Preventing Route Leaks using a Decentralized Approach: An Experimental Evaluation

Miquel Ferriol Galmés
Roger Coll Aumatell
Albert Cabellos-Aparicio
Universitat Politècnica de Catalunya
Barcelona, Spain
{mferriol, roger.coll.aumatell, acabello}@ac.upc.edu

Shoushou Ren*
Xinpeng Wei
Bingyang Liu
Huawei Technologies Co.,Ltd.
Beijing, China
{renshoushou, weixinpeng, liubingyang}@huawei.com

***Abstract*—In the inter-domain routing infrastructure, a route leak is defined as a violation of the routing policy agreed between two Autonomous Systems (AS). Route leaks have resulted in large-scale outages on the Internet, taking down several services. Although route leaks seem a simple problem, the solution is complex because: (i) ASes consider -partially- routing policy private, (ii) lack of a formal and standard language to express routing policy and (iii) BGP lacks adequate cryptographic-based security. In this paper, we present an experimental analysis of a distributed ledger-based architecture that provides a solution to route leaks. Specifically, the routing policy is unambiguously expressed using a formal language, that is then stored in a blockchain. This decentralized architecture allows private policies and interfaces seamlessly with the current BGP infrastructure, requiring no changes to routers. We build a prototype to evaluate our proposed architecture using Hyperledger, we analyze its performance using a real-world BGP dataset. Our results show that our architecture scales linearly with relevant metrics. Additionally, we validate the architecture preventing an artificially introduced route leak in a realistic 10 AS topology.**

I. Introduction

The Border Gateway Protocol (BGP) is the protocol that *glues* the Internet. BGP provides inter-domain routing to achieve reachability and path selection. However and beyond its technical capabilities, the Internet is also a business-oriented environment and as such, there is a need to express and transmit complex and rich fine-grained policies.

Unfortunately and in some cases, such policies are violated, as a notable example *route leaks* occurs when one Autonomous System (AS) violates the routing policy agreed with another AS. More specifically, when an AS incorrectly propagates a route learned from another AS. For instance in a *peering* connection ASes are expected to exchange traffic from the peering ASes or its customers. If an ASes further propagates this route then we consider it a route leak.

Route leaks are a real concern in today's Internet. First, they are quite common [1] and second, they result in important outages on Internet connectivity. As an example, Google had significant disruption in 2008 when some of its customer-facing services went down due to a route leak [5].

Route leaks seem a simple problem, but its solution is hard to find. On the one hand BGP lacks proper *deployable* cryptographic-based mechanisms at the data-plane (see for further information [6]–[8]). On the other hand, there is no standard way to communicate routing policies among ASes. At the time of this writing, this is typically done using out-of-band (website, email, etc) mechanisms. Thus, the overall inter-domain routing is prone to misconfiguration. To further exacerbate this issue, some ASes consider -partially- its routing policy as confidential and they avoid full public disclosure.

Typically, in today's Internet routing policy is signalled using BGP communities [2], a standardized extension of BGP. This is an optional attribute attached to BGP messages, used to request a specific action from a target AS. As an example, an AS can request

*Corresponding author.

This work was supported by the Spanish MINECO under contract TEC2017-90034-C2-1-R (ALLIANCE) and the Catalan Institution for Research and Advanced Studies (ICREA).

 DOI 10.1109/ICNP49622.2020.9259367

a collaborating AS not to announce a prefix through certain links.

BGP communities lack a formal definition, and the meaning of the community value is agreed among groups of collaborating ASes, often through out-of-band mechanisms. This fact, combined with the lack of proper security mechanisms protecting BGP, means that communities are error-prone and again, a source of many misconfiguration and security threats in inter-domain routing [3].

This paper is an extension of our previous work [4]. In our former work, we presented a solution to the route leak problem by addressing the two fundamental challenges of this issue. First, we defined a formal and expressive language to express, using BGP communities, routing policy. This fixes the inherent error-prone nature of BGP communities and routing policy. The formal language is transmitted securely in a blockchain-based solution. Finally, the participant ASes can download the other ASes' policy and automatically install it using standard BGP route filters. This will be used to ultimately enforce the routing policy requiring no changes to the BGP data-plane.

In this paper, we present an open-source prototype built using Hyperledger as well as its experimental evaluation. Our results show that our prototype has an end-to-end latency of a few seconds and that the architecture scales linearly with relevant metrics. Finally and to evaluate the validity of our architecture, we deploy it in a realistic 10 ASes topology using a real-world BGP policy dataset, and we prevent an artificially introduced route leak. Both the prototype and the BGP routing policy dataset can be found at [20], [24].

II. Overview of the Proposed Architecture

A. Architecture

Here we present a brief overview of the architecture, we refer the interested reader to [4] for the details. It is important to note that the main design rationale of the architecture is that it does not require any change on BGP routers, the BGP protocol or BGP communities. Our solution works seamlessly with the current inter-domain infrastructure while achieving its goals.

A step-by-step example of our architecture is shown in Figure 1. First, an expressive and formal language is defined. This language enables network operators to define the different routing policies in a standardized way. This prevents problems related to misconfigurations caused by the ambiguity or the ill-defined nature of BGP communities. Once the policy is defined, it will be uploaded to the distributed ledger (top figure).

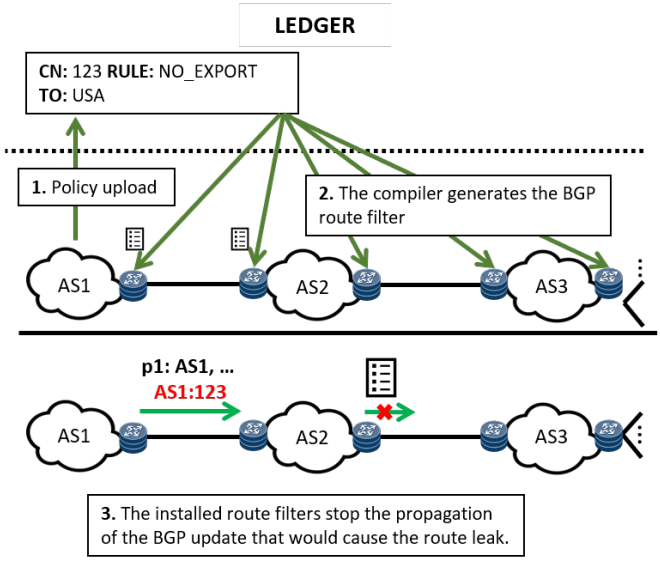

Fig. 1: Overview of the architecture.

The aforementioned ledger is shared among the participant ASes and it is used to communicate securely the different defined routing strategies. In this particular case, we expect that the business relationship that already exists between the ASes creates a motivation to participate and a certain degree of trust. Finally, the different ASes will download the routing policy and install them into BGP Routers. To do so, we provide a compiler that translates the formal language onto standard BGP route filters.

B. Proposed Formal Language for BGP Routing Policy

This section summarizes the formal language used to define the different routing policies. To make sure that the language covers a wide variety of policies, we have analyzed a set of real-world routing policy datasets [18] [25] as well as the available literature [26] [27]. Again, we refer the interested reader to [4] for the details.

The proposed language contains 5 parameters, specifically:

- ASN: The Autonomous System Number.
- CN: The Community Number.
- Rule: The policy, most common ones are LOCAL_PREFERENCE, PREPEND, NO_ANNOUNCE, NO_EXPORT.
- Value (optional): This attribute is only applied to some type of rules. Normally it is an integer and defines the quantity of a given effect.

- To (optional): what the rule refers to. When this attribute refers to a certain geographical location, the ISO 3166-1 is used.

Table I shows an example of a human-readable BGP policy definition (using BGP communities) and its equivalent in our formal language.

Natural Language
286:70 → Set Local Preference to 91
286:14 → Prepend 4 times to european peers
286:49 → Do not announce to US peers

Defined Formal Language				
ASN	CN	Rule	Value	To
286	70	LOCAL_PREFERENCE	91	-
286	14	PREPEND	4	EZ
286	49	NO_ANNOUNCE	-	US

TABLE I: Example of natural community definition inspired by a real world routing policy found in [18] and its translation to the proposed formal language

III. Experimental Evaluation

In this section, we experimentally analyze the performance of the proposed architecture. The main goal is to understand how the architecture scales with respect to relevant network parameters. For this we have built a prototype using Hyperledger [21] and Quagga [22], and deployed it in a local testbed for its analysis. For reproducibility, our prototype is open-source and can be found in [24].

A. Prototyping the Distributed Ledger

To build our distributed ledger prototype, we use Hyperledger since it fulfils all the needed requirements. The Hyperledger network consists of the following components. A ledger that registers the status of the entire system, this includes the state that describes the status of the ledger in a given moment and the blockchain that records all transactions. The chaincode that contains the logic of the system, in our case it is responsible for storing the definition of the routing policy provided by each AS. In addition, the network contains three different types of peers: (i) endorsers, responsible for the validation of a transaction (approve or reject), (ii) anchors, responsible for storing a copy of the blockchain, and (iii) orderers, they keep the ledger consistent across the network. Finally, the channel, where the different peers communicate, this allows groups or organization to create separate ledgers.

In our scenario, some ASes want to keep some policy private from other organizations on the same channel. An option for this is to create a new channel including just the organizations that need access to the data. However, creating separate channels results in additional overhead (different chaincode versions, different policies, etc). To fix this, we use private data collections, which enable a pre-defined subset of organizations on a channel the ability to endorse, commit, or query private data without having to create a separate channel.

In our prototype, we use a consensus algorithm based on a permissioned voting system. The participant ASes need to define a specific endorsing policy, this means the appropriate number of endorsers' signatures that are needed to accept a transaction. This is flexible enough to reflect the existing trust among the participant ASes.

B. Experimental Results

To analyze the performance and scalability of the prototype we run the following experiments in a local machine (Ubuntu 16.04, Intel-Core i7.4790K @4.00GHz, 8GB RAM), with negligible network latency between peers and using only one AS.

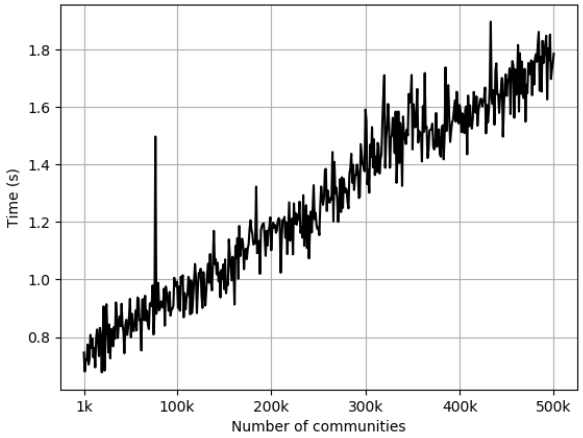

Fig. 2: Compiling time by number of communities

First, we measure the time to compile a community, this is the time it takes to read the policy from the distributed ledger and to transform onto BGP routing filters, for this we use the Quagga syntax [22]. Figure 2 shows that the time to compile the community and produce the resulting filters is linear with the number of communities.

Second, we analyze the write latency as a function of the number of endorsers. The write latency includes the time it takes to write a new policy using the formal language onto the distributed ledger. Figure 3 shows the

time of adding a new community as a function of the number of endorsing peers. The results show that the writing latency is linear with the number of endorsers, this is expected since each new endorser is an additional signature that the issuer has to verify.

Third, we analyze the chain size with respect to the number of communities stored in the chain (figure 3). As the figure shows the size grows linearly with the number of communities, our results show that even with a very large number of communities (100k) the overall size is in the order of hundreds of MBs.

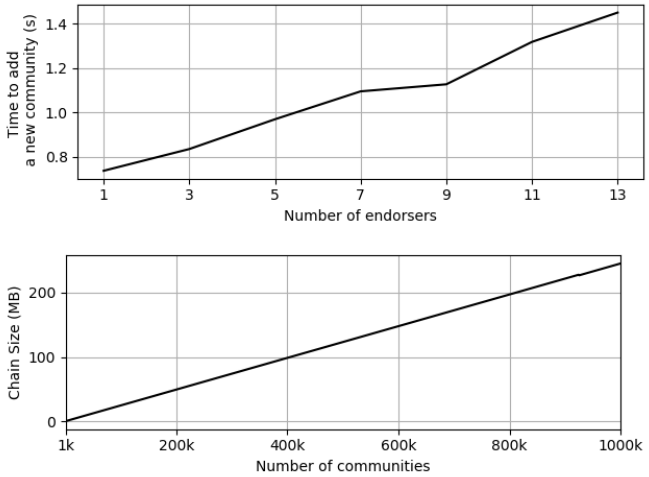

Fig. 3: On top: Total chain size by number of communities. On bottom: Write latency as a function of the number of endorser peers

Finally, the following table summarizes the scalability trends that we have found from our experiments. As the table shows the prototype shows linear scalability with respect to relevant network metrics, demonstrating its feasibility.

Variable 1	Variable 2	Relationship
Chain size	Number of communities	Linear
Time to add a new community	Number of endorsers	Linear
Compiling time	Number of communities	Linear

TABLE II: Scalability trends of our experimental prototype

IV. Preventing and Measuring Route Leaks in a Realistic Topology

To validate the proposed approach, we evaluate the prototype in a realistic topology of ASes with real-world data. The objective of the experiment is to verify if the system can detect and stop a BGP Update that would cause a route leak.

For the scenario, we deploy the Hyperledger prototype in a topology as shown in figure 5. The topology includes 10 ASes and follows a scale-free graph, a common configuration found on the Internet [23]. Each AS contains a client able to read and write a policy from the chain and an endorsing and an anchor peer. All the different ASes are on the same channel that contains a chaincode that includes different real-world communities. For the data we use a real-world dataset of routing policy, more details can be found in the next section.

A. BGP Community Dataset

We obtain the BGP routing policy dataset from [18]. This consulting firm publishes a database of BGP communities from 94 ASes. The network administrator of each AS writes in natural language the definition of the routing policy in the form of BGP communities. An example of the information contained in the dataset can be found in table III.

Community String	**Effect**
AS64497:970	Do not send route to NA(North America)
AS64497:975	Set local preference to 10 in NA.
AS64497:980	Do not send route to EU(Europe)
AS64497:985	Set local preference to 10 in EU.
AS64497:3001	Prepend AS64497 1 time.
AS64497:3010	Do not announce to AS64508

TABLE III: An example of a BGP community found in the dataset (anonymized with private AS numbers)

We transform the natural language written dataset onto the formal language described in the previous section. For this, we use the Google Cloud Natural Language API [19]. With this, we obtain a dataset written in our formal language (see section II-B) that can be parsed by our Hyperledger-based prototype.

The resulting parsed dataset written in our formal language contains 27 ASes and a total of 458 communities. Note that we do not translate all the available policies since some of the natural language used to express them is too complex to be parsed with our tool. This is not an issue since network administrators are expected to write their policy unambiguously using the formal language, and not in natural language.

Figure 4 shows the number of communities defined per AS. The figure sketches a long-tail distribution, where most of the ASes have few communities while few of them define a large set of communities. This is expected since typically large ASes have more complex policies to serve its downstream costumers. Also, the

same figure (inset) shows the different types of communities. As shown by the figure, the NO_ANNOUNCE and PREPEND are the most common ones. Such communities are common in the routing policy violation that results in route leaks. The routing policy dataset can be found at [20].

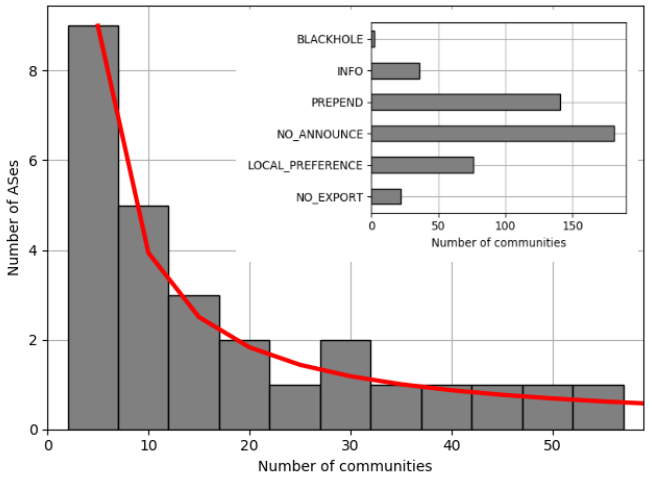

Fig. 4: Histogram of communities per AS. Inset, histogram of the types of communities

B. Experimental Results

To test if the system can detect and block a BGP Update that results in a route leak, we use multiple nodes located in different Amazon Web Services VMs. For this, we deploy our Hyperledger prototype in a realistic topology as shown in figure 5. The topology includes 10 ASes, where each AS is configured with a random set of communities from the real-world dataset. With this, we emulate a real-world scenario. In this case, we decided to use an endorsement policy that requires at least 1 signature from each of the 10 participant ASes.

In addition to this, we artificially introduce a route leak. Besides the policy obtained from the real-world dataset, AS1 also uploads a community identified as 1:333 indicating that a BGP update tagged with this community must not leak. This is stored in the distributed ledger and automatically configured as BGP route filters in the other ASes. Then, AS1 sends a BGP Update tagged with this community and the expected behaviour is that the configured BGP route filters of the other ASes block the BGP message.

To validate this behaviour we manually inspect the propagation of the BGP update message that would result in a leak. We confirm that the message is blocked by the installed BGP route filters.

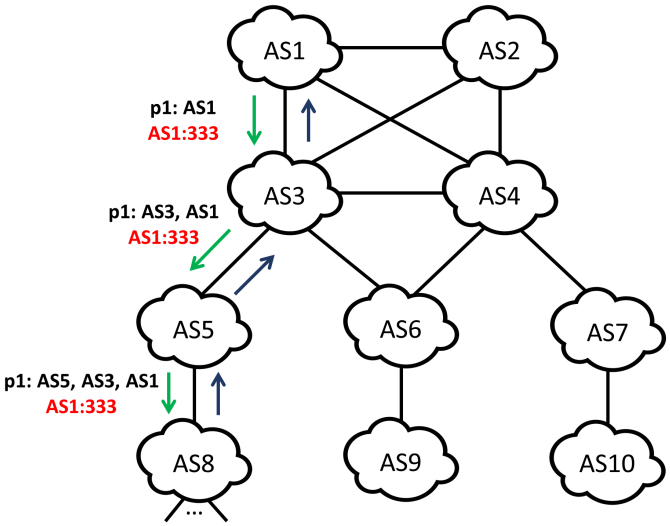

Fig. 5: Topology used for the real-world experiment

In addition to this, we also measure the end-to-end latency, specifically the time elapsed since the routing policy is written until it is compiled and installed as BGP route filters. This process involves the writing, propagation, validation and storage of the policy in the distributed ledger. We measure this for each AS in the system, obtaining $(N-1) \times N$ measurements.

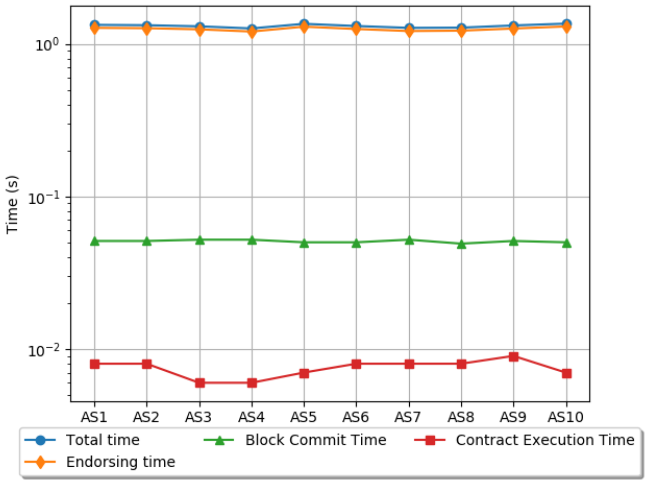

Fig. 6: End-to-End latency of the propagation time of the policies (writing, propagation and compiling) in the realistic scenario. Please note that in this experiment we set the batch timeout to 0.

Figure 6 shows the total end-to-end latency, we plot the total time as well as its decomposition of the different steps of the prototype (endorsing time, contract execution time and block time). As the figure shows the

end-to-end latency is in the order of 1 second, and -as expected- it is dominated by the endorsing time.

V. Conclusions

In this paper, we have presented an open-source prototype of a blockchain-based solution to prevent route leaks. Our experimental evaluation shows that the prototype scales linearly with respect to relevant metrics and that introduces negligible delay when you discount the communication one. Also, we show the prototype in a real-world scenario by preventing a route-leak in a 10 ASes topology.

References

[1] Goldberg, S. (2014). Why is it taking so long to secure internet routing?. Communications of the ACM, 57(10), 56-63.

[2] R. Chandra, P. Traina, and T. Li. B, "BGP Communities Attribute" IETF RFC 1997, 1996.

[3] Butler, Kevin, et al. "A survey of BGP security issues and solutions." Proceedings of the IEEE 98.1 (2009): 100-122.

[4] Ferriol, Miquel et al. "Securing Route Leaks using a Decentralized Approach" Proceedings of the IFIP Networking 2020 Conference, 2020

[5] Aftab Siddiqui, "Route Leak Causes Major Google Outage", https://www.internetsociety.org/blog/2018/11/route-leak-caused-a-major-google-outage/

[6] M. Lepinski, S. Kent "An Infrastructure to Support Secure Internet Routing", Internet Engineering Task Force (IETF), 2012

[7] M. Lepinski, Ed., K. Sriram, Ed. "BGPsec Protocol Specification", Internet Engineering Task Force (IETF), 2017

[8] Geoff Huston, An Update on Securing BGP, https://labs.ripe.net/Members/gih/an-update-on-securing-bgp

[9] Zhao, Meiyuan, Sean W. Smith, and David M. Nicol. "Aggregated path authentication for efficient BGP security." Proceedings of the 12th ACM conference on Computer and communications security. ACM, 2005.

[10] Lychev, Robert, Sharon Goldberg, and Michael Schapira. "BGP security in partial deployment: Is the juice worth the squeeze?." ACM SIGCOMM Computer Communication Review. Vol. 43. No. 4. ACM, 2013.

[11] Butler, Kevin, et al. "A survey of BGP security issues and solutions." Proceedings of the IEEE 98.1 (2009): 100-122.

[12] Zhao, Meiyuan, Sean W. Smith, and David M. Nicol. "The performance impact of BGP security." IEEE network 19.6 (2005): 42-48.

[13] Sriram, et al. "Problem Definition and Classification of BGP Route Leaks", IETF RFC 2016, 2016.

[14] BGP Leak Highlights the Fragility of the Internet with Real Consequences, https://blog.catchpoint.com/2019/06/26/ bgp-leak-internet-fragility/

[15] Siddiqui, Muhammad Shuaib, et al. "Route leak identification: A step toward making Inter-Domain routing more reliable." 2014 10th International Conference on the Design of Reliable Communication Networks (DRCN). IEEE, 2014.

[16] Streibelt, Florian, et al. "BGP Communities: Even more Worms in the Routing Can." Proceedings of the Internet Measurement Conference 2018. ACM, 2018.

[17] Donnet, B., & Bonaventure, O. (2008). On BGP communities. ACM SIGCOMM Computer Communication Review, 38(2), 55-59.

[18] BGP Community Guides, https://onestep.net/communities/

[19] Google Cloud Natural Language,https://cloud.google.com/natural-language/

[20] BGP Communities dataset, https://github.com/MiquelFerriol/SecuringBGP/tree/master/dataset

[21] Hyperledger Fabric, The Linux Foundation Projects, https://www.hyperledger.org/projects/fabric

[22] Quagga Routing Site, https://www.quagga.net/

[23] AS Core: Visualizing IPv4 Internet Topology at a Macroscopic Scale in 2017, https://www.caida.org/research/topology/as_core_network/2017/

[24] SecuringBGP, https://github.com/MiquelFerriol/SecuringBGP

[25] BGP Community Dictionary Dataset, https://www.caida.org/data/bgp-communities/

[26] Donnet, Benoit, and Olivier Bonaventure. "On BGP communities." ACM SIGCOMM Computer Communication Review 38.2 (2008): 55-59.

[27] Quoitin, Bruno, and Olivier Bonaventure. A survey of the utilization of the BGP community attribute. No. UCL-Université Catholique de Louvain. 2002.

[28] I. Shrubbery Networks. (2004). RANCID—Really Awesome New Cisco Config Differ. http://www.shrubbery.net/rancid/

[29] A. Lutu, M. Bagnulo, and O. Maennel, "The BGP visibility scan-ner," in Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS), Turin, Italy, Apr. 2013, pp. 115–120.

[30] Urbina Cazenave, Iñigo Ortiz, Erkan Köşlük, and Murat Can Ganiz. "An anomaly detection framework for BGP." 2011 International Symposium on Innovations in Intelligent Systems and Applications. IEEE, 2011.

[31] Ćosović, Marijana, Slobodan Obradović, and Ljiljana Trajković. "Classifying anomalous events in BGP datasets." 2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE). IEEE, 2016.

[32] Jin, Jian. "BGP Route Leak Prevention Based on BGPsec." 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall). IEEE, 2018.

[33] D. Gupta, A. Segal, A. Panda, G. Segev, M. Schapira, J. Feigenbaum, J. Rexford, S. Shenker, A new approach to interdomain routing based on secure multi-party computation, 11th Workshop on Hot Topics in Networks (HotNets), ACM, Redmond, Washington, USA, 2012, pp. 37–42.

[34] A. Hari, T.V. Lakshman, The internet blockchain: a distributed, tamper-resistant transaction framework for the internet, 15th Workshop on Hot Topics in Networks (HotNets), ACM, Atlanta, GA, USA, 2016, pp. 204–210.