FragAttacks: summary of findings

Mathy Vanhoef
Draft version 1, 8 March 2021
Table of contents

Design flaws:

1. Design flaw: aggregation attack (CVE-2020-24588)
2. Design flaw: mixed key attack (CVE-2020-24587)
3. Design flaw: fragment cache attack (CVE-2020-24586)
Table of contents

Implementation flaws allowing trivial plaintext injection:

4. Accepting plaintext frames (CVE-2020-26140 / 26143)
6. Plaintext broadcast fragments (CVE-2020-26145)
7. Cloacked aggregated frames (CVE-2020-26144)

Other implementation flaws:

8. Pre-auth EAPOL forwarding (CVE-2020-26139)
9. Non-consecutive packet numbers (CVE-2020-26146)
10. Mixed plain/encrypted fragments (CVE-2020-26147)
11. No fragmentation support (CVE-2020-26142)
Aggregation Attack

CVE-2020-24588
Root cause

› The “is aggregated” flag in the Wi-Fi header is not protected:

An adversary can flip the “is aggregated” flag

Payload will be parsed differently → allows packet injection
# Impact

<table>
<thead>
<tr>
<th>Target</th>
<th>Preconditions</th>
<th>Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>Client connects to attacker’s server</td>
<td>Inject packets to client</td>
</tr>
<tr>
<td>AP</td>
<td>AP is vulnerable to <a href="https://example.com">CVE-2020-26139</a></td>
<td>Inject packets to client</td>
</tr>
<tr>
<td>AP</td>
<td>Client connects to attacker’s server and this client uses predictable IP IDs</td>
<td>Inject packets to AP</td>
</tr>
</tbody>
</table>

Example attack: make client use a **malicious DNS server** or **bypass the AP’s NAT** to directly access local devices
Mixed Key Attack
CVE-2020-24587
Root cause

- Fragments encrypted under different keys are reassembled:

\[ Enc_k(Frag_0(s)) \quad Enc_k(Frag_0(s')) \quad Enc_m(Frag_1(s')) \quad Enc_m(Frag_1(s)) \]

- Refresh session key from \( k \) to \( m \)

- Receiver will decrypt & reassemble fragments \( Frag_0 \) and \( Frag_1 \)

- Can be abused to forge frames by mixing fragments
<table>
<thead>
<tr>
<th>Target</th>
<th>Preconditions</th>
<th>Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP</td>
<td>Client connects to attacker’s server and client sens fragmented frames and the network refreshes session keys (= unlikely in practice)</td>
<td>Exfiltrate data sent by client</td>
</tr>
<tr>
<td>Client</td>
<td>Only a theoretic concern (see paper)</td>
<td>Theoretic (see paper)</td>
</tr>
</tbody>
</table>

Example attack: **exfiltrate** a web browser **cookie** of the client when plaintext HTTP is used.
Fragment Cache Attack
CVE-2020-24586
Root cause

- The fragment cache isn’t cleared when (re)connecting:

  - Disconnect & let client connect

  - Attacker’s fragment $Frag_0$ & the client’s $Frag_1$ is reassembled

  - Can be abused to exfiltrate & forge frames by mixing fragments
## Impact

<table>
<thead>
<tr>
<th>Target</th>
<th>Preconditions</th>
<th>Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP</td>
<td>Target is an enterprise network <em>and</em> client sends fragmented frames (fairly unlikely)</td>
<td>Exfiltrate data sent by client <em>and</em> inject packets to AP</td>
</tr>
<tr>
<td>Client</td>
<td>Client will connect to the adversary’s network <em>(but won’t trust it)</em> <em>and</em> the AP sends fragmented frames (seems unlikely)</td>
<td>Inject packets to client</td>
</tr>
</tbody>
</table>

Example attacks: exfiltrate a plaintext browser cookie, make client use a malicious DNS server, bypass the AP’s NAT
Implementation Flaws: trivial plaintext injection
Accepted plaintext frames (CVE-2020-26140 / 26143)

Accepting **plaintext frames** (CVE-2020-26140)

› Examples: some routers, some dongles on Linux/Windows

Accepting **fragmented plaintext frames** (CVE-2020-26143)

› Examples: many dongles on Windows, some FreeBSD APs

→ Can **inject frames** independent of network config
Plaintext broadcast fragments (CVE-2020-26145)

Some devices accept plaintext broadcast fragments

› Sometimes only accepted while connecting
› Treated as full frames!
› Examples: MacOS, iOS, and Free/NetBSD APs

→ Can inject frames independent of network config
Cloaked aggregated frames (CVE-2020-26144)

Some accept **aggregate frames that resemble EAPOL frames**

› Sometimes only accepted while connecting
› 2\(^{nd}\) subframe of aggregate frame can contain arbitrary data
› Examples: Huawei Y6’, Nexus 5X, FreeBSD, LANCOM APs

→ Can **inject frames** independent of network config
Implementation flaws with other impact
Non-consecutive packet numbers (CVE-2020-26146)

Accepting fragments with non-consecutive packet numbers
› Related fragments must have consecutive packet numbers
› But almost nobody checks this! Only Linux does.

Can abuse this to exfiltrate data sent by a client if:
› The client is tricked into visiting the attacker’s server
› The client sends fragmented frames
Mixed plain/encrypted fragments (CVE-2020-26147)

Some reassemble **mixed plaintext and encrypted fragments**

› Practically all devices are affected

Can **abuse to inject frames**

› If 1st fragment must be encrypted:
  » Inject frames when combined with other vulnerabilities (non-trivial)

› If last fragment must be encrypted:
  » Inject frames when another device sends fragmented frames
Pre-auth EAPOL forwarding (CVE-2020-26139)

Some APs forwards EAPOL frames before sender is authenticated

› Examples: Net/FreeBSD APs and $\frac{2}{4}$ home routers

Arrows:
1. Associate
2. EAPOL frame
3. EAPOL frame

→ Abuse to inject frames in combination with aggregation attack (CVE-2020-24588)
No fragmentation support (CVE-2020-26142)

Some devices don’t support fragmentation

› They *treat fragmented frames as full frames*
› Examples: OpenBSD and ESP12-F

Abuse to *inject frames* when:

› Another device sends fragmented frames
› This other device visits the attacker’s server
Discussion
Practicality vs. impact

Perhaps we’re lucky:
› Widespread flaws $\rightarrow$ relatively trickly to exploit in practice
› Trivial to exploit flaws $\rightarrow$ not widespread in practice (?)

Important concerns remain:
› Significant #devices affected by trivial to exploit flaws
› Every Wi-Fi device affected by one or more flaws
› Combining flaws increases practicality of certain attacks

$\rightarrow$ Patch now before attack improve!