THE TCP AUTHENTICATION OPTION (TCP-AO)

Melchior Aelmans
Juniper Networks
Why do we need TCP security?
MOTIVATION

• What are we protecting?
  – Long-lived TCP sessions
  – Examples
    • Routing protocols (BGP, LDP)
    • Long-lived TCP sessions between other applications

• What are we protecting against?
  – Blind insertion attacks
  – Replay attacks
BLIND INSERTION ATTACK ON A BGP SESSION

- Router A maintains a BGP session with Router B
  - They exchange many routes over many hours

- Node C sends a few packets per second to Router B for many hours
  - IP source address: Router A (spoofed)
  - Payload: TCP
    - RST bit set
    - Destination ports: BGP (179)
    - Random sequence numbers

- B discards most packets, because their sequence numbers are invalid

- Sooner or later, C sends a packet with a valid sequence number

- BGP session resets
TCP MD5
LEGACY SOLUTION: TCP-MD5 [RFC 2385]

- Sending and receiving nodes are configured with a pre-shared key

- Sending node procedures
  - Calculate a Message Authentication Code (MAC) for each TCP segment
    - Use MD5 to calculate MAC
    - Calculate MAC over the TCP segment and the pre-shared key
  - Include an MD5 Signature Option in each segment
    - MD5 Signature Option includes MAC

- Receiving node procedures
  - Calculate a MAC for each received TCP segment
  - Discard the packet if the calculated MAC does not match the received MAC
TCP-MD5 IS DEPRECATED

• New requirements
  – Change pre-shared keys without resetting TCP session
  – Support multiple authentication algorithms

• Pre-shared key change
  – It is difficult to change TCP-MD5 pre-shared keys without resetting the TCP session
  – It is difficult to reset TCP sessions that support BGP
  – Therefore, TCP-MD5 pre-shared keys were rarely changed

• Authentication algorithm agility
  – MD5 has been replaced by stronger authentication algorithms
  – Even stronger authentication algorithms are expected in the future
• Monday, June 5 2006 - NANOG 37

  Ron Bonica - Authentication for TCP-based Routing and Management Protocols

• June 2010

  RFC5925 published

• Tuesday, June 26 2018 - NANOG 73

  Ignas Bagdonas - Lightning Talk: BGP Transport Security - Do You Care?

• Monday, October 19 2020 - NANOG 80

  Melchior Aelmans - It is time...to replace MD5
TCP Authentication Option
TCP-AO [RFC 5925] REPLACES TCP-MD5

• Supports
  – Pre-shared key change without resetting TCP session
  – Multiple authentication algorithms
TCP-AO CONCEPTS

• Master Key Tuple (MKT)
  – One or more MKTs are configured on each node
  – Used to derive traffic keys

• Traffic key
  – Used to generate a MAC for each TCP segment

• TCP-Authentication Option
  – Used to authenticate TCP segments
  – Contains a MAC, KeyID and RNextKeyID
    • KeyID identifies MKT and traffic key that were used to generate MAC
    • RNextKey identifies MKT and traffic key that the receiving node should use when generating a MAC for the next segment it sends
MKT CONTENTS

• A TCP connection identifier
  – Source address, destination address, source port, destination port
  – Wildcards allowed

• A TCP Options flag (determine which TCP options are covered by MAC)

• Identifiers
  – Sending: Used to generate KeyID on outbound segments
  – Receiving: Used to resolve KeyID on inbound segments

• An authentication algorithm

• Master key (i.e., keying material)

• A key derivation algorithm
TRAFFIC KEYS

• Four traffic keys are derived from each MKT
  – SEND_SYN
  – RECEIVE_SYN
  – SEND-OTHER
  – RECEIVE-OTHER
## THE TCP AUTHENTICATION OPTION

<table>
<thead>
<tr>
<th>Kind=29</th>
<th>Length</th>
<th>KeyID</th>
<th>RNextKeyID</th>
</tr>
</thead>
</table>

Message Authentication Code (MAC)  
(Variable Length)

Flexible support for multiple algorithms today and extensible with future algorithms
PULLING IT ALL TOGETHER: KEYING

• Each node is each configured with one or more MKTs
• Each node derives four traffic keys from each MKT
• Each node independently determines which MKT is active
  – Method is beyond the scope of RFC 5925
  – Many implementations specify a start-time and an end-time for each MKT
PULLING IT ALL TOGETHER: AUTHENTICATION

• Sending node procedures
  – Calculate a Message Authentication Code (MAC) for each TCP segment
    • Use the appropriate authentication algorithm
    • Calculate MAC over the TCP segment and an active traffic key
  – Include a TCP-AO in each segment
    • MD5 Signature Option includes MAC, KeyID and RNExtKeyID

• Receiving node procedures
  – Calculate a MAC for each received TCP segment
    • Use algorithm and traffic key associated with the received KeyID
  – Discard the packet if the calculated MAC does not match the received MAC
IMPLEMENTATION STATUS AND FURTHER READING

Implementation status:
– Cisco: Stable since IOS XR 6.6.3 and 7.0.1
– Juniper Networks: 20.3R1
– Huawei: targeted for Q2 2021

Further information:
– Nokia & Juniper interoperability test: https://github.com/TCP-AO/Interoperability-testing
– Configuration examples: https://github.com/TCP-AO/Configuration-examples
RELATIONSHIP WITH GTSM [RFC 5082]

• GTSM protects eBGP sessions
  – Sender sets TTL to 255
  – Receiver rejects packets containing eBGP if TTL is less than 254

• TCP-AO still needed to protect eBGP sessions from attackers that are one hop away

• TCP-AO still needed to protect iBGP sessions from internal attack

➢ Security best practices implement many layers of protection, don’t rely on just one mechanism!
Interoperability testing
JUNIPER AND NOKIA INTEROP TEST RESULTS

• Successful interop test using TCP-AO for BGP finished in June 2020
• Established multihop IPv4 and IPv6 BGP sessions over the Internet
• No need to meet or bring routers for testing in person
• Tested with HMAC-SHA-1-96 and AES-128-CMAC-96 algorithms
LESSONS LEARNED #1 – SEND AND RECEIVE ARE CONFIGURED FROM THE ROUTER’S PERSPECTIVE

Juniper

```bash
# show security authentication-key-chains
key-chain ao_aes_chain {
    key 0 {
        secret "$9$xk3NVYq.53/taZnCu1yrwYg4UHf5F/AOz3"; #
       SECRET-DATA
        start-time "2020-6-16.01:00:00 +0530";
        algorithm ao;
        ao-attribute {
            send-id 9;
            recv-id 2;
            tcp-ao-option enabled;
            cryptographic-algorithm aes-128-cmac-96;
        }
    }
}
```

Nokia

```bash
configure system security {
    keychain "interoptest-aes" {
        tcp-option-number {
            receive tcp-ao
            send tcp-ao
        }
        receive {
            entry 9 {
                authentication-key
                "yzClLKIFsAVR91AobUXUT/ppPzL7bVxBrNNG" hash
                algorithm aes-128-cmac-96
                begin-time 2020-06-09T04:00:00.0Z
            }
        }
        send {
            entry 2 {
                authentication-key
                "yzClLKIFsAVR91AobUXUT/ppPzL7bVxBrNNG" hash
                algorithm aes-128-cmac-96
                begin-time 2020-06-09T04:00:00.0Z
            }
        }
    }
}
```

- Send and receive IDs must match each other
- TCP-AO supports multiple algorithms, make sure you are using are the same one
LESSONS LEARNED #2 – FIREWALLS MAY CHANGE TCP HEADERS

- The TCP MSS option was modified by a firewall in the path between the routers
- This caused the MAC calculation to fail on the receiver and the BGP session would not come up
- The TCP-AO option worked as expected to protect against modified packets!
CALL TO ACTION

Operators:
• Think about how TCP-AO fits into your overall routing security strategy
• Router vendor implementations are available now, start looking at them
• Ask for TCP-AO in RFPs/RFIs if it’s missing

Developers:
• There is no ecosystem of open source implementations and tools yet
• Need kernel implementations: Linux and *BSD
• Need support in tools: tcpdump, Wireshark, etc.
• Need features in routing implementations: BIRD, FRR, goBGP, OpenBGPD, etc.
• Juniper and Nokia can provide implementations for testing!
THANK YOU