Inter-domain cooperative DDoS protection problems and mechanism

draft-nishizuka-dots-inter-domain-mechanism-00

Kaname Nishizuka	NTT
Liang Xia	Huawei
Jinwei Xia	Huawei
DaCheng Zhang	Alibaba
Luyuan Fang	Microsoft

April 2016 Buenos Ayres

New Features of DDoS Attacks

Traffic of a single attack is up to 500 Gbps.

Massive numbers of Internet of Things (IoT) terminals may become zombie hosts.

On-premise traffic mitigation solutions cannot eliminate attacks.

Attacks are launched globally with hybrid attacks

Over 90% of the carriers are seeking cloud-based network traffic mitigation solutions.

Merits of Inter-domain Cooperative DDoS Protection

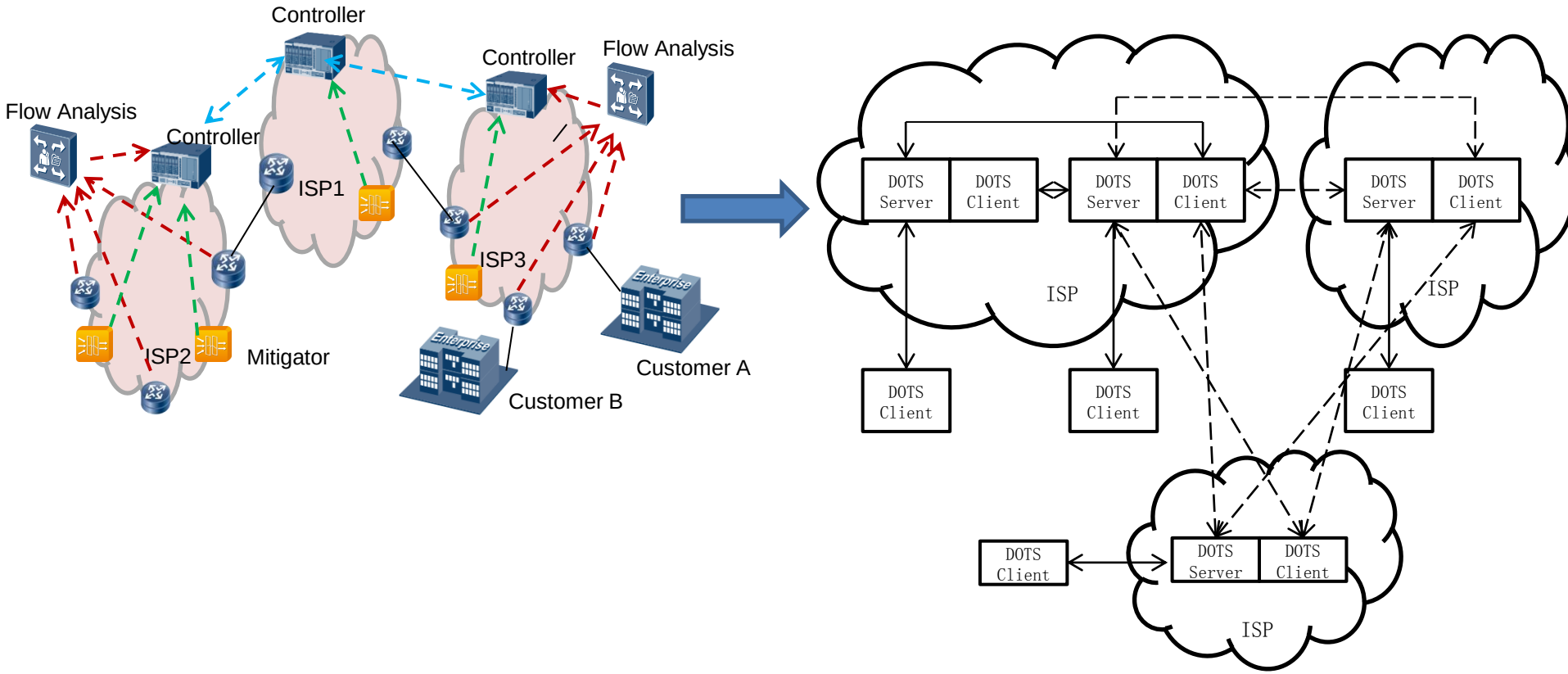
- *Resource and capability sharing among operators:*
  - Resource: CPU, memory, bandwidth, etc;
  - Capability: intelligence, filtering, blackholing, DPI, etc;
- *Comprehensive DDoS protection optimization:*
  - Near source mitigation to reduce the useless attack traffic in network;
  - The only way to solve the inter-domain uplink congestion problem;
  - Relieve the burden for individual operator.

A golbal on-demand DDoS mitigation service with unlimited bandwidth, territory, resource, ...

Challenges for Inter-domain Cooperative DDoS Protection

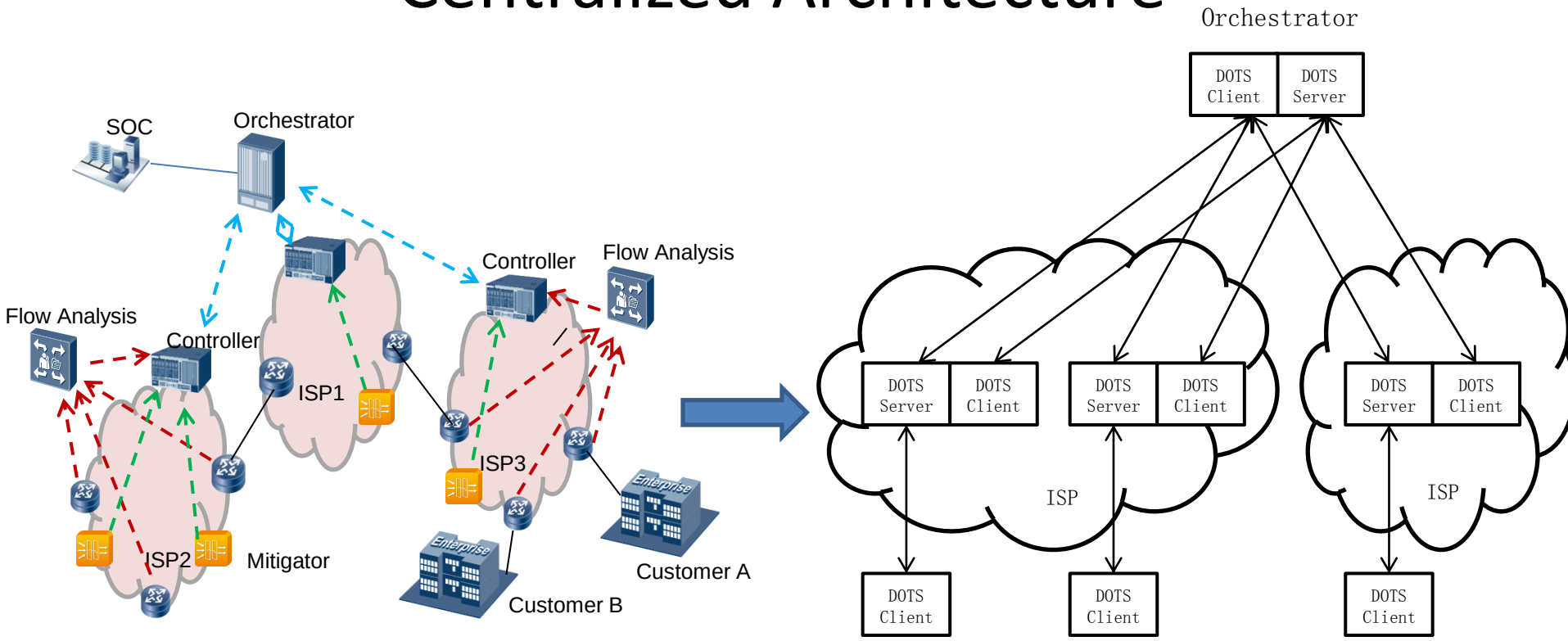
1. Bootstrapping Problems (automatic provisioning):
   - Trust relation and secure channel set up;
   - Auto-discovery and capability negotiation, etc.
2. Coordination problems:
   - How to get the appropriate mitigation service from other operators with high efficiency: make the decision based on information sharing;
   - Near source mitigation: spoofed address, privacy protection;
   - Others: accounting, returning path, etc.

Distributed Architecture



- *Peer-to-peer coordination;*
- *customer<->DOTS client, ISP controller<->DOTS server + DOTS client;*
- *The inter-domain coordination can be a repeated process;*
- *A straightforward and simple solution for the DDoS protection cooperation among small number of ISPs:*
  - ✓ *The incomplete information may not lead to the most optimized operation;*
  - ✓ *Configurations become more complex and error prone as the number of ISPs increases;*
  - ✓ *By repeated coordination among multiple ISPs, It may take a long time to enforce the mitigation.*

Centralized Architecture



- *the centralized orchestrator is the core component to the inter-domain system;*
- *customer<->DOTS client, ISP controller<->DOTS server + DOTS client, orchestrator<->DOTS server + DOTS client;*
- *The inter-domain coordination is bridged by the orchestrator;*
- *Comparing to distributed architecture:*
  - ✓ *The orchestrator has the HA problem;*
  - ✓ *Centralized way facilitates the automatic provisioning of DDoS protection resource and comprehensive information for overall optimized mitigation;*
  - ✓ *Direct communication with orchestrator guarantees quick and fixed DDoS response time.*

Inter-domain DDoS Protocol

- Secure channel (signaling, data):
  - Requirements: confidentiality, integrity and replay attack protection;
  - Mutual authentication: bidirectional certificate authentication ([ITU-T X.509]), bidirectional digital signature authentication;
  - Solution in this draft: https + JSON;

- Specification for protocol and messages (no difference for all architectures):
  - Provisioning stage
  - Signaling stage
  - heartbeat message

Provisioning Stage Protocol

- Registration process: facilitate the auto-discovery and capacity negotiation between the DOTS client and server;
  - Messages: registration, registration response, registration cancelling, registration cancelling response;
  - Operations: The DOTS client (in customer side, or in ISP controller, or in orchestrator) registers (or cancels registration) to the DOTS server in orchestrator (centralized architecture) or other ISP controllers (distributed architecture);

```
METHOD:POST - URL:{scheme}://{ho
registration body:
{
"customer_name": string;
"ip_version": string;
"protected_zone": string;
"protected_port": string;
"protected_protocol": string;
"countermeasures": string;
"tunnel_information": string;
"next_hop": string;
"white_list": string;
"black_list": string;

}
registration response body:
{
"customer_name": string;
"customer_id": string;
"access_token": string;
"thresholds_bps": number;
"thresholds_pps": number;
"duration": number;
"capable_attack_type": string;
"registration_time": string;
"mitigation_status": string;
}
```

```
METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/
          registration_cancelling
 registration cancelling body:
 {
 "customer_id": string;
 "reasons": string;
 }
 registration cancelling response body:
 {
 "customer_id": string;
 "result": string;
 }
```

Signaling Stage Protocol

- During DDoS attack: mitigation service request and status exchange;
  - Messages:
    - DOTS client to server: the messages of mitigation request, mitigation scope, mitigation efficacy notification, mitigation termination, heartbeat, ...
    - DOTS server to client: mitigation status notification, mitigation termination notification, heartbeat, ...
    - The corresponding response messages
  - Operations: DDoS mitigation request → mitigation scope update, mitigation efficacy/status notification, heartbeat → mitigation termination notification → mitigation termination

Signaling Stage Protocol

```
METHOD:POST - URL:{scheme}://{h
        mitigation_request
mitigation request body:
{
"access_token": string;
"traffic_protocol": string;
"source_port": string;
"destination_port": string;
"source_ip": string;
"destination_ip": string;
"time": string;
"dstip_current_bps": string;
"dstip_current_pps": string;
"dstip_peak_bps": string;
"dstip_peak_pps": string;
"dstip_average_bps": string;
"dstip_average_pps": string;
"bandwidth_threshold": string;
"type": string;
"severity": string;
"mitigation_action": string;
}
mitigation response body:
{
"access_token": string;
"mitigation_id": number;
"policy_id": number;
"description": string;
"start_time": string;
"current_bps": string;
"current_pps": string;
}
```

```
mitigation status request response body:
{
"mitigation_id": number;
"mitigation_status": number;
"source_port": string;
"destination_port": string;
"source_ip": string;
"destination_ip": string;
"TCP_flag": string;
"start_time": string;
"end_time": string;
"error_num": number;
"routing_state": string;
"forwarded_total_packets": number;
"forwarded_total_bits": number;
"forwarded_peak_pps": number;
"forwarded_peak_bps": number;
"forwarded_average_pps": number;
"forwarded_average_bps": number;
"malicious_total_packets": number;
"malicious_total_bits": number;
"malicious_peak_pps": number;
"malicious_peak_bps": number;
"malicious_average_pps": number;
"malicious_average_bps": number;
"record_time": string;
}
```

Implementation Related

- We have some running codes behind it, most for the centralized way;

- The solution focuses on the protocol for inter-operator use cases;

Next Steps

- Solicit Comments and feedbacks

- Keep on improvement, including:
  - be aligned to DOTS use cases, requirements, architecture drafts;
  - More descriptions about secure channel, transport protocol, DOTS relay features;
  - More details about DOTS messages.

Thanks!

Liang Xia (Frank)