Towards Practical Private Internet Routing using MPC

Privacy-preserving interdomain routing at Internet scale (PETS’17)
SIXPACK: Securing Internet eXchange Points Against Curious onlooKers (CoNEXT’17)

Gilad Asharov (Cornell Tech, US)
Marco Canini (KAUST, SA)
Marco Chiesa (KTH Stockholm, SE)
Daniel Demmler (TU Darmstadt, DE)
Michael Schapira (Hebrew University of Jerusalem, IL)
Thomas Schneider (TU Darmstadt, DE)
Gil Segev (Hebrew University of Jerusalem, IL)
Scott Shenker (UC Berkeley, US)
Michael Zohner (TU Darmstadt, DE)
MOTIVATION:
BGP AND ROUTING ON THE INTERNET
Motivation

The Border Gateway Protocol (BGP) connects the Internet

- Route *computation* between ISPs
- Route *dispatch* at IXPs

Issues with BGP

- Slow convergence
- Privacy

We use MPC to approach these!
Privacy-Preserving Inter-Domain Routing

BGP for **computation** of inter-domain routes for the Internet

Original Idea [GSP⁺, Hotnets’12] – Only toy example, impractical runtime

**Our Work** [ADS⁺, PETS’17] – Real-world parameters:

>51,000 autonomous systems (domains) with >196,000 connections

Topology from the CAIDA AS relation dataset

We protect the **relations** between ASes

Customer / Provider or Peering

More generic: Allow routing based on private AS **preferences**.
Privacy-Preserving Inter-Domain Routing

Centralized approach: faster & privacy issues solved by MPC

2 computational parties (CPs), running our protocol

CPs are semi-honest and non-colluding

Each AS secret-shares his relation info/preferences with the CPs
Privacy-Preserving Inter-Domain Routing

Routing based on relationship between nodes:

*Customers pay providers* to route traffic

*Peers route traffic for free*

“Economically driven” routing instead of shortest paths

High-level Neighbor Relation Algorithm:

**Plaintext input:** Topology, Target AS – Private input: EP-Relations

10 iterations for customer relation hops

1 iteration for peer hops

10 iterations for provider hops

**Private output:** for every AS the next hop to target AS
BGP Example – Notation

Provider 0 gets paid by Customer 1

Peers 2 3
BGP Example

Public network topology

Node 16 is added
BGP Example

Routes through **customers** to 16
BGP Example

Routes through peers to 16
BGP Example

Routes through **providers** to 16
Preference-Based Routing

Routing based on **export policy** and **preference** between nodes:

- ASes decide which routes are *published (exported)*
- ASes have *preferences* for their neighbors

High-level Neighbor Preference Preference Algorithm:

**Plaintext input:** Topology, Target AS – **Private input:** EP - Preferences

21 Iterations:

for all ASes:

- for all of the ASes neighbors:

  find highest **preference** neighbor with *published* route to target

**Private output:** for every AS next hop to target AS
Privacy-Preserving BGP – Circuit

Algorithm implemented as Boolean circuit evaluated with GMW

- **SIMD operations**
  - 1 Operation for multiple bits in parallel
  - Process all nodes in parallel
- **Efficient MUX with vector ANDs in GMW**
  - only 1 OT for \(n\)-bit values
- **Tree structure for depth-efficient parallel evaluation**
- **ASes evaluated in groups of similar degree**

*Algorithmic optimization:* Exclude stub nodes
BGP Stub Nodes

Stub nodes: Nodes that are *only* customers, not peers, not providers.

85% of all ASes!
BGP MPC Benchmarks – Internet Topology

Setup Phase Runtime

Online Phase Runtime

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup Phase</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Online Phase</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Full, Neighbor Pref.**
- **Full, Neighbor Relation**
- **No Stubs, Neighbor Pref.**
- **No Stubs, Neighbor Relation**
BGP MPC Benchmarks – RIR Topology

Setup Phase Runtime

Online Phase Runtime

- RIPE
- ARIN
- LACNIC
- APNIC
- RIPE-DE
- AFRINIC

Legend:
- Full, Neighbor Pref.
- Full, Neighbor Relation
- No Stubs, Neighbor Pref.
- No Stubs, Neighbor Relation
Possible Deployment

Instantiate one party with a somewhat trustworthy entity:

RIPE, DENIC, NANOG, etc. – often co-located at IXPs

Parallel Execution for fault tolerance / robustness

Software-Defined-Networking for deployment
[CDCSS, CoNEXT’17]

SIXPACK – PRIVACY-PRESERVING ROUTE DISPATCHING AT IXPS
Internet Exchange Points (IXPs)

Members (ASes) connect to IXP to exchange routing information via BGP. IXP dispatches routes based on export policies & auxiliary information.

**Problem:** Export policies & preferences are sensitive business information! Survey with 119 network operators confirm privacy & control issues.
SIXPACK: Securing Internet eXchange Points Against Curious onlookers.
We split the IXP into (at least) 2 computational parties than run SIXPACK.
Route servers are semi-honest and non-colluding.
Dispatch all routes that are allowed by the export policy of Member mA
Goal: Find a single best route for every member

Based on the private combination of

- Export Policy (as before)
- Local Preference of Members
- Congestion and other Quality of Service info from the IXP
SIXPACK: Implementation

MPC Implementation using the GMW protocol in ABY

Demonstrator in Python that simulates network members and their route announcements and withdrawals

Simulation based on a network trace from one of the largest IXPs in the world (750 members, ~10 BGP updates / sec)
SIXPACK: Runtimes

MPC implementation ready for **real-time** application!
[DSZ, NDSS’15]

ABY – A FRAMEWORK FOR IMPLEMENTING MPC PROTOCOLS
ABY – A Framework for Efficient Mixed-Protocol Secure Two-Party Computation

Framework for hybrid secure 2-party computation

Efficient Mixed-Protocol Secure Computation:

- Arithmetic Sharing
- Boolean Sharing (with the GMW protocol)
- Yao's Garbled Circuits

Separate Setup Phase (precomputable) and Online Phase
ABY – Development

Function ⟷ Circuit ⟷ Protocols ⟷ Optimizations

Idea

OT-Extension
Efficient Garbling
Fast Conversion

…
ABY – The Framework

Open-source C/C++ framework: encrypto.de/code/ABY

Many recent optimizations included
Abstracts from underlying circuit and protocol details
  Many building blocks already included: ADD, MUX, MIN, …
Efficient conversion between protocols, based on OT
Built-in performance analysis
Continuously improved and extended
Outlook

Security against malicious (active) Adversaries

Secure Multi-Party Computation ($n>2$ parties)
Summary

Privacy-Preserving **BGP Route Computation** at Internet-scale

Privacy-Preserving **Route Dispatching** at IXPs at real-world scale with practical performance

The **ABY framework** as a tool for implementing MPC protocols
Thanks for your attention!

Questions?

Contact: encrypto.de
References


