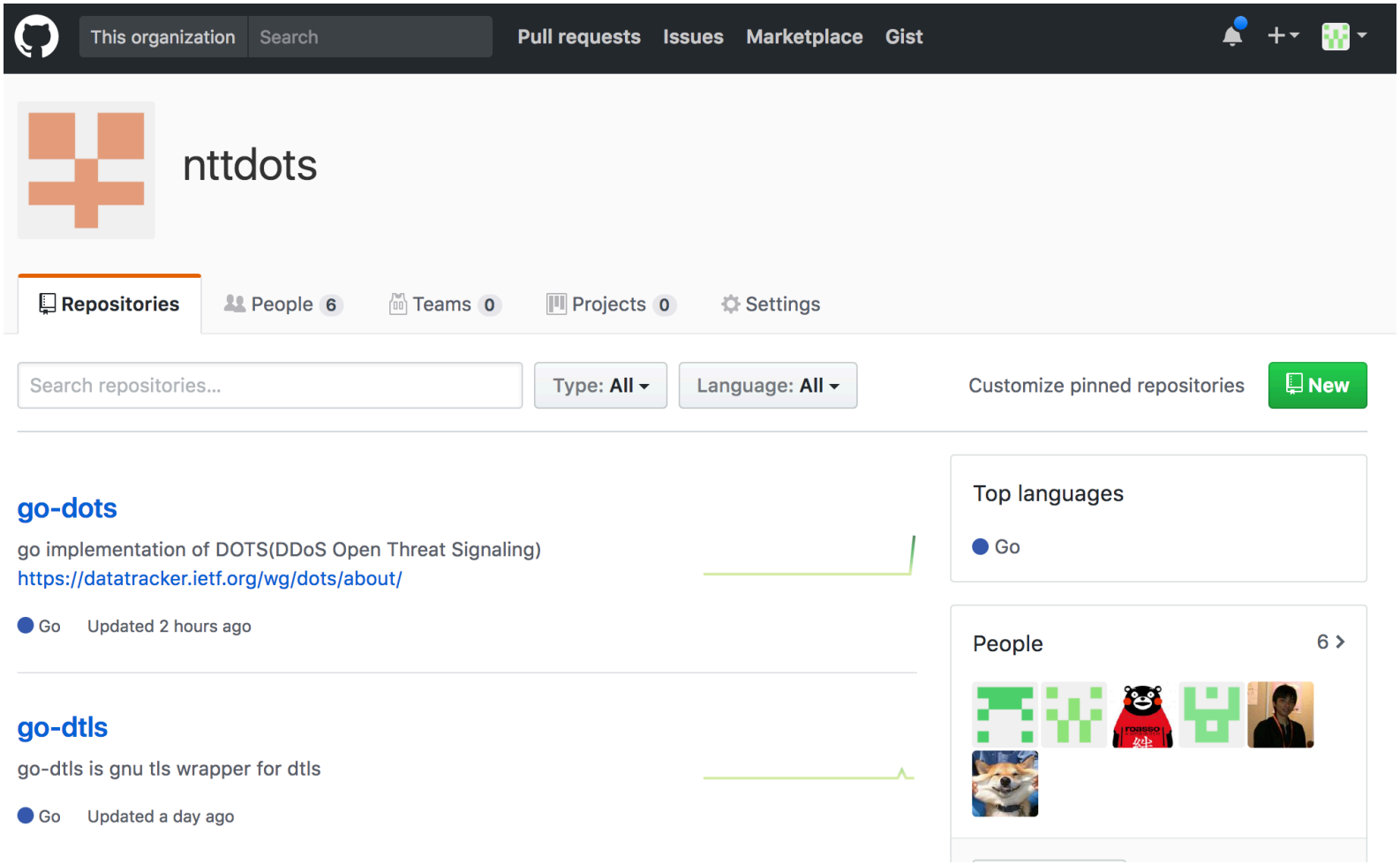
Go implementation of DOTS

DOTS WG
2017.07.20

Kaname Nishizuka
(NTTCommunications)

We opened the code !!

- https://github.com/nttdots



What was developed in hackathon

- made the code easy to be deployed in various environments
  - made docker-compose files for each services
  - refined configuration part

- clarified the documents
  - for newcomers to this field

Demos and Interests

- made a demonstration of one user scenario
  - on a portable docker environment
  - triggering “blackhole routing” from victim side

- attracted 4 people and showed them demo

We do demo on Bits-n-Bites

- Today: 19:15-21:15
- Prosím, visit us on the site.

Demo: Go implementation of DOTS

Demo scenario:
Enabling DDoS Protection in an upstream network by DOTS protocol

https://github.com/nttdots/go-dots

DOTS is:

- **D**DoS **O**pen **T**hreat **S**ignaling
- Automation and Standardization of signaling for DDoS protection
- "ask for help!" from a victim to an upstream provider
  - inter-organization / including authN and authX in spec

What you can see in this demo:

- A DOTS client sends a mitigation request to a DOTS server over DOTS signal channel.
- The DOTS server receives and validates the request, then starts mitigation by kicking a blocker
- In this demo, the blocker is a gobgp server which triggers "blackhole routing" in a service operator's network

Signal Channel

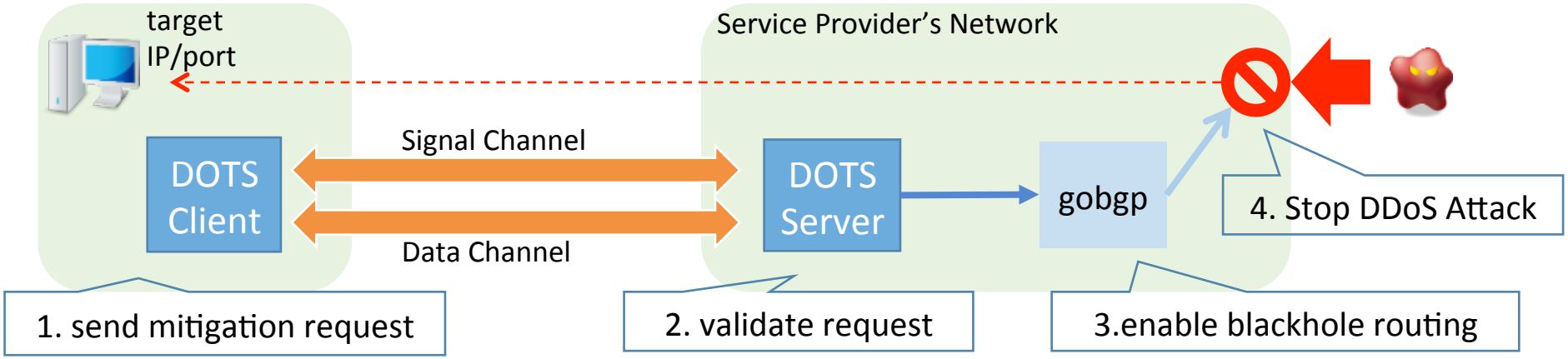
```
+--------------+
|    DOTS      |
+--------------+
|    CoAP      |
+--------------+
| TLS |  DTLS  |
+--------------+
| TCP |   UDP  |
+--------------+
|    IP        |
+--------------+
```

Data Channel

```
+--------------+
|    DOTS      |
+--------------+
|  RESTCONF    |
+--------------+
|    TLS       |
+--------------+
|    TCP       |
+--------------+
|    IP        |
+--------------+
```

Mitigation Request Model

```
module: ietf-dots-signal
    +--rw mitigation-scope
       +--rw scope* [mitigation-id]
          +--rw mitigation-id        int32
          +--rw target-ip*           inet:ip-address
          +--rw target-prefix*       inet:ip-prefix
          +--rw target-port-range* [lower-port upper-port]
          |  +--rw lower-port    inet:port-number
          |  +--rw upper-port    inet:port-number
          +--rw target-protocol*     uint8
          +--rw fqdn*                inet:domain-name
          +--rw uri*                 inet:uri
          +--rw alias*               string
          +--rw lifetime?            int32
```



Lessons Learned(1/3)

1. Need more description on specification of mutual authentication
   - (D)TLS based-on client certificate
     - tend to use self-signed certification (in lab)
     - how can we bind the (D)TLS channel and customer (mitigation scope)
     - CN(or SNI) should be used? (it's not clearly documented)
   - what else for mutual authentication

Lessons Learned(2/3)

2. Still searching for good RESTCONF library
   - As an alternative, CoAP/DTLS can be used for data channel
   - but we want to implement it on RESTCONF, if we can.

Lessons Learned(3/3)

3. Zero heartbeat mode should be allowed
   - As a starting point of implementation in lab
   - Also there are several usecases (as discussed in the last IETF meeting)
   - “MUST” in REQ.SIG-003 should be relaxed?

IANA considerations

- need assignment for default port number
  - 4646/udp for signal channel (from draft-mortensen-dots-over-udp)
  - 4647 for data channel?

implementation specific problems

- Traffic data collection
  - traffic information should be returned from DOTS servers
    - incoming traffic / blocked traffic / passed traffic
  - need additional software component to collect those data from network equipment or mitigation boxes
    - very implementation specific but required
- Partially valid request
  - When a mitigation request includes valid scope and invalid scope at the same time, what is the appropriate behavior?
    - reject all? / pass valid request only?

Next Step

- As an OSS,
  - adopt to the various deployment scenario
  - keep going on the implementation of WG drafts and make feedback to the spec

- your feedback is welcome☺

DOTS is getting popular!

- We'd like to do interoperability testing at the next hackathon in IETF100
  - signal channel interop will be the 1st step