Privacy-Preserving Network Verification System Scalable for Internet Infrastructures

Melody Yu[1†], Jaber Daneshamooz[2]
[1]Sage Hill School, 20402 Newport Coast Dr, Newport Beach, CA 92657
[2]Department of Computer Science, University of California, Santa Barbara, CA 93106
[†]corresponding author: ocmelodyyu@gmail.com

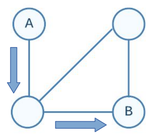
UC SANTA BARBARA
Research Mentorship Program

Abstract

- Internet infrastructure comprises of autonomous systems that interact with each other through routers
- Routers transmit data packets to each other following Border Gateway Protocol (BGP) peering protocol
- BGP peering protocol often undergoes changes, which leads to the possibility of misconfigurations which cause large-scale outages [1]
- These outages cost on average a cost of 9,000 dollars per minute and millions of dollars of revenue lost [2]
- **Network verification** allows AS administrators to test and find misconfigurations
- Current network verification systems run slowly, cannot scale to larger databases on the Internet, and risk data leaks of BGP protocol information [3]
- We aim to create a network verification system that is efficient, privacy-preserving, and scalable
- We combine graph algorithms modified for loop detection, privacy-preserving multi-party computation, and heuristics to **create an efficient and privacy-preserving network verification system**

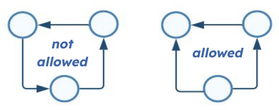
Border Gateway Protocol

- Internet: network of autonomous systems (AS)
- AS communicate and send data packets
- Routers communicate following **Border Gateway Protocol (BGP) peering protocol**
  - BGP peering determines the best path to send data

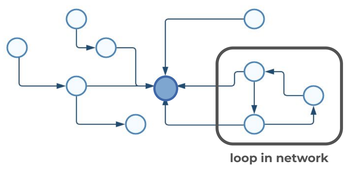


Path of sending a data packet from autonomous system A to autonomous system B

- BGP peering is susceptible to vulnerabilities
  - Misconfigurations can create **routing loops** that prevent data packets from reaching their destination

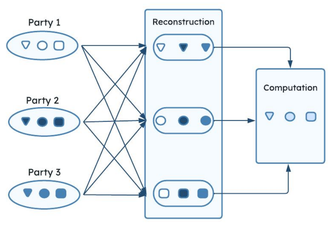


- System operates on **forwarding information base (FIB) graphs** (nodes are autonomous systems)
  - Graph should not contain any loops
  - All nodes should lead to one **destination node**



Example forwarding graph with an error

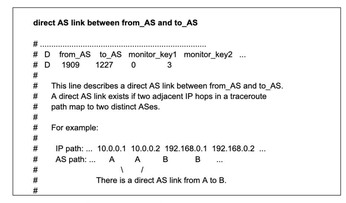
- Network verification system must be privacy preserving
  - BGP policies and configurations are often private data due to security and commercial reasons
- We use **multi-party computation (MPC)** [4] to maintain system privacy
  - Allow for a dishonest majority of parties
  - Multiple parties/servers with separate private inputs stored in separate shares
  - Information must be re-aggregated back together with all parties present for computation
- Implemented through **SCALE-MAMBA framework** [5]



Multi-Party Computation Example
Information is obfuscated and stored in separate parties which must be re-constructed back to calculate results

Graph Algorithm Benchmarking and Analysis

- Research on different algorithms for loop detection in a graph
- Build test graphs from the CAIDA: IPv4 Routed /24 AS Links Dataset [6]
  1. **AS-link graphs** from CAIDA: IPv4 Routed /24 AS Links Dataset [6] with varying nodes (each node represents an autonomous system) and edges to evaluate algorithm scalability
  2. **FIB table graphs** to test algorithm performance on Internet infrastructure snapshot data
     - Pick a random AS as the destination node
     - Running a randomized breadth-first search backwards from the destination node to generate a FIB graph



AS Link snapshot from Center for Applied Internet Data Analysis (CAIDA): IPv4 Routed /24 AS Links Dataset [6]

- Implement loop detection algorithms and experiment testbench code (input/output, execution time) in Python
- Benchmark loop detection algorithms by running each algorithm five times and calculating the average execution time
- Compare execution time to find the best loop detection algorithm for our network verification system

Efficient Graph Algorithm for Loop Detection

	Nodes	Edges	BFS	DFS	Topology	Tarjan's	DSU	Johnson
AS Link 1	280	1384	0.0019	0.0011	0.0015	0.0015	0.0572	0.8426
AS Link 2	1421	5500	0.0063	0.0054	0.0067	0.0071	0.0459	0.7315
FIB 1	25061	25066	0.0197	0.0409	0.0686	0.0666	0.0744	22.2109
FIB 2	25061	25089	0.0255	0.0542	0.0637	0.0718	0.0806	21.8804
FIB 3	25061	25071	0.0232	0.0486	0.0588	0.0686	0.1108	22.4681
FIB 4	25061	25067	0.0234	0.0426	0.0619	0.0732	0.0887	22.7568
FIB 5	25061	25079	0.0238	0.0519	0.0664	0.0729	0.0850	22.0689

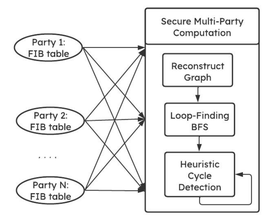
Graph algorithms' performance in detecting loops in various networks, recorded in seconds

- Graph algorithms with the smallest execution times: breadth-first search (BFS) and depth-first search (DFS)
- Depth-first search (DFS) is slightly faster in the AS-link graphs
- Modified breadth-first search (BFS) is faster on all FIB graphs
  - Due to the unique functional graph (nodes all have outdegree of one) property of FIB graphs
  - Our BFS is modified to loop through all edges instead of using a queue
- Network verification system actually runs on FIB graph, so we choose **breadth-first search** for network verification system

Proposed Network Verification System with SCALE-MAMBA

STEP 1: Transform graph for loop detection

- Read all the edges from edge list input file, splitting into private inputs and shares for separate parties that can be re-aggregated in parts during computation utilizing multi-party computation
- Re-aggregate entire graph and run breadth-first search for loop detection



Multi-party computation approach

INPUT: list of edges
(a_1, b_1)
(a_2, b_2)
...
(a_N, b_N)

where (a, b) denotes an edge between autonomous systems a and b

OUTPUT: ("YES" or "NO") indicating whether the graph contains a loop

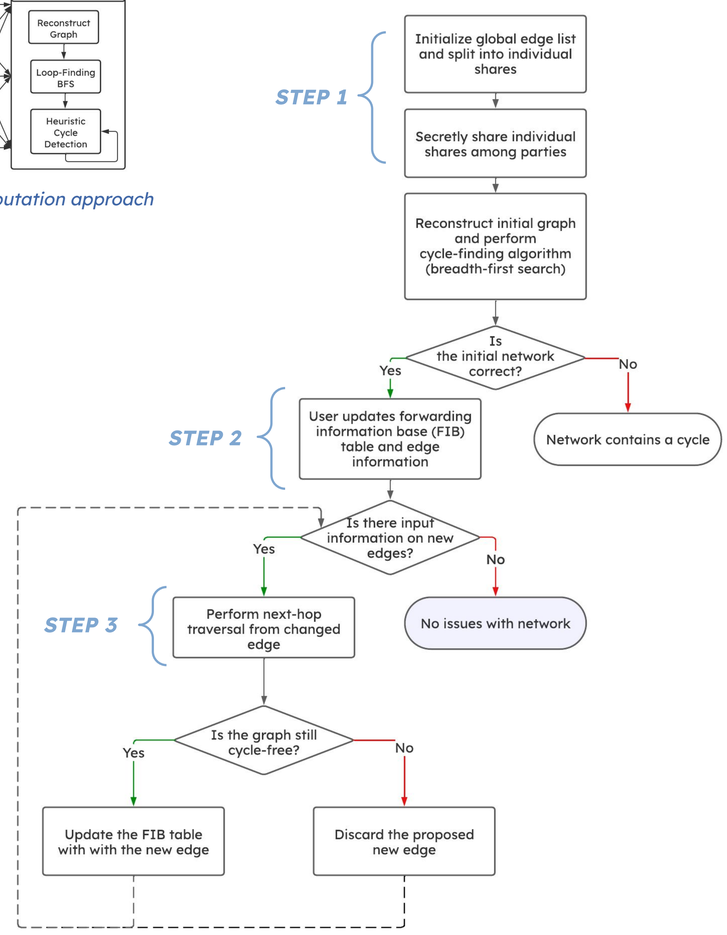
STEP 2: Allow autonomous system administrators to test changes made to BGP misconfigurations

- Allow AS administrator to update FIB table of specific node
- Read in new edge in the format *(a, b)* where AS *a* now points to AS *b*

STEP 3: Check for misconfiguration in new graph

- Re-aggregate <u>part of the graph</u>:
  - Edge changes only affect nodes directly connected to the edge
  - Only look at the edges connected to the changed protocol
- Follow the next-hop values until
  1. Reach a node marked as already visited → contains loop (incorrect)
  2. Reach 15 hops → unable to reach destination node (incorrect)
  3. **Reach destination node → no new loops created (correct)**

- If there are no misconfigurations, update the graph
- Otherwise, continue waiting for new configuration changes



Proposed network verification system logic from beginning to end

Conclusion and Future Work

- As Internet infrastructures grow in scale and complexity, network verification becomes more important than ever
- We create a network verification system that is efficient, privacy-preserving, and scalable, utilizing **breadth-first search** as our initial loop detection algorithm
- Future direction: find more privacy-preserving techniques i.e. using different multi-party computation framework

Acknowledgements:

We thank Dr. Lina Kim (University of Santa Barbara) for her valuable guidance as the Director of the Research Mentorship Program. Special thanks are extended to Zheng Ke for feedback and support throughout the project. Finally, we thank Sucheer Maddury for contributing to this project.

References:

[1] Rekhter, Y., Li, T., & Hares, S. (2006). A Border Gateway Protocol 4 (BGP-4). RFC 4271.
[2] Ponemon Institute. (2016). Cost of Data Center Outages. [Online]. Available: https://www.vertiv.com/globalassets/documents/reports/2016-cost-of-data-center-outages-11-11_51190_1.pdf. Accessed: July 15 2023.
[3] M. Blanton, A. Steele, and M. Alisagari, "Data-oblivious graph algorithms for secure computation and outsourcing," in Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security (ASIA CCS '13), May 2013, pp. 207-218.
[4] Y. Lindell and B. Pinkas, "Secure Multiparty Computation for Privacy-Preserving Data Mining," Journal of Privacy and Confidentiality, vol. 1, no. 1, pp. 59-98, 2009.
[5] Aly, A., Cong, K., Keller, M., Orsini, E., Rotaru, D., Scherer, O., Scholl, P., Smart, N.P., Tanguy, T., Wood, T.: SCALE and MAMBA v1.14: Documentation (2021) https://homes.esat.kuleuven.be/~nsmart/SCALE/
[6] The IPv4 Routed /24 AS Links Dataset - <June 24, 2023 - August 5, 2023>, https://www.caida.org/catalog/datasets/ipv4_routed_topology_aslinks_dataset/