Private Routing in the Internet

Francesco Tusa
Department of Electronic and
Electrical Engineering
University College London
London, United Kingdom
francesco.tusa@ucl.ac.uk

David Griffin
Department of Electronic and
Electrical Engineering
University College London
London, United Kingdom
d.griffin@ucl.ac.uk

Miguel Rio
Department of Electronic and
Electrical Engineering
University College London
London, United Kingdom
miguel.rio@ucl.ac.uk

***Abstract*—Despite the breakthroughs in end-to-end encryption that keeps the content of Internet data confidential, the fact that packet headers contain source and IP addresses remains a strong violation of users' privacy. This paper describes a routing mechanism that allows for connections to be established where no provider, including the final destination, knows who is connecting to whom. The system makes use of inter-domain source routing with public key cryptography to establish connections and simple private symmetric encryption in the data path that allows for fully stateless packet transmission. We discuss the potential implications of real deployment of our routing mechanism in the Internet.**

***Index Terms*—privacy, routing security and privacy, source routing, network security**

I. Introduction

Although end-to-end encryption has proved considerably good to protect the confidentiality of data, the fact that IP headers are transmitted as plaintext through the network incur a significant lack of privacy. Every network provider that forwards the packets knows who the source and the destination are and can potentially perform traffic analysis, based on IP addresses, in order to track down the usage of a particular service and the entities (users) involved in the communication.

If users want to protect the confidentiality of their connections they have a set of limited choices. They can use a virtual private network service with added cost in performance and financial cost. They can also use onion routing services like ToR which also have a serious impact on performance.

This paper presents Private Routing (PR), a novel routing mechanism that allows for users to establish private connections using inter-domain source routing which makes it extremely hard for any given provider to identify the communicating entities. The paper is organised as follows: section II presents an overview of the system. Section III details how the map dissemination works. Section IV explains in detail how sessions are established. Section V describes some related work. We finish with a discussion of open questions in section VI and conclusions in section VII.

II. Overview

End points establish sessions for private communication across a sequence of domains in the Internet. Packets to initialise the session and to exchange data during the session do not identify the end points by public IP address. This makes it impossible for intermediate domains or eavesdroppers to identify who is communicating with whom or the full details of the sequence of domains forming the path between end points.

Source hosts select the path to the destination that meet the required characteristics of the session, e.g. to meet performance targets such as throughput or latency, to increase resilience to failures by avoiding shared paths for critical connections, or to avoid or include certain domains in the path for policy/administrative reasons. Once the path to the desired destination host has been determined by the source host it is encrypted so that neither the destination host nor the full path can be reverse engineered, but so that each domain can easily identify the next hop for forwarding packets towards the destination.

Our private routing scheme uses two types of encryption in the two main phases of a session. During session initialisation strong public-key cryptography [1] is used for the Encrypted Packet Route (EPR) created by the source host that contains the encrypted sequence of domain hops and the final destination host identifier. This form of encryption is secure but has two main drawbacks: there is a computational overhead for decrypting the next hop, and a large minimum length of ciphertext per hop, which potentially makes the number of bits required in the EPR impracticable for a reasonable overhead to be conveyed in every packet header during the data transfer phase of the session.

To reduce the performance impact, a lighter form of encryption for the path and destination is used during the data transfer phase rather than using the full EPR. Each domain uses its own secret encryption method using a symmetric private key to encrypt/decrypt the next hop. The resulting Encrypted Source-Destination Path (ESDP) and Encrypted Destination-Source Path (EDSP) is constructed hop-by-hop during the session initialisation phase which is then used in the headers of each packet during the data transfer phase.

PR uses inter-domain source routing based on inter-domain connectivity maps provided by extensions to BGP similar to BGP-LS [2]. These maps allow the calculation of the best routes which then trigger the establishment of sessions with a given destination. Each domain only knows the preceding and next domain and not the full path. The entire workflow

978-1-6654-4005-9/21/$31.00 ©2021 IEEE

involves three stages:

1) Inter-domain map dissemination.
2) Path calculation and session establishment using public key cryptography. These sessions assume the same inter-domain path in both directions and do not create any state in the routers. This session establishment message needs to be interpreted and updated by one router in each domain.
3) Data transfer using ESDP/EDSPs based on private symmetric keys per domain.

Private routing allows for private connections without the disadvantages of VPNs or onion routing. Users do not need to subscribe to a, possibly costly, third party service and there will not be performance penalties caused by detouring through off-path servers. The session establishment part is similar to onion routing but it is done without any network detouring. The use of source routing actually allows for improved performance as source hosts can select paths for the connections according to service performance requirements. It also does not rely on public key cryptography for every data packet. Just for the first one.

III. Map dissemination

The first step in PR is domain map propagation. The global connectivity map of PR domains is built using a link-state protocol and is sent to every device and updated accordingly, as illustrated in Figure 1. Our assumption is that this map is pushed periodically to users whenever there are inter-domain topological updates. Although this task may seem challenging we think it is perfectly feasible even today (see discussion section).

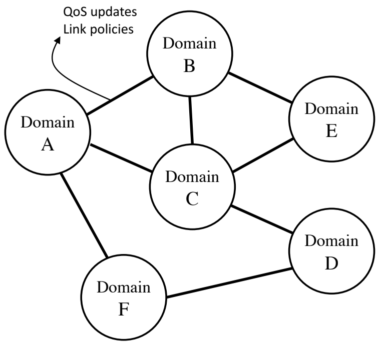

Fig. 1. Map Dissemination

The map consists of three parts:

1) Static information about the domains: country, administration, contact and the domain's public keys signed by certificate authority. Note that some edge domains who are not offering publicly-available services may not want to propagate this information to the public and choose instead to selectively disseminate it through other means only to authorised users. Note that an edge domain can prune this map before giving it to its users although it cannot guarantee that these removed links will not be used if discovered by another way.
2) Policies. It also includes the type of link (customer-provider, peer-to-peer) to provide sufficient information in order for users to avoid building routes that are not valley-free [3]. A user should never build a route that uses customer-provider peering as transit. If this is violated by a user's path selection the domain will reject the connection. Each domain may tag a particular link with given policies. These may or may not be enforceable. Enforceable policies include for example: not forward packets from domain A to domain B. Non-enforceable policies can include: do not use link 1 to reach final domain C.
3) Performance information about links and domains (e.g. link load) may be obtained or collected through parallel information systems. PR does not require domains to volunteer this information themselves which may be difficult to trust anyway. Providers like Thousand Eyes [4] should be able to provide this information on a domain-neutral basis.

We foresee that our domains are roughly equivalent to today's Autonomous Systems (ASes). Nevertheless in future developments, edge domains can establish new domains with less overhead than today's ASes, providing organizations like IANA allow it.

IV. Session Establishment

The second step in PR involves bidirectional route construction. A sender will use an algorithm, e.g. shortest-path or a variant to maximise throughput or improve resilience with latency guarantees [5], for example. It may also apply its own specific policies to avoid, e.g., certain domains or geographical regions. Note that due to this being a source-routing system it is not necessary for all users to use the same routing algorithm. After calculating the desired path the source host will construct the EPR to be used in session initialisation. In the example of Figure 2, the path to be encrypted from the source-host is $\langle$ domain A, link 2, domain B, link 6, domain C, link 7, domain D, destination-host id $\rangle$. Note that although this example uses globally unique link identifiers this is not necessary in practice. Each domain only needs to identify which of their outgoing links should be used for the next hop and these locally unique identifiers will be conveyed in the domain map used by the source host to construct the EPR.

The EPR is constructed as follows: the source-host encrypts the outgoing link id of domain A using the public key of domain A (the public key having been disseminated to the host through the domain map), which we denote as $E^p_A(2)$. E^p denotes we are using public key cryptography, the subscript of A indicates it is using the public key of domain A, and we are encrypting outgoing link identifier "2" using that key. This is repeated for each domain hop to construct the sequence of encrypted hops, with the final element of the sequence being the destination host identifier encrypted with the public key of the destination domain, D: $E^p_D(dest)$. Note that the destination host identifier does not need to be publicly addressable; an identifier local to the destination domain can be used provided

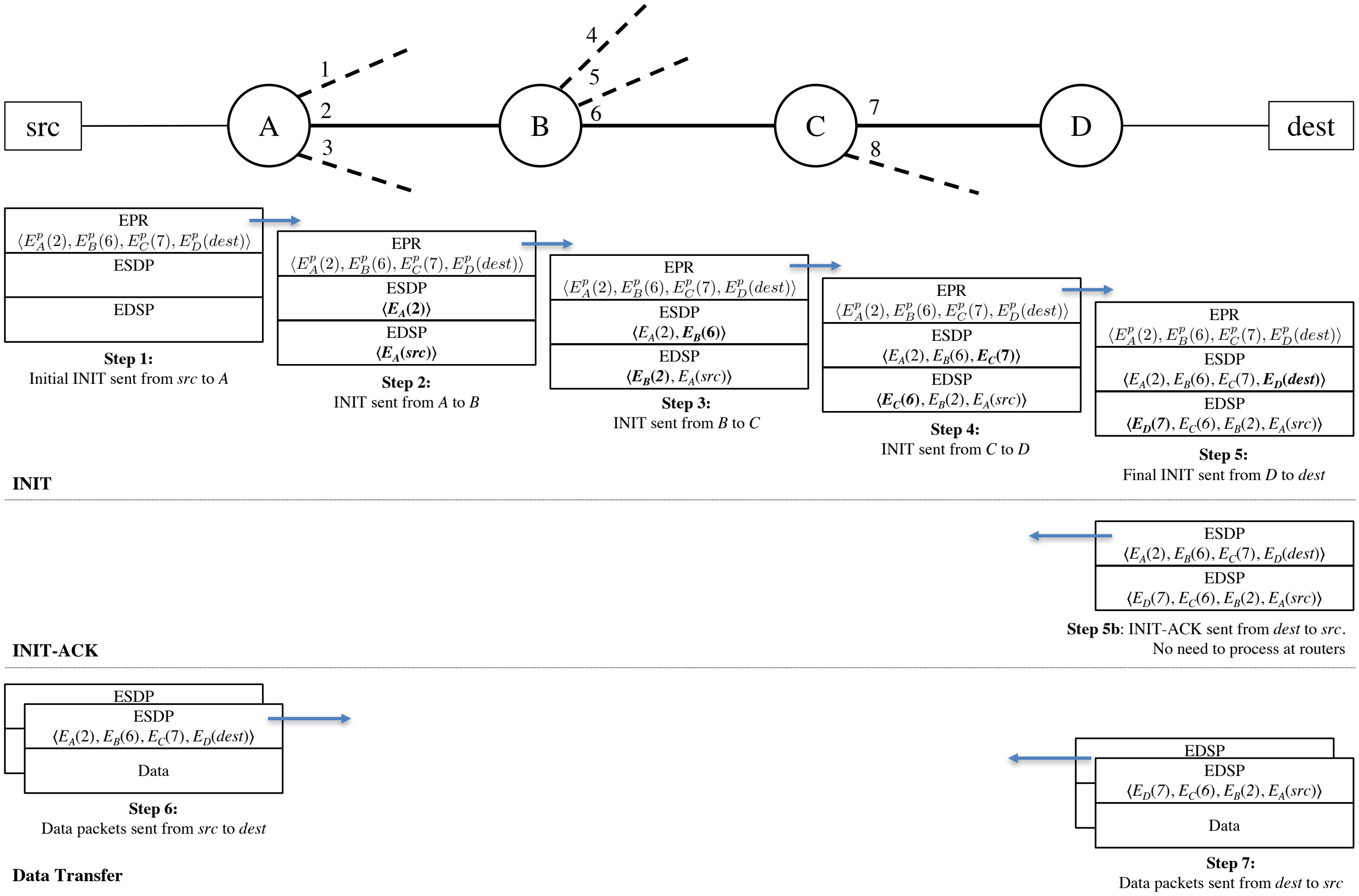
src
A
B
C
D
dest
1
2
3
4
5
6
7
8
EPR
$\langle E^p_A(2), E^p_B(6), E^p_C(7), E^p_D(dest)\rangle$
ESDP
EDSP
Step 1:
Initial INIT sent from *src* to *A*
EPR
$\langle E^p_A(2), E^p_B(6), E^p_C(7), E^p_D(dest)\rangle$
ESDP
$\langle \boldsymbol{E_A(2)}\rangle$
EDSP
$\langle \boldsymbol{E_A(src)}\rangle$
Step 2:
INIT sent from *A* to *B*
EPR
$\langle E^p_A(2), E^p_B(6), E^p_C(7), E^p_D(dest)\rangle$
ESDP
$\langle E_A(2), \boldsymbol{E_B(6)}\rangle$
EDSP
$\langle \boldsymbol{E_B(2)}, E_A(src)\rangle$
Step 3:
INIT sent from *B* to *C*
EPR
$\langle E^p_A(2), E^p_B(6), E^p_C(7), E^p_D(dest)\rangle$
ESDP
$\langle E_A(2), E_B(6), \boldsymbol{E_C(7)}\rangle$
EDSP
$\langle \boldsymbol{E_C(6)}, E_B(2), E_A(src)\rangle$
Step 4:
INIT sent from *C* to *D*
EPR
$\langle E^p_A(2), E^p_B(6), E^p_C(7), E^p_D(dest)\rangle$
ESDP
$\langle E_A(2), E_B(6), E_C(7), \boldsymbol{E_D(dest)}\rangle$
EDSP
$\langle \boldsymbol{E_D(7)}, E_C(6), E_B(2), E_A(src)\rangle$
Step 5:
Final INIT sent from *D* to *dest*
INIT
ESDP
$\langle E_A(2), E_B(6), E_C(7), E_D(dest)\rangle$
EDSP
$\langle E_D(7), E_C(6), E_B(2), E_A(src)\rangle$
Step 5b: INIT-ACK sent from *dest* to *src*.
No need to process at routers
INIT-ACK
ESDP
ESDP
$\langle E_A(2), E_B(6), E_C(7), E_D(dest)\rangle$
Data
Step 6:
Data packets sent from *src* to *dest*
Data Transfer
EDSP
EDSP
$\langle E_D(7), E_C(6), E_B(2), E_A(src)\rangle$
Data
Step 7:
Data packets sent from *dest* to *src*

Fig. 2. Session Establishment

that it has been conveyed in the map and used by the source node in the construction of the final element of the EPR.

Once encrypted, no party can decrypt the entire path and destination host identifier without access to the private keys of all domains in the path. Domain B, for example, will know the identity of the preceding domain (A) because the session initialisation request will arrive from domain A on incoming link 2, and it will discover the identity of the outgoing link (6), and hence the next hop domain (C), after it has decrypted $E^p_B(6)$ using its own private key, but it will not be able to discover the identity of further downstream domains (domain D in this example) and it will be unable to decipher the destination host identifier.

As plaintext domain identifiers are not used anywhere in the EPR, a hop counter is required to be conveyed in the initialisation packet along with the EPR. The hop counter is zero when initiated by the source host and it is incremented by each domain as it processes and forwards the EPR. When a domain receives an INIT message it uses the hop counter as the index into the sequence of hops in the EPR to identify which element it should decrypt to discover the outgoing link identifier.

After the path is determined by the source host, the session is established along the path using a first INIT message that conveys the calculated EPR between domains. The elements of the EPR are decrypted at each domain and two addresses (ESDP and EDSP) are progressively calculated and built as the INIT message traverses the path. These encrypted paths/addresses are used in all subsequent packets of the data transfer phase of the session.

ESDP and EDSP use a lighter form of encryption compared to the public key cryptography used to construct the EPR. Each domain uses its own secret encryption method and private symmetric key to substitute the plaintext outgoing link identifier with an encrypted version. Referring to Figure 2, domain A substitutes its element of the EPR, $E^p_A(2)$ with $E_A(2)$ in the ESDP, where E_A denotes it is using the private symmetric key of domain A. At the same time the reverse path is constructed - in this case domain A adds the encrypted version of the source host identifier to the EDSP: $E_A(src)$. Domain B adds the next elements of the ESDP and EDSP and so on until the destination domain is reached. Finally the destination domain forwards the INIT message to the destination host with the fully constructed ESDP and EDSP.

The private symmetric key encryption method used in each domain uses a session-specific identifier as a salt for both encryption and decryption operations. The sessionID is conveyed in the header of all data packets along with the ESDP or EDSP during the packet transfer phase. The salt is required to make mappings between plaintext and ciphertext specific to each session to avoid malicious domains or eavesdroppers building up data across multiple sessions to potentially learn plaintext to ciphertext mappings and to eventually guess the private symmetric keys used by domains.

The sessionID is constructed from a deterministic hash of the original EPR that each domain calculates when constructing the ESDP/EDSP during the session initialisation phase. While it would be possible for the source host to use a random number or nonce for the sessionID tying it to the EPR prevents malicious domains from exhaustively testing arbitrary salt values to learn plaintext to ciphertext mappings (as discussed further in section VI).

The full process is illustrated in Figure 2 where a session between source and destination hosts is being established.

- Firstly the client prepares the INIT message containing the EPR, where each hop is encrypted with the public key of the preceding domain. The hop counter is initialised to zero.
- In step 2 domain A decrypts the first element of the EPR to reveal that the next hop is over outgoing link 2. It calculates the sessionID from the hash of the full EPR and uses this as a salt for its private symmetric key encryption of the outgoing link, which it adds as the first element of the ESDP and its encryption of the source host identifier which it adds as the first element of the reverse path in the EDSP. Domain A increments the hop counter and forwards the INIT message to domain B over link 2.
- In step 3, domain B uses the hop counter as an index to see it is responsible for the second element of the EPR.It decrypts that the next hop is domain C over outgoing link 6 and adds its encrypted elements to the ESDP and EDSP using the calculated hash of the EPR as sessionID for the salt of its encryption. It increments the hop counter and forwards the INIT message to domain C.
- In step 4, domain C adds to ESDP and EDSP components of the path similarly to step 3.
- In step 5, domain D adds the destination host identifier to the ESDP and the final element of the reverse path to the EDSP and forwards the INIT message to the destination host. Now that destination has both fully constructed ESDP and EDSP addresses, it can already send packets to the source using the EDSP. The first packet returned is the INIT-ACK which is used to send the fully constructed ESDP to the source. Note that as they are fully constructed in step 5 the ESDP and EDSP addresses in the INIT-ACK do not need to be further processed by the domains. The EDSP used as the address in the header of the INIT-ACK needs to be accompanied with the sessionID, which will be used as the salt for the decryption of the next hop for forwarding the INIT-ACK in each of the domains along the reverse path.

Steps 6 and 7 represent the data transfer phase of the session.

- In step 6, the source sends packets using the ESDP and the sessionID as the address. At each hop the corresponding part of the ESDP - as indexed by the hop counter - is decrypted and the next hop domain calculated. When arriving at the destination domain the destination host identifier is decrypted and the packet is sent to the destination host.
- In step 7, the destination host sends packets to the source using the EDSP and sessionID. At each hop the corresponding part of the EDSP is decrypted and the next hop domain calculated.

By using encryption we ensure that no domain in the path knows the full list of domains in the path. Only the origin domain will know who is the sender of the packet and only the destination domain can see the destination identifier/address. It is important to note that no per-flow state is kept in the routers per session at any time, even during session establishment.

V. Related Work

Source routing has been defined for decades [6] and several works proposed to build on it. Examples include the Nimrod architecture [7], Pathlets [8], NIRA [9], MIRO [10] and [11]. In the last decade work on segment routing [12] has gained popularity and has seen some deployments. Source routing has also been deployed in data centres [13]. Adoption has been limited by security concerns [14] but these do not really apply to PRI since we use domains as the unit in our sources.

Our private source routing has similarities with Tor/onion routing [15] in the way that the full path is hidden to other routers. However, rather than implementing overlay routing as in Tor, PRI is designed as a network infrastructure protocol that allows nodes to have even more efficient routes than today.

The initial session establishment borrows some ideas from RSVP [16] and connection oriented protocols like ATM [17]. The INIT message needs to be intercepted and processed by some routers. However, this needs to be done by only one router per domain and, crucially, does not create any state in the routers.

In previous work we defined a user centric framework [18] that included the establishment of private connections but with a significant impact in router performance due to the use of per-flow state. In this paper we propose a completely different method that does not require state to be maintained by routers.

VI. Discussion and open questions

A. Security analysis

We use two forms of private addressing in our Private Routing scheme: EPRs are used in INIT messages during session initialisation and ESDP/EDSPs used in the headers of all packets during the data transfer phase of the established sessions. EPR is based on strong public-key cryptography where each element of the EPR sequence is the next hop encrypted using the public key of the domain forwarding the INIT message. Provided that the private keys of domains are not revealed, no party is able to decrypt the entire path. Guessing private keys through brute force attacks is computationally expensive and the security implications have been extensively studied in the literature [1].

The encryption scheme used for ESDPs and EDSPs depends entirely on a secret symmetric method kept private to each domain. As both encryption and decryption are undertaken by the same entity - the domain undertaking the next hop forwarding of packets - there is no need for any key to be revealed to the source or destination hosts or to any other domain. This significantly improves security while allowing for the size of ciphertext to be minimal. The algorithm for mapping plaintext to ciphertext and vice versa is kept secret and depends upon a salt - which is the sessionID in our case. Different salts will result in different mappings.

One possible attack model is that a malicious domain attempts to learn the secret mapping used by downstream domains. If this were possible then the malicious domain could observe the encrypted ESDP or EDSP and reverse the encoding to reveal the domain path and destination identity of sessions traversing its domain.

To undertake such an attack the malicious domain would need to gather sufficient data samples of plaintext and ciphertext mappings. It could gather these by initiating false sessions from its own domain and observing the encrypted next hops returned by downstream domains. However, as a salt is needed for every encryption/decryption the attacking domain would need to explore false sessions using a significant proportion of the salt range in order to guess the secret mapping algorithm. We have opted to tie the session id/salt to the destination address to avoid the possibility of malicious attackers being able to explore encodings using arbitrary salts. The sessionID/salt is determined by a well-known deterministic hash method of the full EPR. Although it is possible for attackers to craft specific salts to probe the encryption method of downstream domains this will result in INIT messages to a very wide range of destination hosts, making the attack only possible if the attacker is able to collude with a very large number of destination hosts that also represent the range of values of sessionID/salt.

One possible approach to make such attacks even more difficult would be to make the secret encryption algorithm used in each domain time-dependent. When processing the INIT messages, domains would mark the forwarded INIT message with the time-to-live (TTL) of their encryption method, which will be returned to the source in the INIT-ACK. Once the TTL expires a source would need to initiate a new INIT message to obtain the new ESDP/EDSP for the EPR. With this approach attackers would need to restart their probing and secret guessing from scratch in every TTL period.

B. Advantages and disadvantages of source routing

At the core of PR is the ability to use inter-domain source routing. This presents several additional advantages. Clients can decide for specific paths given quality of service requirements; they can establish disjoint paths with the destination to improve resilience. They can avoid particular untrustful domains. However, despite source routing being defined previously for IPv4 and IPv6, its use has been historically discouraged for security reasons. This opposition has faded in recent years with the advent of segment routing. We believe that adding privacy to the list of advantages will be a strong incentive for providers allowing its use. We see as future work ways of providers minimizing security attacks.

C. Scalability of domain map propagation

The size of the data used by PR for the inter-domain routing link-state is an important aspect to be considered. The connectivity maps need to be propagated to every client/end-hosts together with any future updates. Although at first glance this might represent a challenge, some relevant facts should be taken into account when analysing the scalability of this approach in the long-term. First of all, those maps do not need to be transmitted to all of the potential thousands of domains in the system. Furthermore, studies on BGP suggest that the required update frequency [11][12] is not very high. Finally, the number of updates due to possible failures will tend to reduce as networks become more reliable.

D. Connections within the same domain

The way packets are routed within domains is not prescribed by PR. As such, providers will be offered full flexibility for intra-domain traffic engineering.

E. Connections traversing a small number of domains

PR does not allow path privacy if both the source and the destination within a packet belong to the same domain. Moreover, privacy is compromised when less than three domains are specified within a PR path. As a workaround, for paths of two domain hops, either the source or destination domain can be duplicated in the source routing INIT message and the repeated domain would just ignore the fake hop being introduced. This will prevent the full domain path from being exposed to either of the two involved domains.

As an example, let us consider a path that traverses only domains D1 and D2, for which a user determined that the PR path should be D1-D1-D2. After decrypting the first hop, Domain 1 will find that the next domain in the list is itself (i.e., again D1). Hence, it will also decrypt the second hop in the list in order to retrieve the actual information about the next domain, namely D2.

Although Domain 2 can see that the path includes two prior hops, it will not be able to access the encrypted information and will not know that the first hop was Domain 1. As already mentioned, a one-to-one mapping between ISPs/ASs and domains is not expected. Therefore, as we anticipate that cloud providers will have their own domains, at least an additional hop would be added to the PR path enabling a further level of privacy.

F. Sticky routes may impact resilience

The set of domains involved in a PR session is established during the initial flow set-up and is only known to the originating node. Therefore, as all the intermediate network domains are not aware of the final destination, it is not feasible to reroute a connection when a network outage occurs. Sticky routes can show low resilience to failures, however, within each domain, PR allows to deal with resilience in the same way as today. As for the inter-domain resilience, end-users are much more involved in the path selection and can set-up several routes, with minimal common links, for critical applications. Since PR maintains and propagates inter-domain link state to the users, these are able to react quickly to failures that affect inter-domain paths.

G. Multicast

Multicast presents challenges from the point of view of privacy. If one wants the network to play a role in replicating packets for network efficiency it is very hard to keep this information entirely private. Given that, in practice, multicast only works in intra-domain there is little we can do to apply the principles of PR to multicast. In theory, The route definition in PR can be extended to build a inter-domain tree, keeping privacy violations limited to the user's domain but this would significantly change the way multicast works today and we leave this for future work,

H. Path asymmetry

One small limitation of our scheme is that it makes it compulsory for inter-domain routing symmetry. Packets in both directions can however use different links in each domain and different links connecting any two domains. This is a necessary implication of the destination not being aware who the source is. We believe this is not a strong limitation.

I. Anycast

Anycast as we know it becomes impossible because routing choices are made by the final users. However, if the localization of several replicas is exposed to the user somehow (e.g. through DNS) than the clients themselves can make the choice of who to connect to.

J. Practical implementation

Although the ideas on this paper can be implemented in a clean slate network, they can also be retrofitted in IPv6.

By reusing the source and destination addresses one can use 256 bits to encode the ESDP and EPSD fields. This will be more than enough to encrypt one final host identifier and several domains. If, for example one uses 64 bits for the encrypted final host identifier (more than enough for any domain in the future) we can still we can still have 8 sets of 24 bits to encode each domain. In the unlikely event that one needs more domains this can be defined in an extension header. The sessionID can be implemented in the flow label field. The hop counter will only need a small number of bits to indicate the number of domains and can be included in this field. This INIT message does not have any constraints in size since it is PDU sent between applications in adjacent domains using TCP.

Performance wise, PIR should add little impact to packet forwarding. Each INIT message needs to be processed by only one router in each domain potentially with the use of SDN packet escalation. Data forwarding adds a simple symmetric decryption to one given component of the EDSP/ESDP which should be negligible.

VII. Conclusions

This paper described a novel method to establish private connections between two end points in the Internet. Using this scheme, neither the final destination nor any domain in the middle is able to obtain the the full source/destination pair to reveal the identity of the communicating entities. The scheme relies on inter-domain source routing allowing sources to have a general choice of the connections' path, which has itself many other advantages. It relies on a soft connection established message that needs to be processed by a single router in each domain. Crucially, per-flow state is not needed for the connection. We discuss the practical implications of our scheme, concluding that there are no major roadblocks to its implementation.

VIII. Acknowledgements

The authors would like to acknowledge the support of Huawei Technologies Co., Ltd.

References

[1] R. L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM*, 21(2):120–126, February 1978.

[2] Ed. L. Ginsberg, S. Previdi, Q. Wu, J. Tantsura, and C. Filsfils. RBGP – Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions, March 2019.

[3] Sophie Y Qiu, Patrick D McDaniel, and Fabian Monrose. Toward Valley-free Inter-domain Routing. In *IEEE International Conference on Communications*, 2007.

[4] https://www.thousandeyes.com.

[5] J. Li, T. K. Phan, W. K. Chai, D. Tuncer, G. Pavlou, D. Griffin, and M. Rio. DR-Cache: Distributed Resilient Caching with Latency Guarantees. In *IEEE INFOCOM*, 2018.

[6] RFC 791 - Internet Protocol DARPA Internet Program Protocol Specification, 1981.

[7] I. Castineyra, N. Chiappa, and M. Steenstrup. RFC 1992 - The Nimrod Routing Architecture, 1996.

[8] P. B. Godfrey, I. A. Ganichev, S. J. Shenker, and I. Stoica. Pathlet Routing. In *ACM SIGCOMM*, 2009.

[9] X. Yang, D. Clark, and A. W. Berger. NIRA: a New Inter-domain Routing Architecture. *IEEE/ACM Trans. Networking*, 2007.

[10] W. Xu and J. Rexford. MIRO: Multi-path Interdomain Routing. In *ACM SIGCOMM*, 2006.

[11] X. Yang and D. Wetherall. Source Selectable Path Diversity via Routing Deflections. In *ACM SIGCOMM*, 2006.

[12] C. Filsfils, S. Previdi, B. Decraene, S. Litkowski, and R. Shakir. RFC 8402 - Segment Routing Architecture, July 2018.

[13] M. Kheirkhah, I. Wakeman, and G. Parisis. MMPTCP: A Multipath Transport Protocol for Data Centers. In *IEEE INFOCOM*, 2016.

[14] David Hoelzer. The dangers of source routing. Technical report, Enclave Forensics.

[15] https://www.torproject.org.

[16] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP), 1997.

[17] Martin De Prycker. *Asynchronous Transfer Mode. Solutions for Broadband ISDN*. Prentice Hall, 1993.

[18] M. Kheirkhah, T. K. Phan, W. XinPeng, D. Griffin, and M Rio. UCIP: User Controlled Internet Protocol. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 279–284, 2020.