Revised Cookie Processing in IKEv2

draft-smyslov-ipsecme-ikev2-cookie-revised

Valery Smyslov
svan@elvis.ru

IETF 109
Using Cookies in IKEv2

The most recent IKE_SA_INIT request is included in the AUTH payload calculation in the IKE_AUTH exchange. In this example it is req2 for both the initiator and the responder.
The most recent IKE_SA_INIT request sent by the initiator is req2, while the responder only received req1, so authentication is failed.
Problem Scenario 2

The most recent IKE_SA_INIT request sent by the initiator is req3, while the responder only received req2, so authentication is failed.
Source of the Problem

- The IKE_SA_INIT request can be sent several times with different content depending on the responder state.
- If there is high probability of packets loss and reordering, then peers may complete the IKE_SA_INIT exchange having different views on what was the most recently sent IKE_SA_INIT request.
- This request message is used in calculation of the AUTH payload, so if peers use different messages authentication would erroneously fail.
Severity of the Problem

• There are some preconditions for this problem to become noticeable
  – network with high probability of packet loss and delay
  – relatively frequent change of responder state (either changing cookie generation secret or changing responder’s mind whether it is under attack)
• It might be rare in normal conditions, but in stress tests we observed that up to 5% of SAs failed due to this problem
  – for customers it looks strange that authentication sometimes failed with proper credentials
• This is a protocol flaw
Proposed Solution Overview

• Revise cookie processing by excluding Notify payload containing cookie (if present) from the IKE_SA_INIT request message when calculating the AUTH payload content
  – the cookie is already verified by the responder, no need to include it into the data to be authenticated
• For backward compatibility make the revised processing negotiable
Responder includes a new notification REVISEDCOOKIE in the message containing COOKIE notification. If initiator also supports this extension, it returns cookie in this notification instead of COOKIE notification.
Revised Cookie Processing

- If peers agreed upon using this extension then the cookie processing is changed
  - no changes in cookie anti-clogging function – responder still sends stateless cookie and when it is returned back by initiator it MUST be verified before message is processed
  
  According to RFC7296 initiator’s AUTH payload is calculated by signing (or MAC’ing) the blob:

  \[ \text{InitiatorSignedOctets} = \text{RealMessage1} | \text{NonceRData} | \text{MACedIDForI} \]

  - if REVISED_COOKIE Notify payload is present in RealMessage1 (i.e. in IKE_SA_INIT request message), then for the purpose of AUTH payload calculation the message is modified as if it contained no this payload
Adjusting IKE_SA_INIT Request for AUTH Payload Calculation

<table>
<thead>
<tr>
<th>IKE SA Initiator's SPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKE SA Responder's SPI</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NextPld1</th>
<th>Version</th>
<th>Exchange</th>
<th>Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Message ID

MsgLen

<table>
<thead>
<tr>
<th>NextPld2</th>
<th>RESERVED</th>
<th>PldLen1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>REVISED_COOKIE</td>
</tr>
</tbody>
</table>

Cookie

Rest of Message

IKE SA Initiator's SPI

IKE SA Responder's SPI

NextPld2 | Version | Exchange | Flags |
---------|---------|----------|-------|
         |         |          |       |

Message ID

MsgLen’ = MsgLen - PldLen1

Rest of Message

REVISED_COOKIE Notify Payload
Thanks

• Comments? Questions?
• Is this problem worth to address?
• Is the suggested approach reasonable?
• WG adoption?