\rangle\langle if : A_X\rangle \tag{3}$$

$$\langle tcp : P_X, N_X, P_Y, N_Y\rangle\langle ilnp : (L_X)\rangle\langle if : (L_X)\rangle \tag{4}$$

B. *Identity Privacy with ILNP*

NID values for ILNP I-LVs may be generated using any method that is used to generate IID values for IPv6 addresses [5], e.g. pseudo-random [10], [11], or algorithmically generated as an opaque, stable value [12].

However, ILNP allows *multiple* NID values to be used simultaneously: each transport layer flow can have a *different* NID value: a NID value can be *generated dynamically* as required for each *individual* transport layer flow. For IPv6, the *same* IID value would be used for *all* flows. For example, the tuple expressions (5) and (6) show the state of two separate TCP flows from the *same* node, one with NID value $N1_X$ and one with NID value $N2_X$.

$$\langle tcp1 : P_X, N1_X, P_Y, N_Y\rangle\langle ilnp : (L_X)\rangle\langle if : (L_X)\rangle \tag{5}$$

$$\langle tcp : P_X, N2_X, P_Y, N1_Y\rangle\langle ilnp : (L_X)\rangle\langle if : (L_X)\rangle \tag{6}$$

So, for ILNP, packets across multiple flows cannot be easily correlated by a remote observer or man-in-the-middle (MITM), and multiple flows cannot be identified as originating from the same user or device.

This approach changes the exposure of end-to-end state invariants, as it constrains the transport state invariant value to packets within an individual flow (aim A1), and makes it much harder to correlate different transport flows from the same device or user, especially over time (aim A2). As the NID value is part of the end-to-end state for the transport protocol, overall, this improves identity privacy compared to IPv6. *Ephemeral NID values in ILNP make it more difficult for an attacker to track flows for individual users based on the identity presented by the IPv6 address field.*

However, even with different NID values for different flows, the same L64 (IPv6 routing prefix) might be visible to a MITM observer who is 'on-path', and we deal with this next with our approach to Location privacy.

C. *Location Privacy with ILNP*

L64 values for ILNP I-LVs are IPv6 prefixes, and so may be discovered and configured automatically in the same way as for IPv6, e.g. via IPv6 Router Advertisement (RA) messages, or via DHCPv6 prefix delegation. So, again, potentially, ILNP has no better (and no worse) location privacy properties in addressing compared to IPv6.

However, ILNP allows *multiple* L64 values to be used *simultaneously*, and a single NID can be bound simultaneously to more than one L64. Recall that the 'single-homed' transport state of IP (expressions (1) and (2)) is bound to a single IP address, and to a single physical interface. However, ILNP end-system state is *not* tied to a physical interface, and the transport state can be 'dynamically multihomed'.

The L64 values and their respective dynamic bindings to NID values and to interfaces can be changed as connectivity changes occur. When multiple L64 values are available to an end-system, a single NID can be bound simultaneously to one or more of those L64 values, and so the end-system can transmit and receive a single flow over multiple IP networks, exploiting multipath transmission. For example, at an end-system using locator L_X, if another locator, L_A becomes available, from expression (4), we have expression (7):

$$\langle tcp : P_X, N_X, P_Y, N_Y\rangle\langle ilnp : (L_X|L_A), (L_Y)\rangle\langle if : (L_X|L_A)\rangle \quad (7)$$

In expression (7), the end-system is now using locator L_X and locator L_A simultaneously, so packets being sent to I-LV $\langle N_Y, L_Y\rangle$ could be sent from either I-LV $\langle N_X, L_A\rangle$, or I-LV $\langle N_X, L_X\rangle$. This *end-system multihoming* at the network layer offers multipath connectivity to *all* transport layer protocols.

Also, as the ILNP L64 value is *not* part of the transport state, unlike the IPv6 prefix, it means that *the L64 value in an ILNP packet can be changed during the lifetime of the ILNP packet, without impacting end-to-end state*. If the end-system is on a multihomed IPv6 network, ILNP allows multiple IPv6 prefixes, e.g. *provider-allocated (PA)* prefixes, to be used as L64 values for a single transport layer flow. As this is done at the network layer, *any* transport protocol can use multiple locators, use multipath capability, and have L64 values re-written along the end-to-end path without impacting end-to-end state. *So, ILNP makes it much harder for a remote observer to mount a MITM attack, even for passive inspection, as the flow can traverse multiple network paths, which can be changed dynamically.*

When multihoming is in use, *provider-independent* (PI) prefixes are not required: indeed, using PA prefixes can improve the location privacy of flows.

So, ILNP changes the exposure of end-to-end state invariants, as topological location in packet headers is no longer linked with the end-system state, and the location information can be changed even for packets in the same flow (aim A1). Identifying topological location for a flow becomes more challenging for an attacker, both for correlating packets within the same flow or across multiple flows (aim A2). *With ILNP, an attacker needs to intercept packets at ingress/egress points to see all packets for a flow, as packets might take different paths across the Internet, even if they have the same NID.*

D. Implementation

For location agility via dynamic multihoming, work on ILNP has demonstrated the use of multiple L64 values for IP-layer mobility with network-layer soft-handover [13], [14]. This shows that multiple L64 values can be used and changed dynamically with end-to-end signalling directly between end-systems, with near-zero gratuitous loss for packet flows, through modifications to end-system state management for the default, in-kernel TCP and UDP implementations.

Combining both the Locator agility, and the use of ephemeral Identifier values, consider the examples of tuple expressions (8) and (9) for two *separate* flows from the *same* node. Each flow sends packets over two separate Locators (two separate interfaces), and each flow has a unique Identity.

$$\langle tcp1 : P_X, N1_X, P_Y, N_Y\rangle\langle ilnp : (L_X|L_A), (L_Y)\rangle\langle if : (L_X|L_A)\rangle \quad (8)$$

$$\langle tcp2 : P_X, N2_X, P_Y, N_Y\rangle\langle ilnp : (L_X|L_A), (L_Y)\rangle\langle if : (L_X|L_A)\rangle \quad (9)$$

This approach does not require the cooperation of a network service provider: only basic, unicast IPv6 connectivity is required, even when multihoming is used. *So, the trust domain boundary can be constrained to the communicating end-systems, with no implicit trust needed for any service providers.* For data privacy of the content/payload of a packet, existing mechanisms, e.g. Transport Layer Security (TLS), can be used as today, and is outside the scope of this work.

In parallel with an on-going, in-kernel implementation in FreeBSD, we created an in-lab emulation, which ran as an overlay on IPv6. This involved the creation of multiple virtual (emulated) networks over a single LAN, by using separate IPv6 multicast addresses as separate 'ILNP networks', each with a different L64. The overlay protocol stack that was created was: ILNP (emulated) / UDP (multicast) / IPv6. This allowed us to have a 'single-lab' test-bed, to assess the efficacy of our approach from the wire-images that are visible on the network, in an easily-configurable manner.

E. Results

A summary of our current emulation results on our in-lab testbed, with multiple NIDs and L64s for a *single* source node (e.g. a single user device), is shown in Figure 2.

Case A. IPv6, with a single IP address, or ILNP with single, fixed I-LV. All flows can be seen and correlated to a single device or user.

Case B. ILNP with multiple NIDs. 3 flows are visible. Each flow uses a unique NID, so an observer cannot easily correlate across flows or link them to a single device or user.

Case C. ILNP with multiple NIDs and multiple L64s. The multiple L64s can be used simultaneously with multiple NIDs, and the multiple L64s could result in multipath flows. This makes 3 flows look like 9 flows. Coupled with the multipath transmission, this makes the cost to an observer or MITM much greater: the observer would need multiple observation points, and would need to coordinate these points to capture even the packets for a single flow.

The scenarios of *Case* B and *Case C* cannot be provided with IPv6, VPNs, or Tor. No cryptographic techniques are required for our approach with ILNP, and IPv6 mechanisms are re-used wherever possible for backwrads compatilibility and ease of deployment.

V. Analysis

Here, briefly, we critique our approach, compare with existing approaches, and discuss the challenges that still exist.

A. Comparing with the use of VPNs and Tor

The use of VPNs and Tor still has one advantage so far: VPNs and Tor change the ingress/egress point of packet flows such that geolocation mechanisms using IP address prefixes cannot be used effectively.

However, for ILNP, a sequence of Locator Re-writing Relay (LRR) functions could provide similar functionality [15], [16]. A 'chain' of ILNP-aware forwarding functions would re-write L64 value(s) in a packet to alter the forwarding path of a

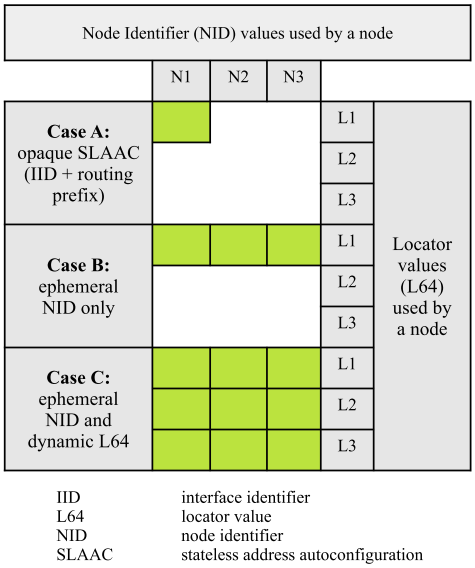
Node Identifier (NID) values used by a node
N1
N2
N3
Case A: opaque SLAAC (IID + routing prefix)
Case B: ephemeral NID only
Case C: ephemeral NID and dynamic L64
L1
L2
L3
Locator values (L64) used by a node
IID interface identifier
L64 locator value
NID node identifier
SLAAC stateless address autoconfiguration

Figure 2. An observed naming matrix for a *single* node with 3 ephemeral NIDs (Np) and 3 dynamic L64s (Lq) for source addressing. The green boxes show how many "users" would be "detected" by an observer or MITM using the source address. *Case A:* A single NID-L64 or a single IPv6 address would link all flows based on the visible source address field in the IPv6 header. *Case B:* 3 NIDs used for 3 different flows with the same L64 could be seen as 3 different users from the same site. *Case C:* 3 different NIDs and 3 different L64s would be seen as 9 different users from 3 different sites, and require multiple observation points to capture packets for any single flow.

packet, so changing its topological ingress/egress point for an observer. A simple, single-hop chain is shown in Figure 3 (based on [16]) as an example. Re-writing the L64 in a packet has no impact on end-to-end state, requires no cryptographic manipulation, and no other changes to the packet, making this an efficient, low-cost function. Between LRRs, normal IP forwarding is used, based on the L64 values in packets, which are normal IPv6 routing prefixes.

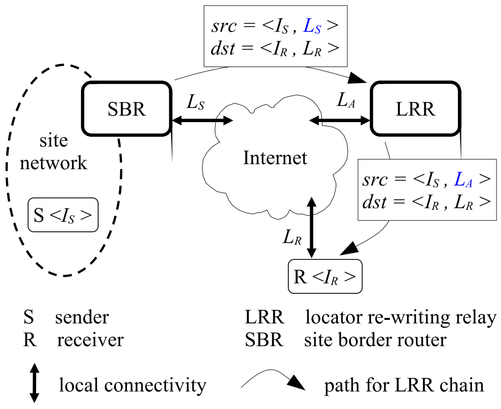

Figure 3. A Locator Re-writing Relay (LRR) chain can provide a degree of location privacy to perturb geolocation services. The packet from S, with NID I_S, is transmitted by the Site Border Router (SBR) from its current location with L64 value L_S to the LRR. The LRR rewrites the value L_S to L_A before forwarding the packet on to the receiver, R, which has NID I_R. R always sees only L_A as the L64 value for S.

The operation and configuration of a LRR chain can be compared to a virtual circuit. A LRR chain does not change the trust model compared to IP forwarding. The level of trust required for LRRs would be almost identical to that required for a service provider that implements correct IP packet forwarding: the LRRs do not have any responsibility for cryptographic protection, unlike in a VPN, or in Tor.

A single LRR, as in Figure 3, does now become a point of interest for an attacker, as it contains the mapping between L_S and L_A. The SBR could also include an LRR function, and the longer the LRR chain becomes, the harder becomes the attacker's task to determine the path taken by a packet. Indeed, it is possible to imagine a community-based LRR chain service, much like the community-based Tor service, implemented using an additional suitable control plane and management plane protocol. (An in-kernel LRR implementation for FreeBSD is currently in progress.)

Also, Tor requires modifications to applications for correct operation, but ILNP requires no special implementation considerations for applications.

B. *Other Identifier/Locator approaches*

There are a number of other Identifier/Locator protocol architectures. A summary of many of the approaches, including ILNP, did not explicitly consider identity privacy and location privacy issues, but did consider some security issues [17]. It is not possible here to review every other Identifier/Locator proposal, but the Locator-Identifier Separation Protocol (LISP) [18] is discussed, as it takes a different approach to ILNP in separating Identity and Location in addressing.

LISP reuses IP addresses with two different semantics: some IP addresses can be End-system Identifier (EID) values, and some IP addresses can be Routing Locator (RLOC) values. LISP uses a 'map-encap' architecture: packets from a source are sent to an entry point in the EID-to-RLOC infrastructure where a RLOC value corresponding to the destination EID is found; the packet is then forwarded, in a tunnel, based on the RLOC value, to an egress point in the EID-to-RLOC infrastructure; and then the packet is forwarded to the destination EID. EID-to-RLOC mappings need to be updated and maintained within a distributed mapping system (consisting of Mapping Servers), via additional control plane protocols.

Compared to ILNP, LISP does not require updates to the end-system OS stacks, but does require the EID-to-RLOC infrastructure and mapping system to be deployed, configured, and maintained within the network. There exists a description of how location privacy and identity privacy could be added to LISP Mobile Node (LISP-MN), an adaptation of LISP to support mobility, via a proxy service and NAT-like functions [19]. It is feasible that this could be applied to LISP in general, and not just in the mobile node situation. However, overall, this would require the trust domain boundary to be extended to the EID-to-RLOC mapping system, and to the proxy service.

Overall, LISP requires the use of tunnels, a distributed mapping function, and additional infrastructure, as it is an overlay service on top of IP. This has, in our view, several disadvantages compared to ILNP:

- LISP has to be deployed in the network, whereas ILNP only needs basic IPv6 forwarding.
- LISP nodes will need to hold EID-RLOC mapping state, and so end users need to trust the provider beyond the ability to simply forward IP packets correctly.
- LISP adds complexity to the networking landscape for deployment and operation, so has an increased cost for a provider in operation compared to ILNP.
- Overall, the distribution of location and identity information across LISP infrastructure increases the attack surface for the end-to-end communication.

C. Applications and API support

The ILNP implementations can support multiple NID values for a node at the protocol level, as defined in ILNP. However, the challenge with using multiple NID values is that the current APIs do not support dynamic NID generation, or NID selection from a set of NID values. The current implementations [13], [14] describe how support for multiple L64 values is enabled without changes to the C sockets API, and there is ongoing work to allow generation of NID values for each new socket that is created.

Such API challenges are also faced by other new transport protocols, such as Multipath TCP (MP-TCP) [20]. MP-TCP performs multihoming at the transport layer, and integrates congestion control that is multipath-aware. Compared to the current ILNP implementation using non-multipath aware TCP (e.g. TCP CUBIC in Linux), MP-TCP has an advantage for congestion control. However, as the L64 values are visible to TCP state management code, a future MP-TCP-like realisation on top of ILNP could focus on the congestion control aspect, using the native addressing mechanism of ILNP.

Similarly, QUIC [21] could work over ILNP, gaining the transport layer benefits of content protection that has been defined for QUIC, with the network-level protection from ILNP that has been described here. (There is ongoing work to explore the operation of QUIC over ILNP.)

VI. Summary

The Identifier Locator Network Protocol (ILNP) changes the addressing model for IP, so that identity and (topological) location are distinct name types within the communication stack. The crisp semantics and use of node identity (NID) and locator (L64) values allows: (a) transport protocol state to be location-independent; and (b) for one-to-many, dynamic bindings between NID values and L64 valued.

Using ephemeral NID values, generated with IPv6 algorithms, and L64 values that are IPv6 routing prefixes, allows ILNP identity privacy and location privacy to be realised in a way that is compatible with IPv6 addressing, routing, and forwarding behaviour implemented in current networks.

With ILNP, no additional cryptographic mechanisms, tunnels, proxies, or mapping functions are required, and so the trust domain boundary for two communicating parties is confined to the those two parties only. The trust model for communication in ILNP follows an end-to-end relationship in providing identity privacy and location privacy.

Current applications can use ILNP with the C sockets API, with dynamic use of L64 values. Work is in progress to allow use of multiple NID/L64 values simultaneously, as well as to implement LRRs. Our emulation results here indicate that the use of the ephemeral NID values for flows, and dynamic L64 values for a node would have a positive impact on protecting identity privacy and location privacy for end users.

Acknowledgement

The authors are grateful to Shen Yan for guidance and feedback that helped to improve the quality of this paper.

References

[1] E. Rescorla and B. Korver, "Guidelines for Writing RFC Text on Security Considerations," IAB, RFC 3552(BCP), Jul 2003.

[2] A. Cooper, H. Tschofenig, B. Aboba, J. Peterson, J. Morris, M. Hansen, and R. Smith, "Privacy Considerations for Internet Protocols," IETF, RFC 6973(I), Jul 2013.

[3] B. Trammel and M. Kuehlewind, "The Wire Image of a Network Protocol," IAB, RFC 8546(I), Apr 2019.

[4] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing," IETF, RFC 4984(I), Sep 2007.

[5] A. Cooper, F. Gont, and D. Thaler, "Security and Privacy Considerations for IPv6 Address Generation Mechanisms," IETF, RFC 7721(I), Mar 2016.

[6] R. Atkinson and S. N. Bhatti, "Identifier-Locator Network Protocol (ILNP) Architectural Description," IRTF, RFC 6740(E), Nov 2012.

[7] ——, "Identifier-Locator Network Protocol (ILNP) Engineering Considerations," IRTF, RFC 6741(E), Nov 2012.

[8] ——, "ICMP Locator Update Message for the Identifier-Locator Network Protocol for IPv6 (ILNPv6)," IRTF, RFC 6743(E), Nov 2012.

[9] ——, "IPv6 Nonce Destination Option for the Identifier-Locator Network Protocol for IPv6 (ILNPv6)," IRTF, RFC 6744(E), Nov 2012.

[10] T. Narten, R. Draves, and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6," IETF, RFC 4941(H), Sep 2007.

[11] F. Gont, S. Krishnan, and R. Draves, "Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6," IETF, RFC 8981(PS), Feb 2021.

[12] F. Gont, "A Method for Generating Semantically Opaque Interface Identifier with IPv6 Stateless Address Autoconfiguration (SLAAC)," IETF, RFC 7217(PS), Apr 2014.

[13] D. Phoomikiattisak and S. N. Bhatti, "End-To-End Mobility for the Internet Using ILNP," *Wireless Communications and Mobile Computing*, vol. 2019, no. Article ID 7464179, Apr 2019.

[14] R. Yanagida and S. N. Bhatti, "Seamless Internet connectivity for ubiquitous communication," in *PURBA2019, Pervasive Urban Applications Workshop (UBICOMP 2019)*, Sep 2019.

[15] R. Atkinson and S. N. Bhatti, "Optional Advanced Deployment Scenarios for the Identifier-Locator Network Protocol (ILNP)," IRTF, RFC 6748(E), Nov 2012.

[16] S. N. Bhatti, R. Atkinson, and J. Klemets, "Integrating Challenged Networks," in *MILCOM 2011 - 30th IEEE Military Communications Conf.*, Nov 2011, pp. 1926–1933.

[17] T. Li (Ed), "Recommendation for a Routing Architecture," IRTF, RFC 6115(I), Feb 2011.

[18] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis, "The Locator/ID Separation Protocol (LISP)," IETF, RFC 6830(E), Jan 2013.

[19] A. Rodriguez-Natal, L. Jakab, V. Ermagan, P. Natarajan, F. Maino, and A. Cabellos-Aparicio, "Location and identity privacy for LISP-MN," in *2015 IEEE Intl. Conf. on Communications (ICC)*, 2015, pp. 5260–5265.

[20] A. Ford, C. Raicu, M. Handley, O. Bonaventure, and C. Paasch, "TCP Extensions for Multipath Operation with Multiple Addresses," IETF, RFC 8684(PS), Mar 2020.

[21] J. Iyengar (Ed) and M. Thomson (Ed), "QUIC: A UDP-Based Multiplexed and Secure Transport," IETF, RFC 9000(PS), May 2021.