P4 VPN Authentication

Authentication of VPN Traffic on a Network Device with P4

Jeroen Klomp

University of Amsterdam
System and Network Engineering
Research Project

June 29, 2016
CoCo Introduction

- Community Connection (CoCo)
  - User-initiated multi-domain VPN service
  - Support eScience
  - Prototype phase; no proper authentication
CoCo Overview

1: create VPN

Customer 1
VPN User

Provider 1
CoCo VPN Portal + Agent
Provider Edge

Provider 2
CoCo VPN Portal + Agent
Provider Edge

Customer 2
VPN User

3: install VPN paths (OpenFlow)

4: use VPN
CoCo Overview

1: create VPN

Customer 1

Provider 1

CoCo VPN Portal + Agent

2: exchange VPN paths (BGP)

Provider Edge

Customer 2

Provider 2

CoCo VPN Portal + Agent

3: install VPN paths (OpenFlow)

Customer 2

4: use VPN

VPN User
# CoCo Overview

<table>
<thead>
<tr>
<th>Customer 1</th>
<th>Provider 1</th>
<th>Provider 2</th>
<th>Customer 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1: create VPN</strong></td>
<td>CoCo VPN Portal + Agent</td>
<td>CoCo VPN Portal + Agent</td>
<td><strong>VPN User</strong></td>
</tr>
<tr>
<td><strong>VPN User</strong></td>
<td><strong>Provider Edge</strong></td>
<td><strong>Provider Edge</strong></td>
<td><strong>VPN User</strong></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td><strong>4: use VPN</strong></td>
</tr>
<tr>
<td><strong>2: exchange VPN paths (BGP)</strong></td>
<td><strong>3: install VPN paths (OpenFlow)</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Introduction

CoCo Authentication

## CoCo Overview

1. **1: create VPN**

2. **2: exchange VPN paths (BGP)**

3. **3: install VPN paths (OpenFlow)**

4. **4: use VPN**
CoCo Overview

Customer 1 | Provider 1 | Provider 2 | Customer 2
---|---|---|---
1: create VPN | CoCo VPN Portal + Agent | CoCo VPN Portal + Agent | VPN User
2: exchange VPN paths (BGP) | Provider Edge | Provider Edge | 2
3: install VPN paths (OpenFlow) | 3
4: use VPN | 4

VPN User

P4 VPN Authentication Jeroen Klomp
P4 Overview

P4 & OpenFlow

1. Apps
2. Northbound API
3. OpenFlow Controller
   - OpenFlow Protocol
4. OpenFlow Agent
5. Driver
6. Programmable Data Plane ASIC

Compile

Auto-Generated API

Target Binary

Copyright © 2016 P4 Language Consortium.
Authentication Use Case

<table>
<thead>
<tr>
<th>Customer 1</th>
<th>Provider 1</th>
<th>Customer 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPN User</td>
<td>CoCo VPN Portal + Agent</td>
<td>Provider 2</td>
</tr>
<tr>
<td>Provider Edge</td>
<td></td>
<td>VPN User</td>
</tr>
</tbody>
</table>

1: establish session
2: session (key) established
3: session key + metadata
4: network traffic + authentication
5: verify and remove authentication
6: network traffic
## Authentication Use Case

<table>
<thead>
<tr>
<th>Customer 1</th>
<th>Provider 1</th>
<th>Customer 2 Provider 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VPN User</strong></td>
<td><strong>CoCo VPN Portal + Agent</strong></td>
<td><strong>VPN User</strong></td>
</tr>
<tr>
<td>1: establish session</td>
<td>2: session (key) established</td>
<td></td>
</tr>
</tbody>
</table>

1. **Customer 1** establishes a session with **Provider 1**.
2. **Provider 1** sends a session key (metadata) to **Customer 1**.
3. **Customer 1** verifies and removes the authentication
4. **Network traffic** and authentication are transmitted to **Customer 2**.
5. **Provider 2** verifies the authentication and establishes a session with **Customer 2**.
6. **Network traffic** is transmitted to **Provider 2**.
Authentication Use Case

1: establish session
2: session (key) established
3: session key + metadata
4: network traffic + authentication
5: verify and remove authentication
6: network traffic

CoCo VPN Portal + Agent

Customer 1
VPN User

Provider 1

Customer 2
VPN User

Provider 2
Authentication Use Case

1: establish session
2: session (key) established
3: session key + metadata
4: network traffic + authentication
5: verify and remove authentication
6: network traffic
Authentication Use Case

1: establish session
2: session (key) established
3: session key + metadata
4: network traffic + authentication
5: verify and remove authentication
6: network traffic

Customer 1
VPN User

Provider 1
CoCo VPN Portal + Agent
Provider Edge

Customer 2
VPN User

Provider 2
Authentication Use Case

1: establish session
2: session (key) established
3: session key + metadata
4: network traffic + authentication
5: verify and remove authentication
6: network traffic

Customer 1

VPN User

Provider 1

CoCo VPN Portal + Agent

Provider Edge

Customer 2

VPN User

Provider 2
Authentication Scheme

- Requirements
  - Secure
  - Support research
Authentication Scheme

- Requirements
  - Secure
  - Support research

- Authentication method
  - Message authentication code (MAC)
Authentication Scheme

- **Requirements**
  - Secure
  - Support research

- **Authentication method**
  - Message authentication code (MAC)
  - Like checksums, MAC algorithms cannot be implemented in P4 itself
    - Use external function provided by the target
    - Possibly a standard library with MAC algorithms?
    - Language needs facility for key input
Authentication Scheme

- Requirements
  - Secure
  - Support research

- Authentication method
  - Message authentication code (MAC)
    - Like checksums, MAC algorithms cannot be implemented in P4 itself
      - Use external function provided by the target
      - Possibly a standard library with MAC algorithms?
      - Language needs facility for key input

- Authentication protocol
Authentication Protocol

- **IPSEC Authentication Header?**
  - Contains all necessary fields
    - Security Parameters Index (SPI): Security Association → session ID
    - Sequence number → replay protection
    - Integrity Check Value (ICV) → variable length MAC
Implementation in P4

- Distinguish sessions
  - Session identifier table containing session IDs and session keys
Implementation in P4

- Distinguish sessions
  - Session identifier table containing session IDs and session keys
- Sequence number
  - Register per session
Implementation in P4

- **Distinguish sessions**
  - Session identifier table containing session IDs and session keys
- **Sequence number**
  - Register per session
- **MAC**
  - Session key mixed with message
  - Hash\(^1\) calculated and stored as metadata via primitive action:
    \[
    \text{modify\_field\_with\_hash\_based\_offset()}
    \]

\(^1\)‘simulated’ via checksum
Simplified Authentication Protocol

- GRE (Generic Routing Encapsulation)
  - Has necessary fields
    - Key: session ID
    - Sequence number
    - Checksum: (mis)used for MAC simulation
  - Easily craft packets e.g., via Scapy

| Offset | Type | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|--------|------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
|        |      | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| 0      | 0    |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 32     | 4    |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 64     | 8    |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 96     | 12   |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 128    | 16   |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |

- Flags: CKS
- Reserved0
- Version
- Protocol Type: 0000 (possibly GRE keepalive)
- Checksum: MAC (CRC16)
- Offset: key (not on wire)
- Key: session identifier
- Sequence Number
- ICMP echo request with random payload

P4 VPN Authentication

Jeroen Klomp
Test Setup

Mininet

P4

Mininet

H1

S1

H2

10.0.0.10

10.0.1.10

Scapy

GRE(key)/ICMP(rqst)/'123'

ICMP(rqst)/'123'

ICMP(rply)/'123'

ICMP(rply)/'123'

P4 VPN Authentication Jeroen Klomp
Demonstration


Results

- Concepts work\(^2\)
  - Packets accepted only with correct key
  - Sequence number correctly checked & updated
  - Multiple session IDs and keys supported simultaneously

\(^2\)although using an extremely weak form of MAC
Results

- Concepts work\(^2\)
  - Packets accepted only with correct key
  - Sequence number correctly checked & updated
  - Multiple session IDs and keys supported simultaneously

- P4 language and software targets still work in progress
  - Problems with dropping traffic
  - Register operations not yet in specification
  - Key length supported?

\(^2\)although using an extremely weak form of MAC
Conclusion

- Authentication with P4 is feasible
  - But requires new P4 features and target support
  - Keep authentication scheme & P4 program simple
Conclusion

- Authentication with P4 is feasible
  - But requires new P4 features and target support
  - Keep authentication scheme & P4 program simple
- Lots of caveats
  - Target limitations; cryptographic algorithms; NAT; IP fragmentation; packet forwarding mode (cut-through); sequence number synchronisation; asymmetric flows
Conclusion

- Authentication with P4 is feasible
  - But requires new P4 features and target support
  - Keep authentication scheme & P4 program simple
- Lots of caveats
  - Target limitations; cryptographic algorithms; NAT; IP fragmentation; packet forwarding mode (cut-through); sequence number synchronisation; asymmetric flows
- Where to go from here?
  - Add cryptographic means to P4
  - Further design CoCo architecture & authentication scheme
    - Implement in P4, controller & client
    - End-to-end authentication & encryption?
Questions?