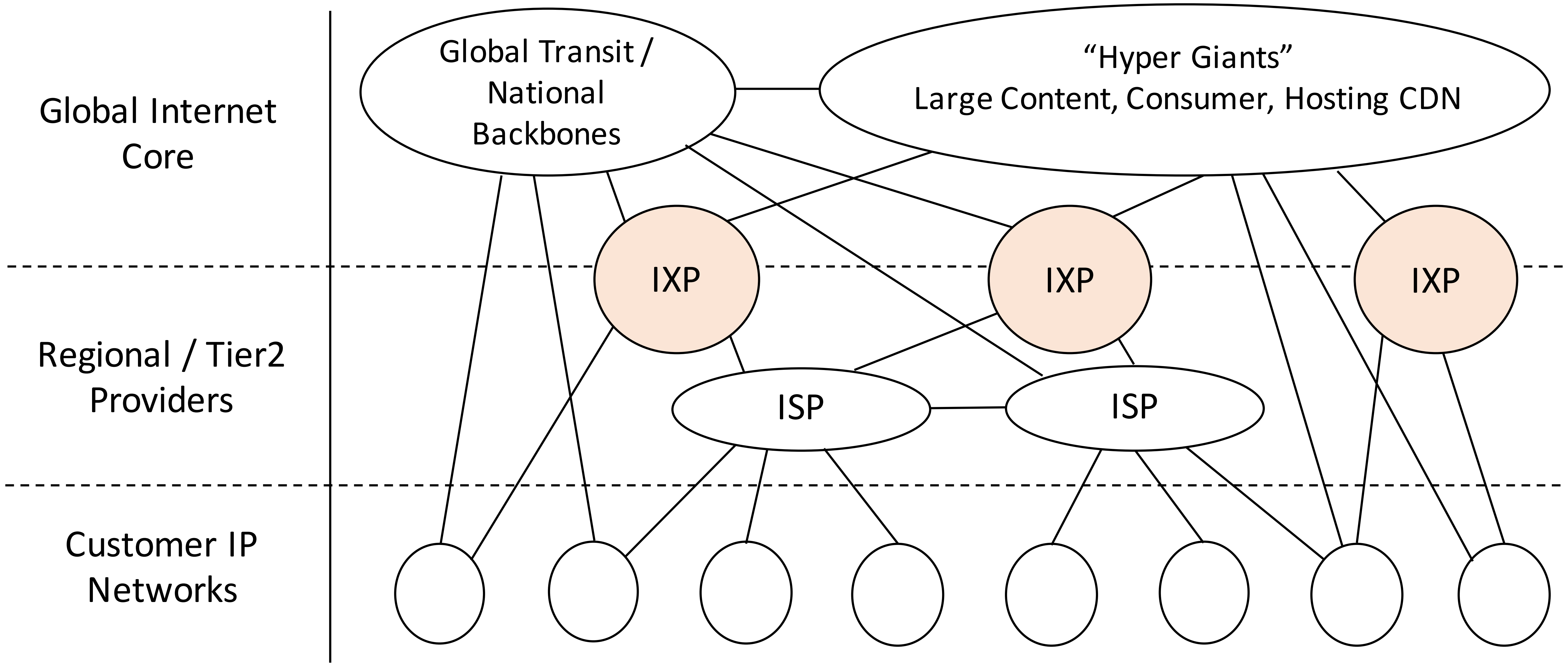
Preserving Privacy at IXPs

Xiaohe Hu[+]

Arpit Gupta[☆], Nick Feamster[☆], Aurojit Panda[*], Scott Shenker[◇]

Tsinghua University[+] PRINCETON UNIVERSITY[☆] NYU[*] Berkeley UNIVERSITY OF CALIFORNIA[◇]

Internet Exchange Points



Interdomain Ecosystem

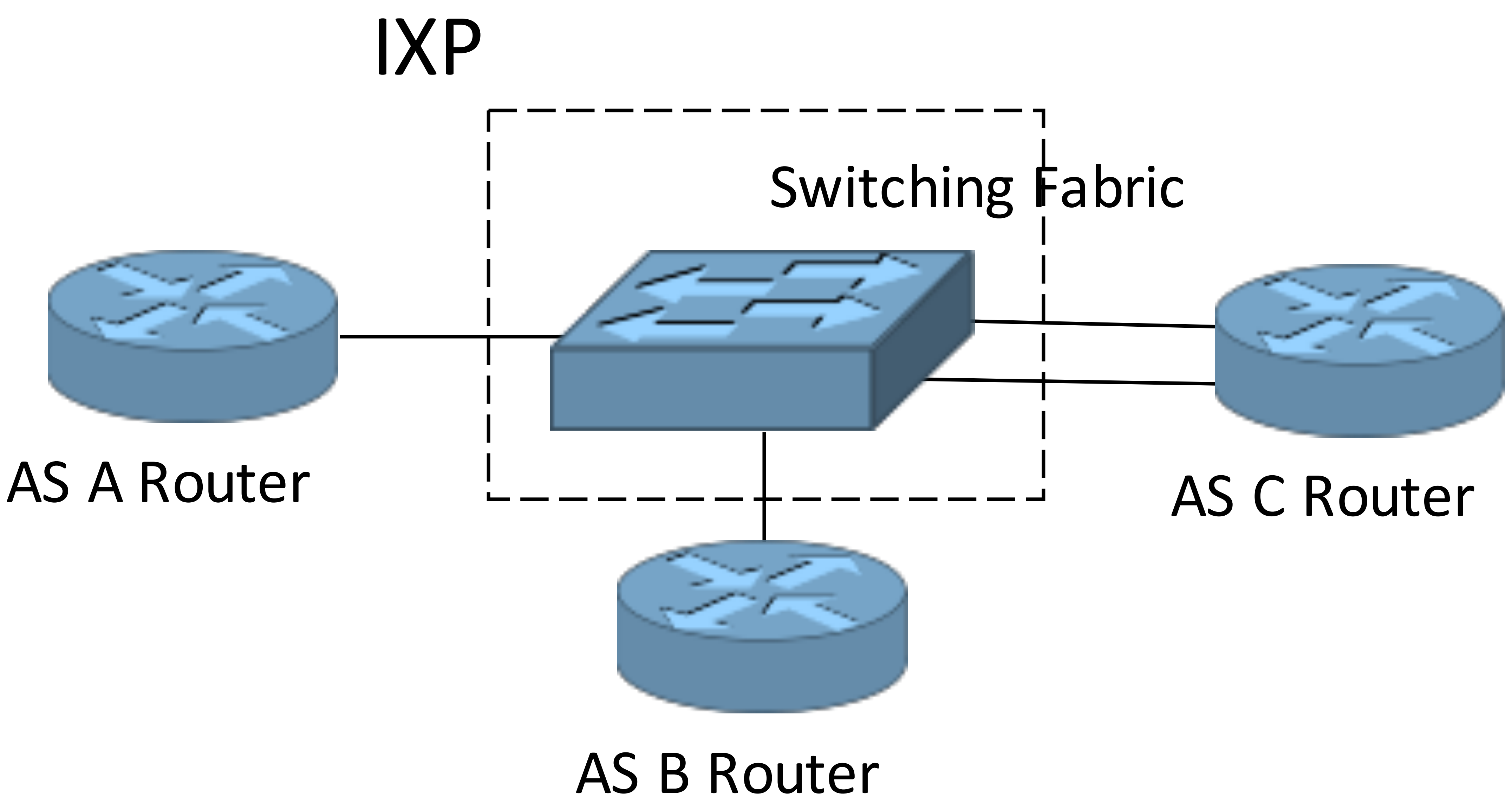
- 901 IXPs in total
- 140 new IXPs in the past year
- Large IXPs
  - 500+ AS members
  - 50K+ peering links
  - 4T+ peek traffic

Labovitz et al., *Internet Inter-Domain Traffic*, SIGCOMM 2010
http://wwww.pch.net/ixp/dir

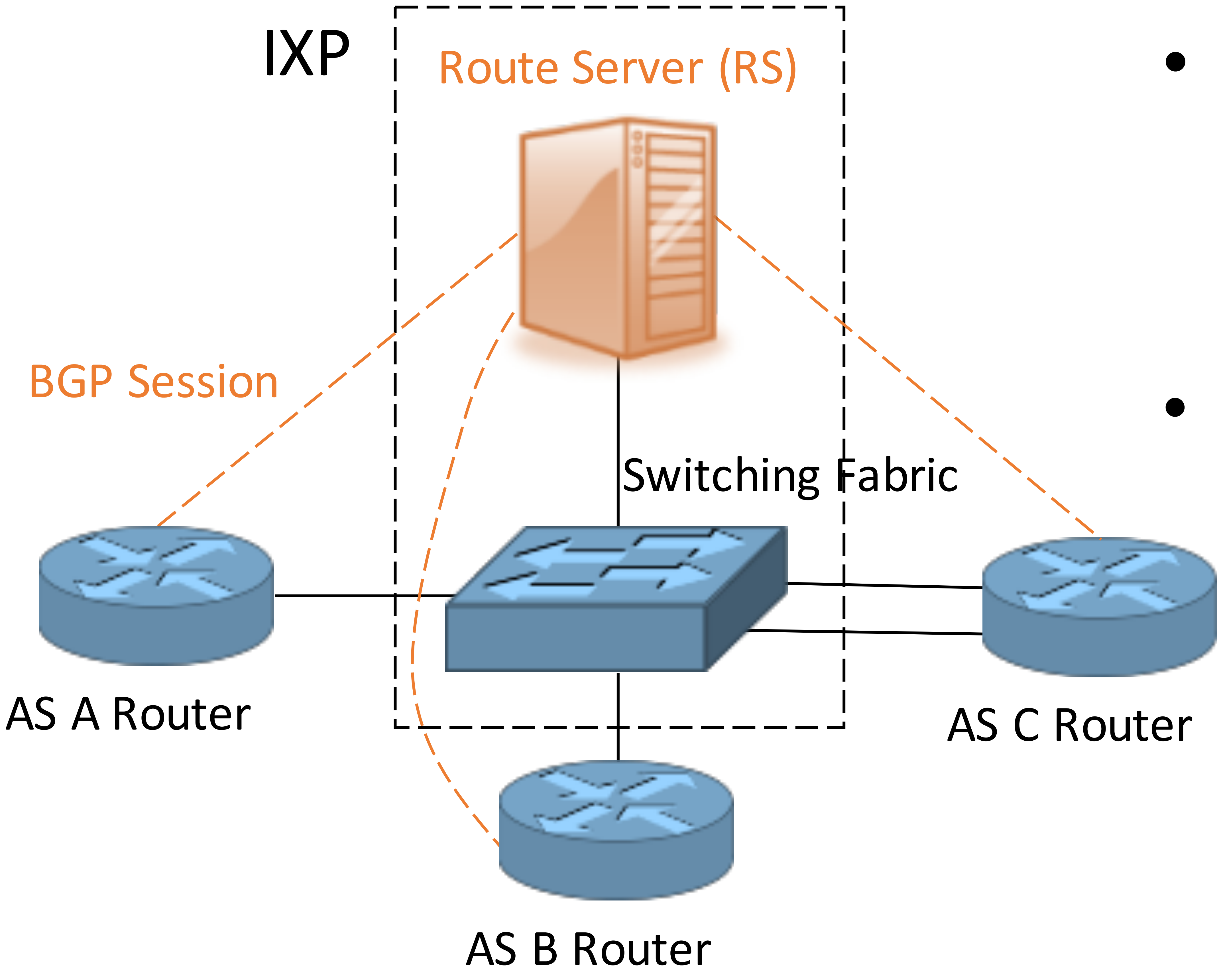
Internet Exchange Points

Scalability challenge for AS BGP Implementation

- 100s or1000s of sessions at large IXPs



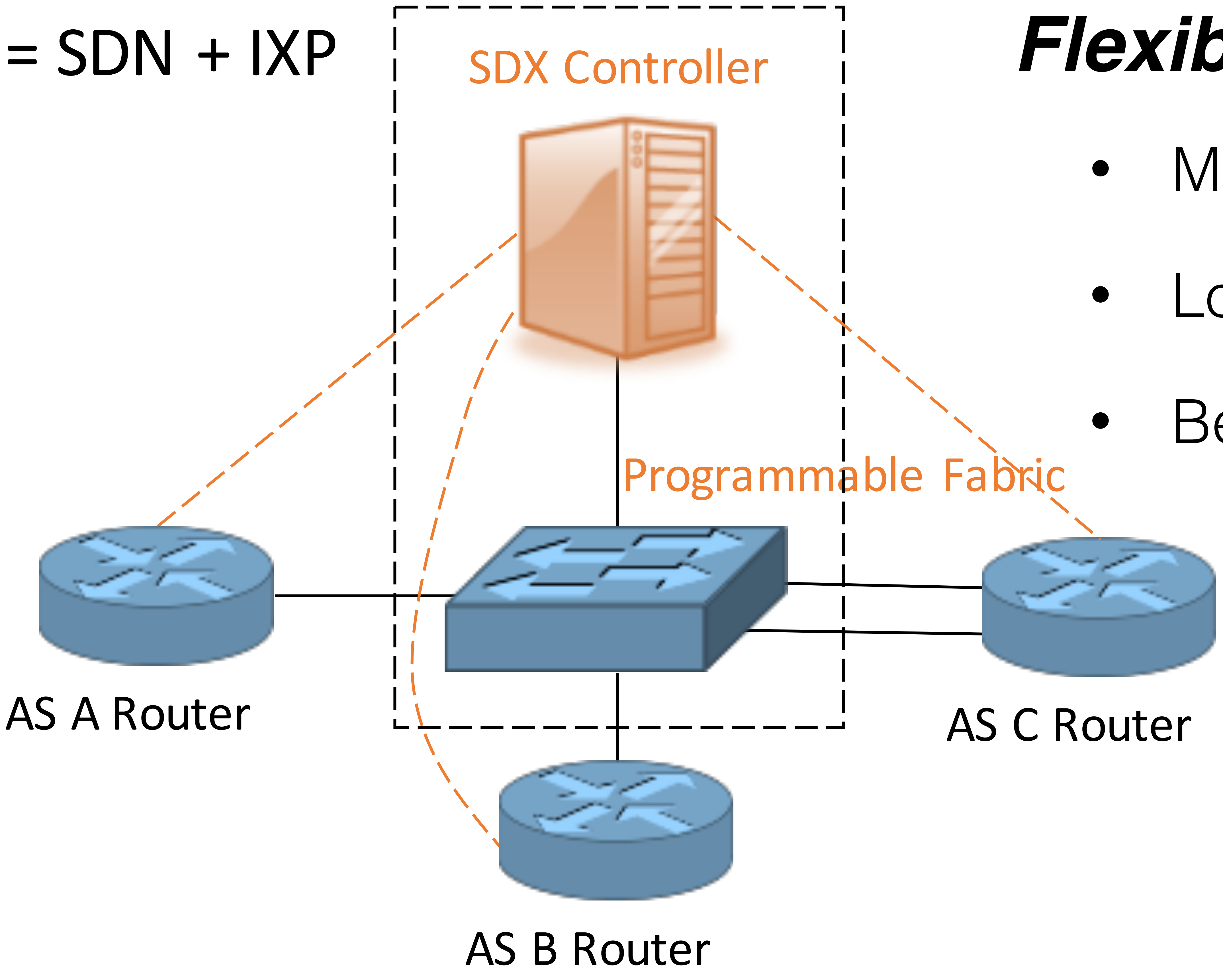
IXP Route Server



- Functionality
  - Aggregating and distributing routes
  - Executing AS policies
- Scalability
  - Sessions from O(n^2) to O(n)

IXP Route Server

SDX = SDN + IXP



Flexibility on functionality extension

- More flexible business relationships
- Load balancing and traffic engineering
- Better security applications

Privacy Concern

- AS policies are revealed to the IXP provider
    - Related to AS commercial resources, agreements and strategies
    - Backup paths, peering relationships, and local preferences on route selection
- No SLA or NDA on data confidentiality

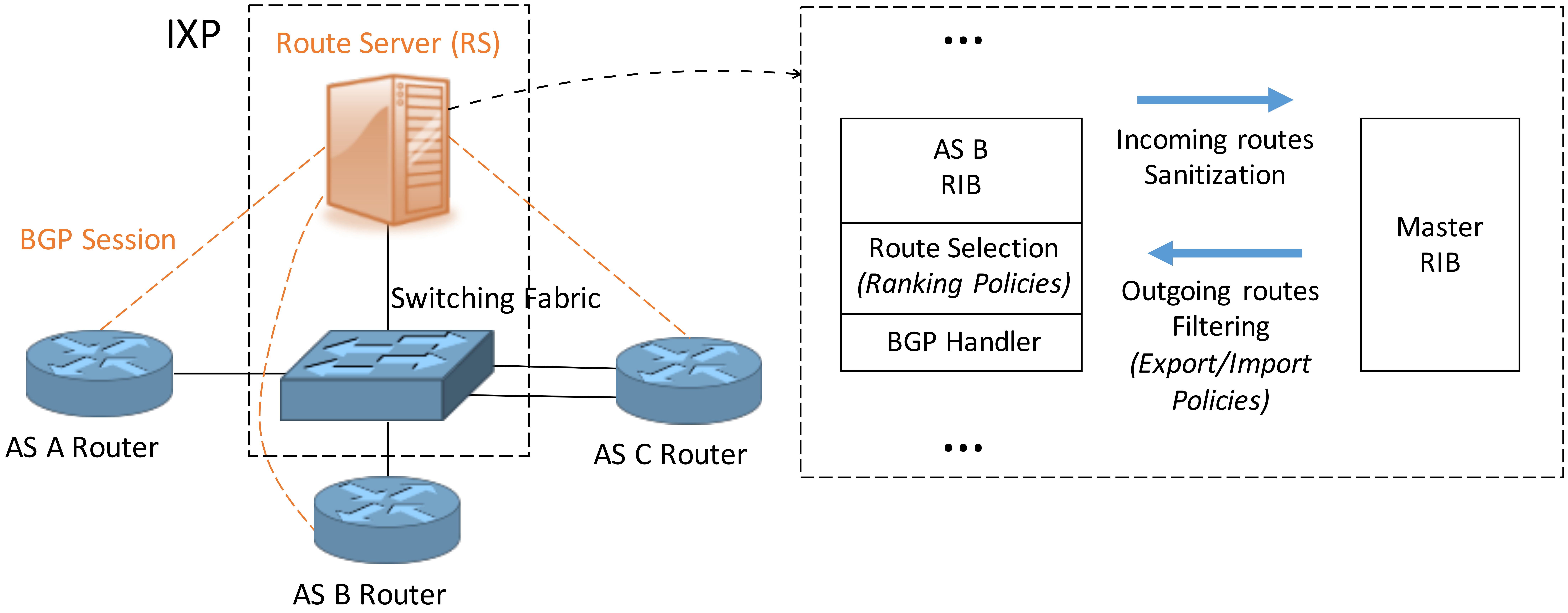
- Concern of network operators
- Impeding the widespread adoption of route servers

Problem Statement

Can we construct IXP route servers which are

- ***scalable***: increasing # of ASes at an IXP **?**
- ***flexible***: supporting functionality extension **?**
- ***privacy-preserving***: protecting AS policies **?**

Route Server Computation



Policy Privacy

Information	Publicly Visible	Route Server Visible
Route Announcements	Yes	Yes
Possible Routes (RIB)	No	Configuration Dependent
Best Route	Yes	Yes
Filtering Policy	No	Yes
Ranking Policy	No	Configuration Dependent
Auxiliary State (*e.g.* intradomain link property)	No	Configuration Dependent
Dataplane Behavior	Yes	Yes

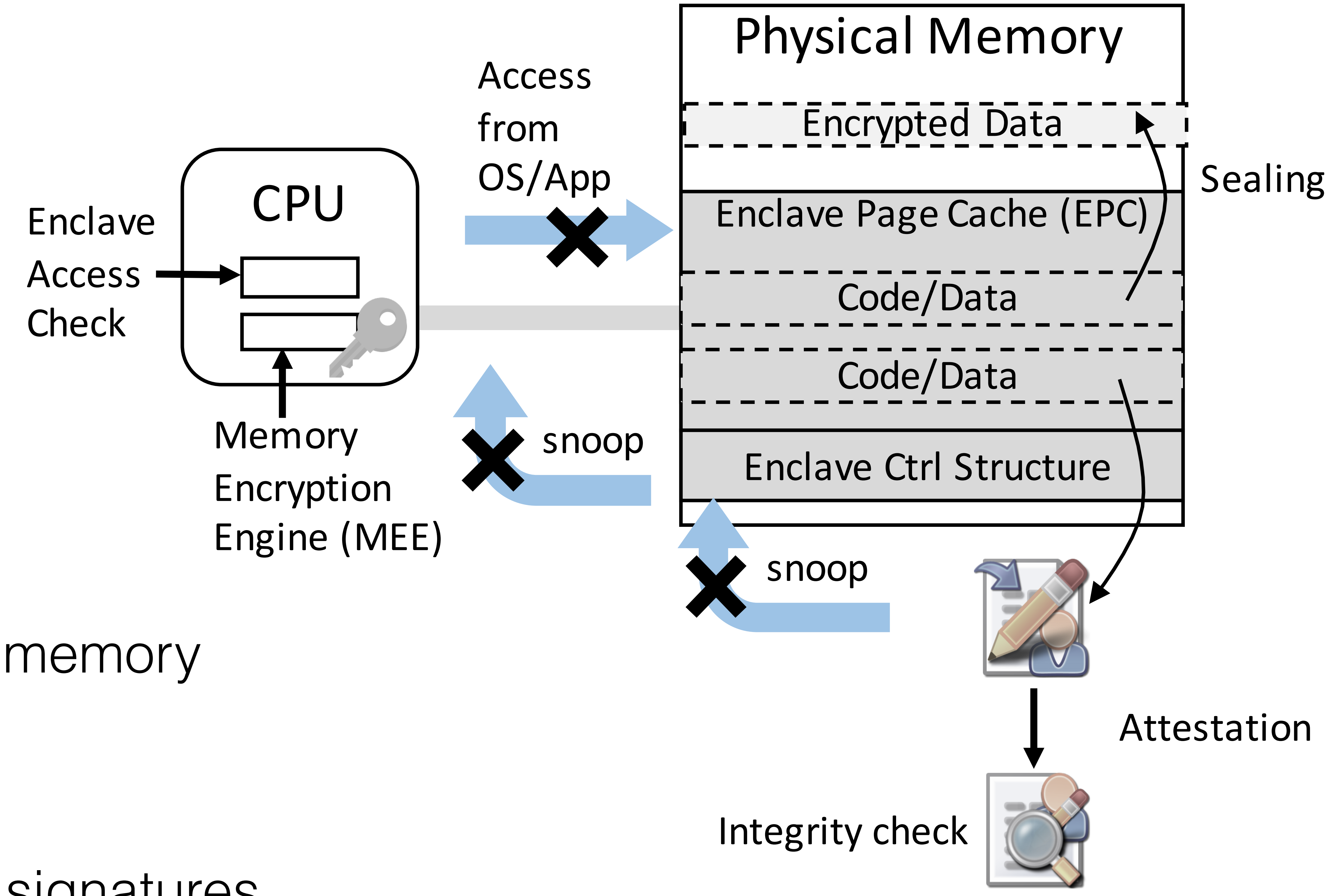
Previous Approach

- Secure Multi-Party Computation (SMPC)
  - Splitting computation across multiple non-colluding players
  - Converting computation into an arithmetic or boolean circuit
- SIX-PACK: a privacy-preserving route server using SMPC

- Limitations
  - Requiring computation outsourced to non-colluding providers
  - Two order-of-magnitude slower than the insecure approach
  - Making it harder to add functionality when minimizing computation with SMPC

Trusted Execution Environment

- A hybrid approach of system and cryptography
  - TEE processor is trusted
  - Hardware guaranteed confidentiality and integrity
  - Current commodity instances such as Intel SGX
- ***Enclave*** abstraction
  - Memory protection
    - ACL from other application accesses
    - (D)Encryption between cache<->enclave<->main memory
  - Remote attestation
    - Verifying code within enclave for remote clients by signatures



Trusted Execution Environment

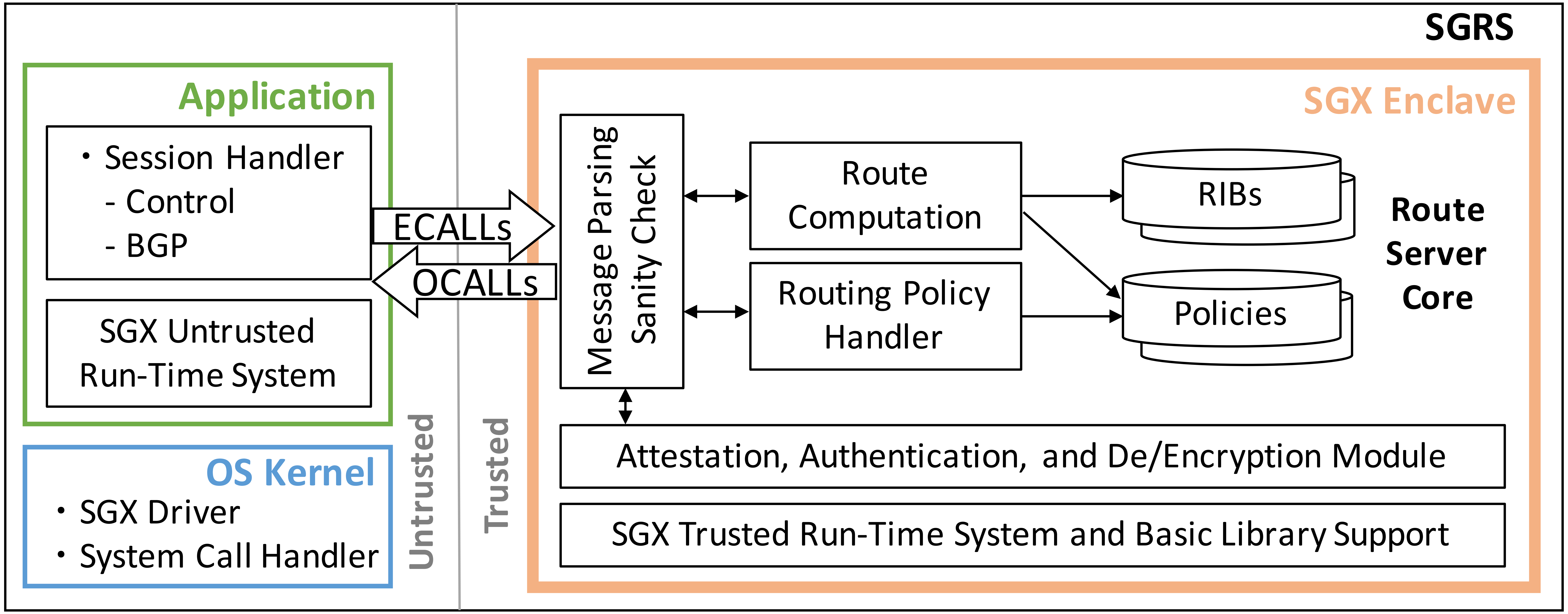
- Threat Model
  - IXPs are honest but curious
  - ASes and IXP trust the hardware vendor and TEE is correct
  - IXPs don’t use side-channel attacks

- Related Work
  - Staying in simulation stage
  - Not to centralize BGP computation

System Design

- ***Scalability: route server in real TEE platform***
  - Identify the untrusted and trusted code and data
  - Protect minimal trusted part within enclave to reduce system calls
- ***Flexibility: little restriction on route server functionality***
  - Consolidate trusted parts in one single enclave
  - Replace trusted-untrusted message passing with TEE transition calls
- ***Privacy-preserving: end to end trustworthiness and confidentiality***
  - Remote attestation, memory protection and secure channels

SGRS = SGX + Route Server



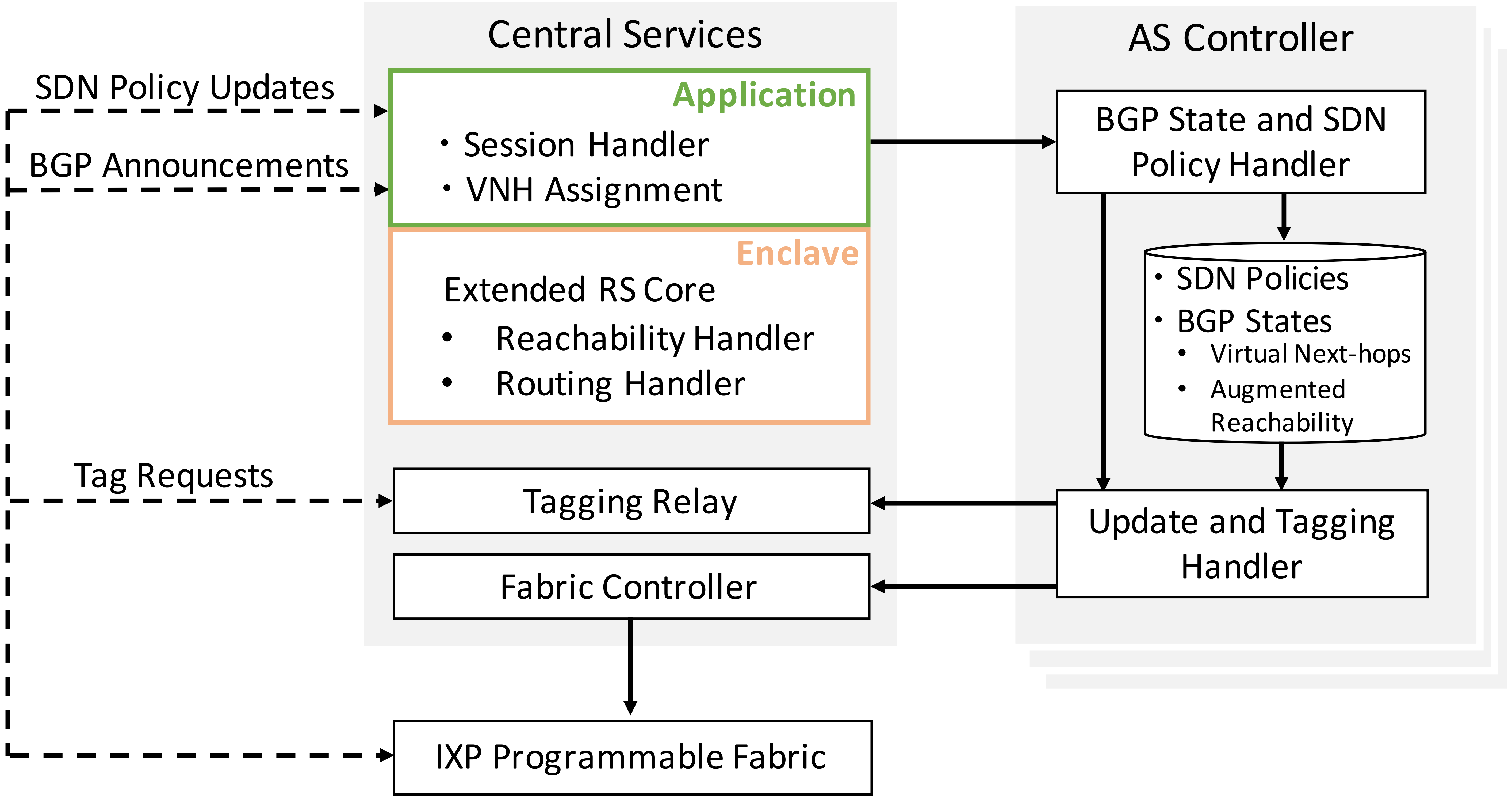
SGDX = SGX + SDX

New private function

- Augment SDN outbound policies with BGP reachability

Consolidate computation

- Run all routing related functions in central services



Implementation Analysis

- SGRS and SGDX trusted part
  - Most functions are written in identical way as general C program
  - SGX related logic
    - Reusable: *enclave_init() remote_attestation()* etc.
    - Transition call interfaces by enclave definition language
    - Application-specific transition call functions
  - Development overhead (*Application-specific LOC / total trusted LOC*)
    - SGRS: 207 / 2241 = 9.23%
    - SGDX: 277 / 2807 = 9.87%

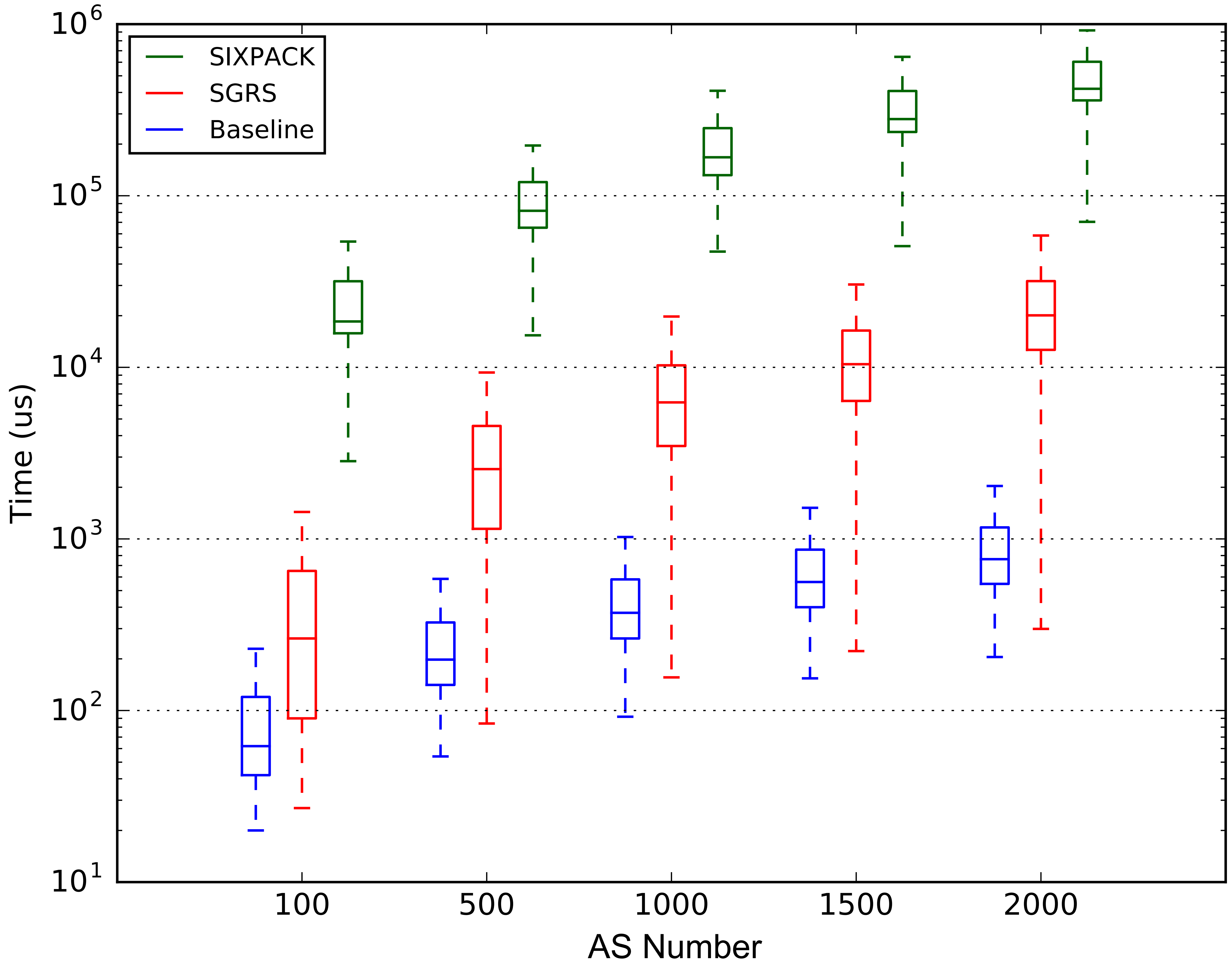
Evaluation

- A 4-core SGX-enabled processor and 64GB DRAM
- Data-sets derived from real-world RIPE RIS data
  - Original data consists of only public BGP updates and RIB dumps
  - Extend AS number with uniform fraction of peering
  - Random local preferences as ranking policies

- Replay real BGP update traces to evaluate BGP update compute time
- SGRS v.s. SIXPACK, SGDX v.s. iSDX

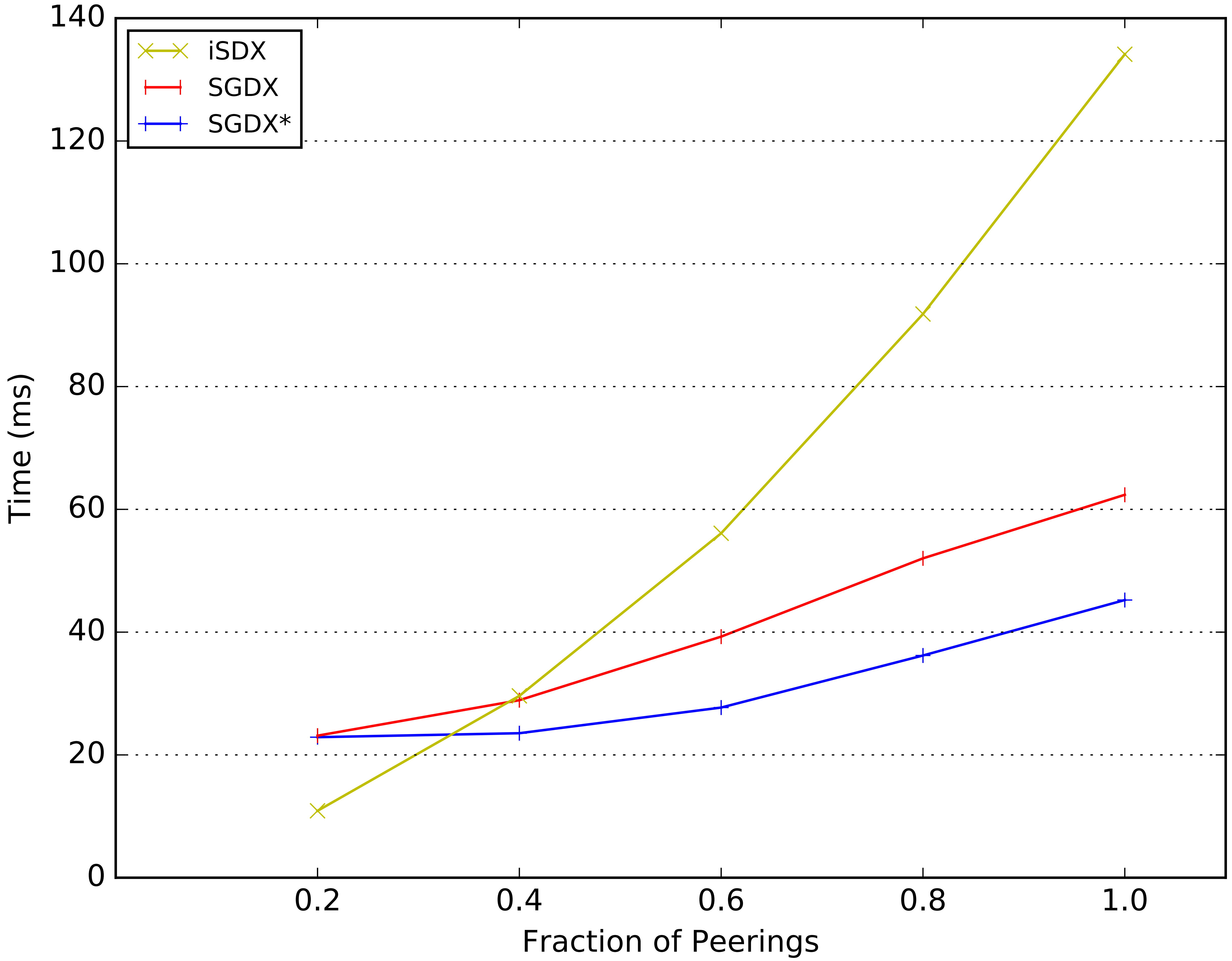
Evaluation

- SGRS is 20x-70x faster than SIX-PACK
- SGRS is 4x-26x slower than Baseline (insecure)



Evaluation

- SGDX is comparable to iSDX ranging from 0.5x-2.1x the processing time of iSDX



Summary

- Propose SGRS and SGDX to preserve privacy at IXPs with TEE
- SGDX is approximately scalable and flexible as iSDX while preserves privacy
- Codebase: https://github.com/huxh10/SGDX

- Future work
  - Expanding the threat model to mitigate side-channel attacks
  - Application extensions with SGDX
  - Automating the privacy-preserving development process