IVeri: A Scalable Privacy-Preserving Interdomain Configuration Verification Tool via Secure Multi-Party Computation

Mingjun Fang*, Yuntao Zhao*, Qiuyue Qin*, Xing Fang*, Huisan Xu*,
Lizhao You†, Qiao Xiang*, Jiwu Shu*°
*Xiamen Key Laboratory of Intelligent Storage and Computing, School of Informatics,
Xiamen University, †School of Informatics, Xiamen University, °Minjiang University

***Abstract*—The fundamental challenge of configuration verification in an interdomain network is *privacy* because each autonomous system (AS) treats its network configuration files as private information and is not willing to share them with others. In this paper, we present IVeri, a scalable privacy-preserving interdomain configuration verification system based on secure multi-party computation. IVeri supports privacy-preserving verification and scalable verification via the following designs: (1) a verification algorithm that meets secure multi-party computation security requirements, (2) a data aggregator that reduces communication overhead while preserving privacy, (3) an algorithm that accelerates the simulation process based on routing algebra, and (4) an incremental verification algorithm that handles minor configuration changes. Extensive experiments with open-source datasets demonstrate that IVeri's optimization techniques significantly improve scalability, enabling the verification of large networks with 1000 ASes in under 3 hours, which outperforms state-of-the-art solutions.**

***Index Terms*—Network reliability, domain-specific security and privacy architectures, secure multi-party computation.**

I. Introduction

Border Gateway Protocol (BGP) [23] is the de facto routing protocol used in interdomain networks. Interdomain BGP configuration errors can cause substantial social and financial losses in interdomain networks. In October 2021, Facebook, Instagram, and WhatsApp suddenly became unreachable [3], with browsers displaying DNS errors when attempting to open them. It was believed the outage was caused by a BGP-related configuration error, which took the operators over 5 hours to fix. Cloudflare suffered an outage in 2019 that affected websites around the world [2]. According to Cloudflare, over 16 million Internet sites utilized their services for performance enhancement, DDoS mitigation, or other features. In November 2018, Nigerian internet service provider MainOne admitted that it made a configuration error during a network upgrade [4]. That error caused a disruption of key Google services by routing traffic to China and Russia.

Given these frequent and serious outages in the real world, it is a top priority for network operators to ensure the correctness of interdomain BGP configurations. Most interdomain network operators depend on query-based tools (*e.g.*, ping, traceroute, and looking glasses) and human interaction (*e.g.*, the NANOG mailing lists and the INOC-DBA phone systems). However, these tools are not only inefficient and error-prone, but also can only find errors after deployment.

Network configuration verification: preventing configuration errors before deployment using formal methods. Over the past decade, substantial progress has been achieved by network configuration verification tools [6], [11], [16], [20], [28], [32], [35]. These tools analyze network configurations before they are deployed on network devices to decide if they will compute a requirement-compliant data plane. Many companies have deployed configuration verification tools in their production networks (*e.g.*, Batfish [11] and Hoyan [32]).

Proposal: preventing interdomain network errors using privacy-preserving interdomain configuration verification. With the success of these tools [6], [11], [16], [32], [35], we propose to extend network configuration verification to prevent network failures in interdomain networks. The fundamental challenge of the proposal lies in *privacy*. Specifically, in an interdomain network, each AS considers its routing configurations as highly sensitive information and must keep them private. In contrast, existing network configuration verification tools require collecting all the routing configurations as input. A strawman solution is to deploy a configuration verification tool at a trusted third party. However, such a third party would become a single point of failure and security vulnerability, and it is also hard to find such a party in the current operation of interdomain networks.

Another design is to build a shadow BGP network among ASes where ASes can interact to check the correctness of their data plane. This simple approach appears reasonable but has the following drawbacks: (1) it is inefficient when verifying network properties under complex scenarios. For example, to verify reachability under k link failures [6], ASes have to run the shadow network C_n^k times to cover all possible failure scenarios; (2) it requires ASes to reveal their private data planes to other parties, causing information leakage.

Mingjun Fang and Yuntao Zhao are co-primary authors. Lizhao You and Qiao Xiang are co-corresponding authors.

979-8-3315-4940-4/25/$31.00 ©2025 IEEE

IVeri: a scalable privacy-preserving interdomain configuration verification tool. While the work by Xu et al. [30] addresses the fundamental privacy challenge of interdomain configuration verification, it fell short in answering several important questions, including (1) how to maintain good scalability in larger networks, and (2) how to respond to minor configuration changes quickly. To this end, we design IVeri, a scalable, privacy-preserving configuration verification tool capable of quickly responding to small configuration changes based on secure multi-party computation (SMPC). IVeri has four key designs (D1-D4):

D1: A privacy-preserving verification algorithm. Traditional BGP simulation algorithm requires the full access of all ASes' configurations, and thus it is not data-oblivious. To perform simulation in a privacy-preserving manner, we need to transform the traditional Algorithm into a data-oblivious version. Therefore, we propose a privacy-preserving control-plane simulation algorithm based on SMPC, named Data-Oblivious Simulation (DO-Simulation). We achieve this by (1) encoding key data into vectors and matrices, which are data-oblivious structures, and (2) making the control flow of the algorithm data-oblivious. After these two steps, the overall algorithm becomes an n-party SMPC algorithm that preserves the private configurations of participating ASes. Once the simulation results are obtained in ciphertext, IVeri uses a data-oblivious data-plane verification (DO-DPV) algorithm to verify the properties of the network.

D2: A privacy-preserving data aggregator scheme. Since SMPC requires every participant to participate in the computation, the communication overhead increases quadratically as the number of ASes in the network grows. To reduce this overhead, we introduce a privacy-preserving data aggregator scheme that optimizes the system. The aggregator is based on Shamir's secret sharing algorithm [24], which securely aggregates configuration data from a large number of ASes into a smaller number of privacy participants. By this method, the number of participants in SMPC is substantially reduced, thereby significantly enhancing the system's performance.

D3: Accelerating simulation using routing algebra. Based on the theory of routing algebra [25], in networks with monotonicity, algebras based on the shortest path are isotone. This implies that a locally shortest path is also the globally optimal path. Using this insight, the update order during routing updates can be determined by the distance to the destination IP prefix. Prioritizing updates for ASes with shorter distances to the destination IP prefix ensures that the update is globally optimal: once an algorithm-selected route updates an AS, no higher-priority route will update that AS. This significantly reduces the number of rounds involved in the simulation, thereby accelerating the simulation process.

D4: A quick response to minor configuration changes. In the real world, it is common to encounter configuration changes after route convergence. When encountering these scenarios, performing a complete re-simulation would lead to many unnecessary redundant computations and additional overhead, as the routing tables of many ASes remain unchanged before and after the update [26], [34]. To address this issue, we propose an incremental verification algorithm. This algorithm only updates the ASes that are affected when configuration changes occur, thus achieving fast updates.

Performance evaluation. We extensively evaluated IVeri using the CAIDA dataset and compared IVeri with the state-of-the-art privacy-preserving verification tool. IVeri's Privacy-preserving Data Aggregator and simulation acceleration algorithms significantly reduce verification time, enabling the verification of large networks with 1000 ASes within 3 hours while maintaining privacy and demonstrating scalability. Furthermore, in various incremental update scenarios, IVeri achieves up to 280× acceleration with little resource overhead.

II. Background and Motivation

A. Background

Secure Multi-Party Computation. Secure multi-party computation (SMPC) [31] addresses scenarios in which n parties $P_i, ..., P_n$ hold private inputs $x_1, ..., x_n$ and wish to compute $y = f(x_1, ..., x_n)$ in such a way that all parties learn y but no P_i learns anything about $x_j, i \neq j$, except what is logically implied by the result y and the particular input x_i that he already knew.

An SMPC algorithm consists of multiple SMPC primitives (*e.g.*, addition, multiplication and comparison). There are different ways to build SMPC primitives. In this paper, we choose to use Shamir's secret sharing [24] because it has been proven to be efficient for designing SMPC primitives involving multiple parties (*e.g.*, $n > 2$) and has good scalability [7], [21]. IVeri is specifically designed for scenarios under the semi-honest model, where all parties follow the protocol, but may attempt to infer additional information from the execution. In such settings, Shamir's secret sharing ensures that secrets remain protected unless the number of dishonest parties exceeds a security threshold. As long as a majority of parties remain honest, sensitive information is protected from exposure.

Shamir's secret sharing. A (t, n)-Shamir's secret sharing scheme allows a secret S to be split and stored at n participants such that an adversary (1) can only reconstruct S when he/she has the pieces from at least t participants; (2) cannot gain any information about S if he/she has pieces from less than t participants. Shamir's secret sharing [24] is based on polynomial interpolation. It works in two phases:

(1) *share generation*: The owner of the secret first selects a random polynomial $f(x)$ of degree $t-1$, where the secret S is carried by $f(0)$ (i.e.,$f(0) = S$). Then, the owner generates secret shares $S_i = f(i)$, where $i = 1, \ldots, n$, and sends S_i to participant i.

(2) *share reconstruction*: By collecting more than t shares, we can generate the original coefficients of the polynomial and reconstruct the secret by the Lagrange Interpolation method, as shown below

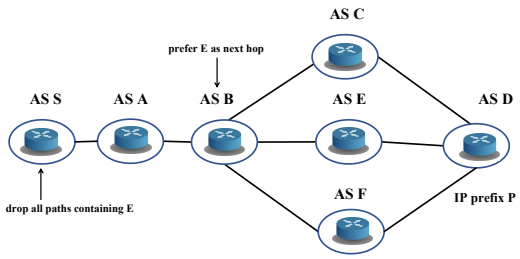


Fig. 1: An example of an interdomain network to illustrate the need for interdomain configuration verification.

$$L_j(x) = \Pi_{i=0}^{t} \frac{x-i}{j-i} mod\ p\ (i \neq j), \quad (1)$$

$$f(x) = \Sigma_{j=0}^{t} S_j L_j(x), \quad (2)$$

where t stands for the secret sharing threshold, S_j stands for the secret shares of the corresponding participant, and p is a prime number indicating the field of computation.

Data-Oblivious computation. Data-oblivious computation [12], [22] is an important concept in building complex SMPC algorithms. Given an algorithm, it is data-oblivious if and only if given any input data, the sequence of operations of this algorithm stays the same. Given an algorithm built exclusively on SMPC primitives, if it is also data-oblivious, it is also an SMPC algorithm [13]. Designing a data-oblivious algorithm is non-trivial because most data structures are not data-oblivious.

B. Motivation and Challenges

A motivating example. Consider the network in Figure 1. All AS routers are configured with default BGP configurations, with the exception of AS B which prefers AS E as the next hop, and AS S which filters out all routes containing AS E. In such a setting, it is easy to observe that AS S cannot reach the IP prefix P. In real-world operations, AS S can only find that it cannot reach P after all ASes deploy their configurations. To troubleshoot this issue, operators typically need to use query-based tools (*e.g.*, ping, traceroute, and looking glasses) and ask for help on public forums (*e.g.*, the Nanog Mailing List). During the troubleshooting time, the IP prefix P will continue to be unreachable from AS S, causing financial and social loss. In contrast, if a network configuration verification tool can be deployed in this interdomain network, AS S can find this misconfiguration before all ASes deploy their configurations, avoiding unnecessary service interruption and loss.

Although using configuration verification tools can achieve verification goals in advance and prevent losses, implementing it across domains faces the following three challenges.

Challenge 1: hard to ensure user privacy when implementing interdomain network configuration verification. When performing interdomain network configuration verification, each AS needs to provide its configuration files as input. However, in most cases, ASes consider their configuration files as private data and are unwilling to disclose their BGP configurations. Therefore, directly extending intradomain configuration verification tools to interdomain scenarios is not feasible. A practical interdomain configuration verification tool must be able to operate while keeping the ASes' configurations private. Operators' feedback to an online survey on the NANOG Mailing List [29] also corroborates this point of view.

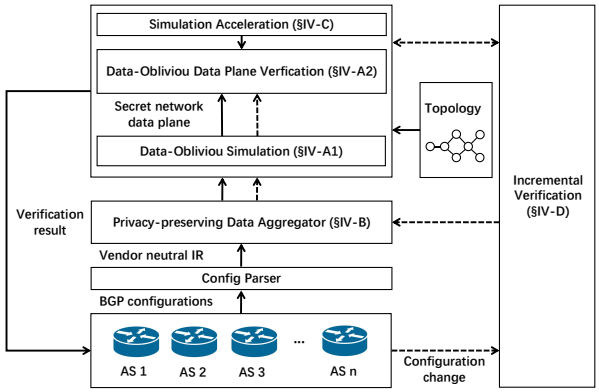


Fig. 2: The architecture and workflow of IVeri.

Challenge 2: hard to scale SMPC-based privacy-preserving verification to large interdomain networks. Interdomain networks in real-world environments are typically large, comprising many ASes. Taking the LHC science network as an example, it comprises over 150 ASes, while the global Internet, on a larger scale, encompasses more than 60,000 ASes. However, adapting verification algorithms into SMPC introduces prohibitive computational and communication overheads that scale quadratically with AS numbers.

Challenge 3: hard to quickly handle configuration changes in interdomain networks. Networks are frequently undergoing changes [27], and these changes render previous route convergence ineffective. When faced with such scenarios that require re-verifying the network, a naive approach would be to invalidate the existing routing table and re-simulate all ASes whenever configuration or topological changes occur, but this would incur significant time overhead. However, some previous work suggests that network configuration or routing changes have minimal impact on routing and forwarding tables in practice. S. Steffen et al. [26] tested various link failures in a topology with around 100 nation-level ISP nodes and found that two-thirds of the failures did not affect forwarding paths. Based on these insights, network changes do not impact all ASes and routing tables.

III. Overview

Figure 2 presents the architecture of IVeri. Each participating AS needs to select a trusted *local server* to upload its BGP configuration. These local servers install the Privacy-preserving Data Aggregator, which is responsible for converting the local configuration into Shamir secret shares and distributing them to the servers participating in the verification computation. Additionally, the ASes must jointly select the *agent servers* that will participate in the privacy-preserving computation. The number of agent servers is determined by a user-specified aggregation parameter. The agent servers receive

Algorithm 1: BGP Simulation

Input: Origin is the announcement node set of the given prefix, **N** is the set of nodes in the network, and ε indicates that only its own path is stored
Output: Best path to the given prefix of each node

Init: $\forall n \in Origin : best(n) \leftarrow \varepsilon, adv(n) \leftarrow \varepsilon$;
Init: $\forall n \in N - Origin : best(n) \leftarrow \emptyset, adv(n) \leftarrow \emptyset$;
while *true* **do**
  $E \leftarrow \{n \in N \mid adv(n) \neq \emptyset\}$;
  if $E = \emptyset$ **then**
    $break$;
  end
  $n \leftarrow pickNode(E)$;
  for ***each*** $peer \in peers(n)$ **do**
    $exports \leftarrow$ `Export(n,peer,adv(n))` ;
    $imports \leftarrow$ `Import(peer,n,exports)` ;
    $adv(peer) \leftarrow$ `Compare(best(peer),imports);`
    $best(peer) \leftarrow adv(peer)$;
  end
  $adv(n) \leftarrow \emptyset$;
end

the secret shares transmitted by the local servers, perform the verification computation, and return the results. These agent servers can either be co-located with some of the local servers or deployed in the cloud. We recommend deploying the agent servers in the cloud, as they need to exchange a large amount of encrypted data during SMPC execution. By utilizing the cloud's high bandwidth and low latency, IVeri can significantly improve its efficiency.

In the inter-domain verification scenario, all participating domains are expected to follow the protocol and provide truthful configurations. Therefore, we adopt a semi-honest security model, assuming that there are no malicious adversaries who would intentionally provide incorrect inputs. Therefore, IVeri uses a semi-honest security model, where all ASes follow the IVeri workflow specifications but may collude to share the information they acquire during the execution process. This model is sufficient for multiple scenarios of interdomain routing, including commercial Internet and collaboration science networks where member networks share resources to collaboratively conduct exascale data transfers, storage, and analytics.

In IVeri, we adopt a simulation-based verification approach (e.g., [11], [32]), where participating ASes collaboratively compute the data plane of the interdomain network using their private BGP configurations and examine the computed data plane to verify the correctness of the configurations. We do not use the SMT-based approach [6] because encoding the paths would result in significant storage overhead and increased logical complexity.

Workflow. The workflow of IVeri is also illustrated in Figure 2. IVeri offers an interface that allows operators from different ASes to define their requirements for AS paths as public information. At the start of the verification process, each AS uses a modified version of Batfish to convert its private BGP configurations into a vendor-neutral intermediate representation (IR). The key BGP attribute values in IR is

Algorithm 2: PriorityCompare (Non-DO version)

Input: r1, r2 are two routes with the same origin
Output: a boolean value indicating whether **r1** is more preferred than **r2**

Function `non-oblivious-PriorityCompare(r1, r2)`**:**
  if $r1.localpref \neq r2.localpref$ **then**
    return 1 if $r1.localpref > r2.localpref$ else 0 ;
  if $r1.aspath.length \neq r2.aspath.length$ **then**
    return 1 if $r1.aspath.length < r2.aspath.length$ else 0 ;
  if $r1.med \neq r2.med$ **then**
    return 1 if $r1.med < r2.med$ else 0 ;
  return 1 if $r1.RibId < r2.RibId$ else 0;

then encoded as integer vectors and matrices, which are sent to the AS's local server as input for the Privacy-preserving Data Aggregator. Our detailed encoded process can be found in [1]. The Privacy-preserving Data Aggregator then uses the Shamir algorithm to generate secret shares of the configuration data and distributes them to the agent servers.

After receiving secret shares, agent servers collaboratively run a data-oblivious simulation (DO-Simulation) to compute a secret (IP prefix, AS path) mapping without revealing individual inputs. The simulation prioritizes updates to ASes closest to the destination prefix. Once routing converges, agents perform data-oblivious data plane verification (DO-DPV) to check if the paths meet public operator requirements. When network changes occur, IVeri uses the incremental verification algorithm to verify network properties instead of full re-simulation. The updated configurations are locally aggregated to avoid information leakage.

Assumption. To better illustrate our design, we assume that each AS behaves as a single eBGP router in the following, which can reduce the topology complexity by ignoring the topology structure inside the AS. This assumption is reasonable since interdomain verification mainly concerns the correctness of configurations used between ASes. We are also actively working on a sound abstraction of many routers in an AS to better model the problem.

IV. Design Details

A. Privacy-Preserving Verification Algorithm

To address the challenge 1, we designed a privacy-preserving verification algorithm. Our privacy-preserving verification algorithm consists of two parts: Data-Oblivious Simulation (DO-Simulation) and Data-Oblivious Data Plane Verification (DO-DPV). First, we use DO-Simulation to obtain the RIB and FIB of each AS and then verify the network's data plane invariants through DO-DPV.

1) DO-Simulation

DO-Simulation is a privacy-preserving control plane simulation algorithm. The objective of DO-Simulation is to let AS agents collectively simulate their BGP configurations to

Algorithm 3: PriorityCompare (DO version)

Input: **r1, r2** are two routes with the same origin
Output: a boolean value indicating whether **r1** is more preferred than **r2**

Function `oblivious-PriorityCompare(r1, r2)`:
- $C0 = r1.localpref.equal(r2.localpref)$
- $C1 = r1.aspath.len.equal(r2.aspath.len)$
- $C2 = r1.med.equal(r2.med)$
- $C3 = r1.RibId.equal(r2.RibId)$
- $CompareRes =$
  - $(1 - C0) * r1.localpref.greater(r2.localpref)$
  - $+ C0 * (1 - C1) * r1.aspath.len.less(r2.aspath.len)$
  - $+ C0 * C1 * (1 - C2) * r1.med.less(r2.med)$
  - $+ C0 * C1 * C2 * (1 - C3) * r1.RibId.less(r2.RibId)$
- return CompareRes;

compute a network data plane stored as a (t, n)-Shamir's secret while keeping their configurations private.

A non-data-oblivious BGP simulation algorithm. Algorithm 1 gives the details of this algorithm for a given destination IP prefix. In the initial state, only the origin node that announces the prefix has an origination route to the given prefix. The origination route will then be put into the advertisement list and later announced to corresponding neighbors.

After initialization, the simulation iteratively selects a node with a non-empty advertisement list. The node sends route advertisements to its neighbors, which are filtered by BGP-like export/import functions. The compare function checks whether an imported route is better than the current best path. If so, the node updates its best path and advertisement list. Advertisements can either announce a new best route or withdraw an invalid one, mirroring BGP behavior. The simulation converges when all nodes' advertisement lists are empty, indicating that each has selected its best route to the prefix.

Transforming the non-data-oblivious Algorithm to a data-oblivious Algorithm. We accomplish this goal in two steps. First, we implement all basic operations (*i.e.*, addition, multiplication, and comparison) in Algorithm 1 using n-party SMPC primitives based on Shamir's secret sharing. Second, we make the control flow of this algorithm data-oblivious.

First, we leverage domain knowledge during the simulation to encode key data (*e.g.*, route map, route announcements, and RIBs) as vectors and matrices, which are data-oblivious data structures. As such, subroutines of Algorithm 1 (*e.g.*, $Import$, $Compare$ and $Export$) can be rewritten as operations of linear scans of these data structures. Second, for each conditional statement (*i.e.*, if) in Algorithm 1, we let DO-simulation execute both branches to ensure that the execution of these statements is also data-oblivious.

As an example, we compare the priority-compare in Algorithm 2 and the non-oblivious version in Algorithm 3, both of which determine the higher-priority route in the simulation. The non-oblivious version reveals secret values (*e.g.*, local preference) to select a branch, violating security guarantees. In contrast, the oblivious version replaces branching with boolean variables (*e.g.*, C0) and arithmetic operations, ensuring identical instruction sequences and indistinguishable access patterns to preserve privacy.

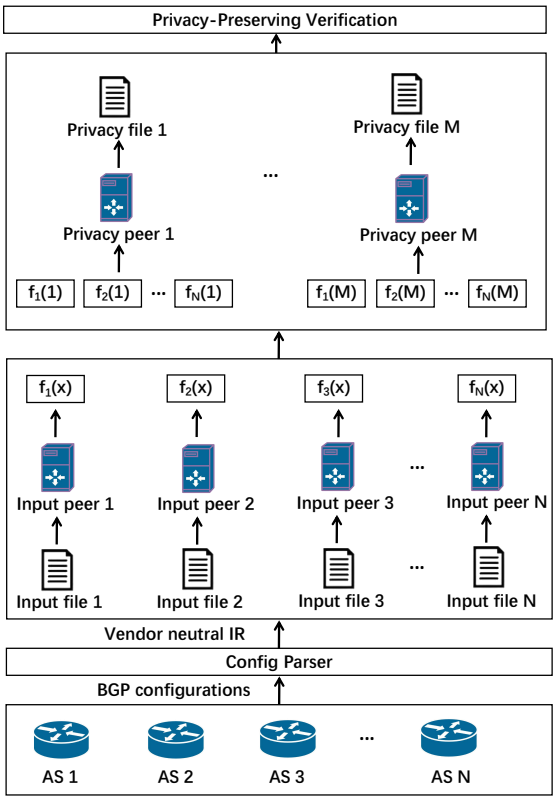


Fig. 3: Workflow of Privacy-preserving Data Aggregator

2) DO-DPV

After DO-Simulation computes a secret network data plane in an (IP prefix, AS path) mapping, distributed among privacy peers as (t, n)-Shamir secrets, the DO-DPV algorithm verifies network properties (*e.g.*, reachability, waypointing). Specifically, DO-DPV uses a DFS-based traversal algorithm to find a path bridging source to the destination AS. To accelerate DO-DPV, we construct an (IP-prefix, integer) mapping before verification process starts. As such, DO-DPV does not need to perform any IP-prefix matching but only integer comparison.

B. Privacy-preserving Data Aggregator

To address the challenge 2, we draw on the work of SEPIA [8] and design a Privacy-preserving Data Aggregator. This aggregator consolidates data from a large number of ASes before performing verification, effectively reducing the number of participants and allowing the IVeri system to scale to large interdomain networks.

Workflow of Privacy-preserving Data Aggregator. Figure 3 illustrates the workflow of the Privacy-preserving Data Aggregator. In this design, we define input peers as independent ASes that provide private BGP configurations as input, and privacy peers as servers that interact with each other to perform the Shamir secret sharing algorithm [24] on the input peers' secret data. After the Privacy-preserving Data

Aggregator aggregates N ASes into M nodes by executing the Shamir secret sharing algorithm [24], IVeri then performs the privacy-preserving verification.

The detailed process of the Privacy-preserving Data Aggregator is as follows. The operator of each AS first uses a modified version of Batfish [11] locally to convert its private BGP configuration into a vendor-neutral intermediate representation (IR). This IR is then encoded into vectors and integer matrices, as these data structures are data-oblivious, unlike the original Batfish IR objects [33]. After generating the vector/matrix format IRs, each input peer applies the Shamir secret sharing method to split its BGP configuration into M secret shares, which are then distributed to M privacy peers. Each privacy peer holds only one secret share of the input peer's private configuration data and cannot reconstruct the full content.

By default, the IVeri system is configured to operate in a semi-honest model. Through the Privacy-preserving Data Aggregator, the BGP configuration information from N ASes is aggregated into M privacy files, effectively reducing the number of participants involved in the subsequent privacy-preserving verification. Moreover, each privacy file preserves the offset of the original data, allowing accurate mapping back to the original configuration files, which ensures that error localization and diagnosis after verification remain unaffected. In implementation, we apply Shamir secret sharing with security parameters (t, n), where n is the number of privacy peers and t is set to half of n. As a result, fewer than half of the privacy peers cannot collude to reconstruct the data, thus ensuring privacy.

C. Simulation Acceleration

To address the challenge 2, we apply João Luís Sobrinho's algebraic routing theory [20], [25]. This theoretical foundation establishes that in a monotonic network, routing algebras prioritizing shorter paths are isotone, meaning that updates based on topological distance lead to globally optimal routing. The key idea is to have each AS prioritize its advertisements based on the topological distance to its neighbors, with higher priority given to those with shorter distances. By prioritizing path advertisements based on topological distance, our design inherently minimizes redundant computations, thereby reducing overhead and enhancing scalability.

However, this theory cannot be applied to non-monotonic networks. The issue arises because route withdrawal occurs when simulating based on the propagation order mentioned earlier. In non-monotonic networks, the optimal route determined by a local AS through updating the shortest path may not align with the highest-priority route according to BGP selection rules. As a result, during the subsequent BGP simulation, a higher-priority route may be selected, updating the current AS's routing table. This causes the previously advertised route to no longer be optimal for the AS. Following BGP design principles, a withdrawal operation is necessary to remove the outdated route and advertise the new highest-priority route.

To support the non-monotonic networks, we implement the route withdrawal operation. We note that in rare cases, such as when each AS prefers the route with longer paths, it may cause an efficiency decrease. However, experiments under all our datasets show that in most cases it can accelerate the simulation process by reducing the frequency of route withdrawals. We implement our design at line 8 of Algorithm 1 with the function $pickNode()$, which prioritizes updates from Autonomous Systems (AS) that have shorter paths.

Reducing the number of computations during simulation. Our accelerated algorithm also leverages publicly available knowledge of topological distance to speed up DO-Simulation. Due to the privacy protection requirements, it is not possible to directly use control flow statements to inspect updates and determine whether a given AS has undergone an update during the DO-Simulation process. To address this issue, We leveraged the topological distance data from the CAIDA dataset [9] to determine if the distance between the current AS and the source AS exceeds the current iteration round. If so, the current AS cannot be updated in this round, as it is farther from the source AS than any AS potentially updatable. This approach reduces computational load without compromising privacy, thereby accelerating the simulation.

D. Incremental Verification

To address the challenge 3, we proposes an incremental verification algorithm to handle minor changes efficiently. In practice, the number of routing table changes caused by configuration updates is limited [26], [27]. Therefore, Our approach reuses previously converged routing tables, accelerate the verification by declaring or withdrawing the affected entries in the original routing table, instead of re-running the simulation from the beginning. Our algorithm can effectively reduce the number of ASes participating in the update process, thereby accelerating the convergence of the network. Our design follows the design paradigm of the Border Gateway Protocol. When changes occur, our algorithm first identifies the affected ASes. It then applies updates to the configuration changes or topology links, and checks and updates its own route-maps and routing tables. After its updates, affected ASes will determine whether the optimal routes have changed. If the optimal routes remain unchanged, the update process will terminate. Because only the optimal routes are advertised to topologically connected neighbors.

If the optimal routes have changed, the affected AS will advertise its latest optimal routes and use the withdrawal operation to broadcast the previous optimal routes that are no longer valid due to the changes. Suppose the routes that need to be advertised are permitted to be passed on to topologically connected neighbors according to the export policy. In that case, these route updates will be sent to the neighboring ASes. The neighboring ASes will then evaluate whether the updated routes are allowed to be added to their routing tables based on their import policies. If allowed, the routes will be sorted

according to the algorithm 3 mentioned earlier to determine whether the optimal routes have changed. If an update occurs, the same process will be followed. This iteration continues until the routes converge.

E. Security Analysis

IVeri assumes a semi-honest security model with an honest majority. We formalize security by analyzing the whole IVeri workflow, which can be divided into three secure phases: (1) transformation of private configuration into secret sharing fragments through data aggregator, (2) data-oblivious verification (DO-Simulation and DO-DPV), and (3) privacy-preserving incremental verification.

Security analysis on data aggregator. The conversion of BGP configurations to a vendor-neutral intermediate representation (IR) is performed entirely within a single AS, eliminating cross-domain communication and external computation to prevent information leakage. Until the IR is input into the privacy-preserving verification algorithm, its security is ensured by the data aggregator. As the data aggregator employs Shamir's Secret Sharing, it guarantees $t - threshold$ security.

Security analysis on data-oblivious verification. For both algorithms, the primitives used for computation include addition, multiplication, greater than, less than, and equal to. We assume that the computing values are $[x]_M$ and $[y]_M$ (indicating values split into M secret sharing). In Shamir secret sharing, addition is fully homomorphic [24]. For multiplication, Shamir secret sharing supports semi-homomorphic multiplication. The equality comparison of two numbers can be reduced to comparing their difference with zero. Therefore, we subtract the two numbers to be compared and check the result against zero. For the comparison with zero, we use the method proposed by Ivan Damgård et al. [10]. We ensure that the numbers involved in the computation are much smaller than the modulus, thereby preventing wrap-around issues.

Security analysis on incremental verification. For ASes that undergo minor configuration updates, the updates will be encrypted by secret sharing and distributed among M privacy peers. Likewise, the routing tables that have previously reached convergence also stay unrevealed as secret shares. So the security of incremental verification can be reduced to that of DO-Simulation and DO-DPV.

V. Performance Evaluation

We implement a prototype of IVeri and conduct extensive evaluation. We describe our experimental setup in §V-A, and mainly study the following four questions in our evaluation: (1) How does IVeri perform across networks of varying sizes? (§V-B) (2) How does the Privacy-preserving Data Aggregator perform? (§V-C) (3) How does the simulation acceleration algorithm perform? (§V-D) (4) How does IVeri perform in incremental scenarios? (§V-E)

A. Methodology

Implementation. We implement the IVeri prototype on MP-SPDZ [19], a widely used cryptographic framework that meets the requirements of our system. In IVeri, the Privacy-preserving Data Aggregator needs to be deployed locally at each AS, and we have implemented it using Java.

Testbed. All experiments are run on five Linux servers with kernel version 5.4.0, each with two Intel Xeon Silver 4210R 2.40GHz CPUs and 128GB of DDR4 DRAM. We run multiple processes on the server to simulate communication between different hosts.

Dataset. We collect topology information and AS relationships from the open-source dataset CAIDA [9] and generate interdomain networks of different sizes based on the collected dataset for various subsequent experiments. The size of the interdomain network ranges from 6 to 1000 ASes. For each network, we consider multiple routing policies [5], [18] and synthesize the configuration of each AS based on the Gao-Rexford condition model. Meanwhile, we account for real-world special scenarios. For example, in some cases, traffic from certain ASes tends to prefer provider rather than customer. We manually add these special cases to the dataset.

Invariants. (1) **Reachability**. Reachability ensures the establishment of effective communication paths between nodes in the network. (2) **Waypoint**. Waypoint is a fundamental concept in network routing that involves the definition of specific nodes or paths within the network for achieving particular routing objectives.

Comparison. We compare IVeri with InCV [30], which also uses SMPC for interdomain network verification. Other network verification tools can't verify data plane invariants such as reachability in interdomain settings while preserving privacy. For all datasets, the verification results of IVeri are consistent with those of InCV, ensuring the accuracy of the IVeri algorithm.

Metric. When evaluating IVeri, we use total verification time and speedup ratio as our evaluation metrics.

B. Question-1: How does IVeri perform across networks of varying sizes?

The first question concerns the capability of IVeri in interdomain network verification. To answer this question, we compare the performance of IVeri with InCV and further scale up the topology to demonstrate the scalability of IVeri.

In the default configuration, IVeri aggregates the network into 3 nodes and enables the simulation acceleration algorithm. Figure 4 shows the comparison results between IVeri and InCV. As the network topology grows, the time overhead of InCV increases exponentially, and when the network expands to 50 nodes, InCV causes the server to crash due to the massive computational overhead. However, in IVeri, optimizations such as the privacy-preserving data aggregator and acceleration modules significantly reduce the computational overhead of SMPC, keeping the curve growing linearly. Additionally, for

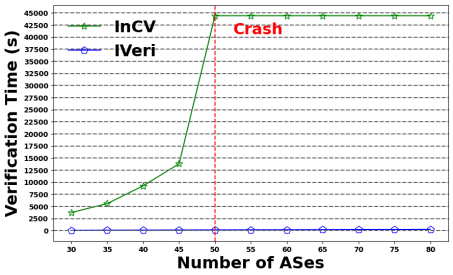
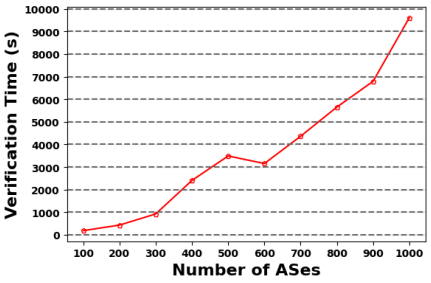
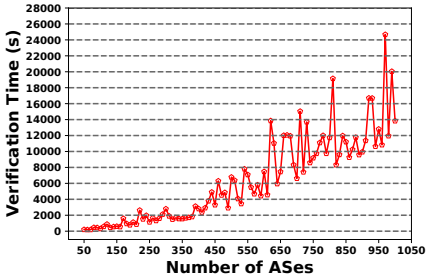


Fig. 4: The verification time of IVeri and InCV.

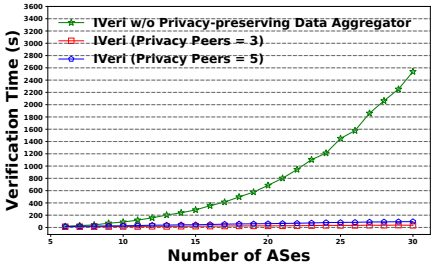


(a) IVeri completed the verification of a network with 1,000 nodes within 3 hours.

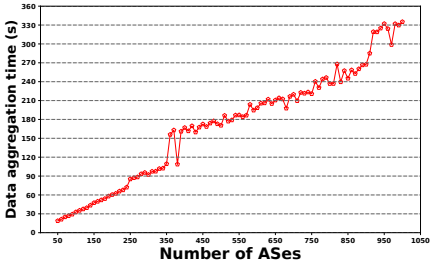


(b) IVeri quickly completes verification even when network fails to meet GAO-REXFORD conditions.

Fig. 5: The verification time of IVeri in large-scale topologies.



(a) The Aggregator significantly reduced the verification time.



(b) Even for large-scale networks, the data aggregation time remains minimal.

Fig. 6: Privacy preserving Data Aggregator performance evaluation.

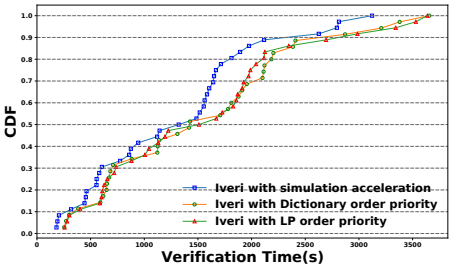
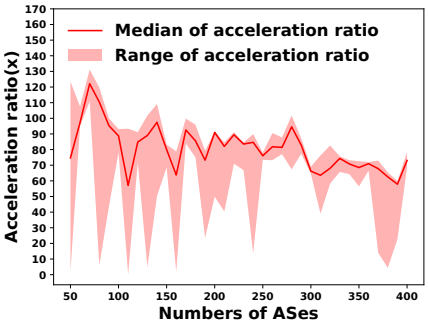
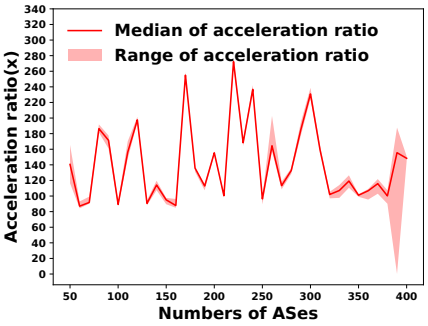


Fig. 7: IVeri's verification time with/without simulation acceleration.



(a) Removing the link scenario.



(b) Changing the local preference scenario.

Fig. 8: The acceleration ratio between incremental verification and complete re-simulation.

all topologies, the time overhead of IVeri is clearly smaller than that of InCV.

Next, we further scale up the topology to 1000 ASes to demonstrate the scalability of IVeri. Figure 5a shows the time overhead of IVeri in a large-scale network, including the total time for the data aggregation, DO-Simulation, and DO-DPV phases. IVeri completes the verification of a network with 1000 ASes within 10000 seconds, while the curve maintains a good trend of linear growth. Considering the emphasis on security in interdomain network configuration verification, we believe that the time overhead of IVeri is reasonable and that it can scale to large networks with good scalability.

To reflect real-world exceptions to Gao-Rexford conditions, we manually incorporate such cases into the dataset. For instance, some ASes may prefer providers over customers or selectively announce prefixes for traffic control. Figure 5b shows IVeri's performance under these conditions. Compared to Figure 5a, the results exhibit slight fluctuations but still follow a linear growth trend with acceptable overhead.

C. *Question-2: How does the Privacy-preserving Data Aggregator perform?*

The second question concerns whether the Privacy-preserving Data Aggregator can actually accelerate the verification of

IVeri. To answer this question, we designed experiments comparing IVeri performance with no Privacy-preserving Data Aggregator, aggregating data into 3 nodes, and aggregating data into 5 nodes. Figure 6a shows the experiment results.

The Privacy-preserving Data Aggregator critically enhances IVeri's scalability by mitigating SMPC's inherent synchronization and computation overhead across participants. As shown in Figure 6a, without aggregation, verification time grows exponentially with network topology expansion – a limitation shared with InCV. The aggregator resolves this by reducing SMPC participants through node aggregation, achieving linear time complexity reduction. This efficiency gain inversely correlates with security assurance: fewer aggregated nodes yield greater time savings but lower security. IVeri therefore implements configurable node aggregation parameters, enabling users to balance verification efficiency and security requirements through adjustable security thresholds.

We further scale up our topology to demonstrate the specific aggregation overhead of IVeri. Figure 6b shows that even for a network containing 1000 ASes, the aggregation time still only takes around 330s. Compared to the time overhead of SMPC, the aggregation time of the Privacy-preserving Data Aggregator is almost negligible, showcasing its powerful performance.

D. Question-3: How does the simulation acceleration algorithm perform?

In the third question, we focus on whether the simulation acceleration algorithm can actually speed up the verification of IVeri. To answer this question, we developed two additional control versions of IVeri that simulate transmission using dictionary order priority and local-preference order priority, respectively. For the dictionary order priority version, we modified the $pickNode()$ strategy in the Algorithm 1, changing the priority from selecting and updating the AS with the shortest path to prioritizing the AS with the lower ASN. For the local-preference order priority version, we use the route's local-preference as the criterion for determining priority, where routes with higher local-preference are given higher priority. Additionally, we removed the optimization that used topological distance to speed up the DO-Simulation.

Figure 7 shows the experimental results. For more than 80% of the topologies, the verification time of IVeri is less than 1800s, while the other two control versions can achieve this only in less than 60% of the topologies. This can be attributed to the design of the Simulation Acceleration algorithm, which decreases the number of route retractions and iterations during simulation, thus accelerating the DO-Simulation process.

E. Question-4: How does IVeri perform in incremental scenarios?

The fourth question concerns whether IVeri can maintain good performance when the network undergoes incremental updates. We primarily consider the following two common incremental scenarios: changing the local preference of an AS and removing the link between two ASes.

Figure 8 shows the experimental results. For each network, we performed 5 repeated experiments to ensure randomness and obtain the median and range of the speedup ratio. Each experiment consisted of: (1) randomly removing one link to simulate a link failure scenario, and (2) randomly selecting one AS to modify its local preference to simulate changes in its local routing policy. We compared the incremental verification time of IVeri with the complete re-simulation time. In all networks, IVeri consistently achieves a positive speedup ratio, with a maximum speedup of up to 280. The link failure scenario shows more fluctuation compared to the local preference modification scenario, as the impact of removing different links on the network simulation varies significantly. We analyzed the cases with lower speedup performance and found that routing changes in these scenarios led to a large number of updates between ASes, causing the incremental update algorithm to approach the complete re-simulation. However, such scenarios are rare in real-world networks, as routing changes typically result in only local updates.

VI. Related work

IVeri is a SMPC application under interdomain scenarios, where participants' inputs are regarded as sensitive and have to be preserved. This topic is related to interdomain routing, network control plane verification, and SMPC.

Interdomain routing. Existing interdomain routing applications try to detect routing path inconsistency [17], or perform fault detection [14] against certain peering ISPs. In IVeri, we try to achieve a more general goal: to perform global verification of multiple forwarding properties with minimal privacy leakage in BGP.

Network control plane verification. Over the past decade, substantial progress has been achieved by network configuration verification tools [6], [11], [16], [20], [28], [32], [35]. These tools analyze network configurations before they are deployed on network devices to decide if they will compute a requirement-compliant data plane. Many companies have deployed configuration verifi- cation tools in their production networks (e.g., Batfish [11] and Hoyan [32]).

SMPC in network. Seagull [15] discusses network verification under privacy concerns. Seagull focuses on detecting forwarding loops in network FIBs, while IVeri mainly targets at verifying network configurations of all participating ASes, which means IVeri has taken into account more attributes in BGP.

VII. Conclusion

We propose IVeri, a privacy-preserving interdomain network configuration verification system using secure multi-party computation. By integrating a privacy-aware data aggregator and routing algebra-based acceleration algorithms, IVeri significantly reduces communication and computational overhead

while maintaining configuration privacy across autonomous systems. The system supports efficient incremental updates for dynamic network changes and demonstrates scalability through experimental validation.

ACKNOWLEDGEMENT

We are extremely grateful to the anonymous reviewers for their constructive and insightful feedback. We also thank Franck Le, Ning Luo, Ruzica Piskac and Shuhao Zheng for their continuous support throughout the preparation and revision of this paper. This work is supported by the National Key R&D Program of China 2022YFB2901502, NSFC Grants #62172345 and #62302409, MOE of China Grant #2021FNA02008, Open Rescarch Projects of Zhcjiang Lab #2022QA0AB05, NSF Fujian China 2022J01004, and the Fundamental Reseearch Funds for the Central Universities of China Grant 20720240049.

REFERENCES

[1] https://github.com/farmerj777/IVeri-appendix.

[2] Lawrence Abrams. BGP Route Leak Causes Cloudflare and Amazon AWS Problems. https://www.bleepingcomputer.com/news/technology/bgp-route-leak-causes-cloudflare-and-amazon-aws-problems/, 2019.

[3] Lawrence Abrams. Facebook, Instagram, and WhatsApp Back Online after BGP Fix. https://www.bleepingcomputer.com/news/technology/facebook-instagram-and-whatsapp-back-online-after-bgp-fix/, 2021.

[4] David Afolayan. How bad is mainone's bgp error and why they must prevent a recurrence. https://technext24.com/2018/11/15, 2018.

[5] Ruwaifa Anwar, Haseeb Niaz, David Choffnes, Ítalo Cunha, Phillipa Gill, and Ethan Katz-Bassett. Investigating interdomain routing policies in the wild. In *Proceedings of the 2015 Internet Measurement Conference*, pages 71–77, 2015.

[6] Ryan Beckett, Aarti Gupta, Ratul Mahajan, and David Walker. A General Approach to Network Configuration Verification. In *SIGCOMM'17*, pages 155–168, New York, NY, USA, 2017. ACM.

[7] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation (Extended Abstract). In *STOC'88*, pages 1–10, New York, NY, USA, 1988. ACM.

[8] Martin Burkhart, Mario Strasser, Dilip Many, and Xenofontas Dimitropoulos. Sepia: privacy-preserving aggregation of multi-domain network events and statistics. In *Proceedings of the 19th USENIX Conference on Security*, USENIX Security'10, page 15, USA, 2010. USENIX Association.

[9] CAIDA. The CAIDA AS Relationships Dataset, 2023. https://publicdata.caida.org/datasets/as-relationships/serial-1/, 2023.

[10] Ivan Damgård, Matthias Fitzi, Eike Kiltz, Jesper Buus Nielsen, and Tomas Toft. Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation. In *Theory of Cryptography Conference*, pages 285–304. Springer, 2006.

[11] Ari Fogel, Stanley Fung, Luis Pedrosa, Meg Walraed-Sullivan, Ramesh Govindan, Ratul Mahajan, and Todd Millstein. A General Approach to Network Configuration Analysis. In *NSDI'15*, pages 469–483, Renton, WA, 2015. USENIX.

[12] Oded Goldreich. Towards a Theory of Software Protection and Simulation by Oblivious RAMs. *Journal of the ACM*, 43(3):431–473, 1987.

[13] Steven Dov Gordon, Jonathan Katz, Vladimir Kolesnikov, Fernando Krell, Tal Geula Malkin, Mariana Raykova, and Yevgeniy Vahlis. Secure Two-party Computation in Sublinear (amortized) Time. In *CCS '12*, page 513–524, New York, NY, USA, 2012. ACM.

[14] Andreas Haeberlen, Ioannis C Avramopoulos, Jennifer Rexford, and Peter Druschel. NetReview: Detecting When Interdomain Routing Goes Wrong. In *NSDI'09*, pages 437–452, Renton, WA, 2009. USENIX.

[15] Sucheer Maddury Jaber Daneshamooz, Melody Yu. Seagull: Privacy preserving network verification system, 2024.

[16] Karthick Jayaraman, Nikolaj Bjørner, Jitu Padhye, Amar Agrawal, Ashish Bhargava, Paul-Andre C Bissonnette, Shane Foster, Andrew Helwer, Mark Kasten, Ivan Lee, et al. Validating Datacenters at Scale. In *SIGCOMM'19*, pages 200–213, New York, NY, USA, 2019. ACM.

[17] Jian Jiang, Wei Li, Junzhou Luo, and Jing Tan. A network accountability based verification mechanism for detecting inter-domain routing path inconsistency. *Journal of Network and Computer Applications*, 36(6):1671–1683, 2013.

[18] Savvas Kastanakis, Vasileios Giotsas, Ioana Livadariu, and Neeraj Suri. 20 years of inferring inter-domain routing policies.

[19] Marcel Kellerl. MP-SPDZ: A Versatile Framework for Multi-Party Computation. In *CCS'20*, page 1575–1590, New York, NY, USA, 2020. ACM.

[20] Nuno P. Lopes and Andrey Rybalchenko. Fast BGP Simulation of Large Datacenters. In *VMCAI'19*, pages 386–408, Berlin, Heidelberg, 2019. Springer.

[21] Takashi Nishide and Kazuo Ohta. Multiparty Computation for Interval, Equality, and Comparison Without Bit-Decomposition Protocol. In *PKC'07*, pages 343–360, Berlin, Heidelberg, 2007. Springer.

[22] R. Ostrovsky. Efficient computation on oblivious rams. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, STOC '90, page 514–523, New York, NY, USA, 1990. Association for Computing Machinery.

[23] Yakov Rekhter and Tony Li. A border gateway protocol 4 (BGP-4), 1995.

[24] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[25] Joao L Sobrinho. An algebraic theory of dynamic network routing. *IEEE/ACM Transactions on Networking*, 13(5):1160–1173, 2005.

[26] Samuel Steffen, Timon Gehr, Petar Tsankov, Laurent Vanbever, and Martin Vechev. Probabilistic verification of network configurations. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, pages 750–764, 2020.

[27] Yu-Wei Eric Sung, Xiaozheng Tie, Starsky HY Wong, and Hongyi Zeng. Robotron: Top-down network management at facebook scale. In *Proceedings of the 2016 ACM SIGCOMM Conference*, pages 426–439, 2016.

[28] Anduo Wang, Limin Jia, Wenchao Zhou, Yiqing Ren, Boon Thau Loo, Jennifer Rexford, Vivek Nigam, Andre Scedrov, and Carolyn Talcott. Fsr: Formal analysis and implementation toolkit for safe interdomain routing. *IEEE/ACM Transactions on Networking*, 20(6):1814–1827, 2012.

[29] Huisan Xu. Network Configuration Survey. https://mailman.nanog.org/pipermail/nanog/2022-November/220861.html, 2022.

[30] Huisan Xu, Qiuyue Qin, Xing Fang, Qiao Xiang, and Jiwu Shu. Toward privacy-preserving interdomain configuration verification via multi-party computation. In *Proceedings of the 7th Asia-Pacific Workshop on Networking, APNET 2023, Hong Kong, China, June 29-30, 2023*, pages 28–33. ACM, 2023.

[31] Andrew C. Yao. Protocols for secure computations. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pages 160–164, Piscataway, NJ, 1982. IEEE.

[32] Fangdan Ye, Da Yu, Ennan Zhai, Hongqiang Harry Liu, Bingchuan Tian, Qiaobo Ye, Chunsheng Wang, Xin Wu, Tianchen Guo, Cheng Jin, Duncheng She, Qing Ma, Biao Cheng, Hui Xu, Ming Zhang, Zhiliang Wang, and Rodrigo Fonseca. Accuracy, Scalability, Coverage: A Practical Configuration Verifier on a Global WAN. In *SIGCOMM'20*, page 599–614, New York, NY, USA, 2020. ACM.

[33] Samee Zahur and David Evans. Obliv-c: A language for extensible data-oblivious computation. *IACR Cryptology ePrint Archive*, 2015:1153, 2015.

[34] Peng Zhang, Aaron Gember-Jacobson, Yueshang Zuo, Yuhao Huang, Xu Liu, and Hao Li. Differential network analysis. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 601–615, 2022.

[35] Peng Zhang, Dan Wang, and Aaron Gember-Jacobson. Symbolic Router Execution. In *SIGCOMM'22*, page 336–349, New York, NY, USA, 2022. ACM.