Closed Issue created 10 months ago by **Sharon Brizinov**

RTMPT dissector inf loop - 100% cpu - denial of service

Summary

It is possible to reach an infinite loop in the RTMPT dissector by generating a specifically crafted RTMP packet due to int underflow vulnerability in the AMF parsing part of the dissector. The packet will consume 100% core cpu, which eventually lead to a denial of service via packet injection or crafted capture file.

I believe that the bug was introduced to the code following this commit - #5351 (comment 400649448)

If that's true, it means that the bug exists for approximately 11 years over many tshark releases.

Technical details

Real-Time Messaging Protocol (RTMP) is a communication protocol for streaming audio, video, and data over the Internet which was mainly used by Flash Player. RTMP is a binary protocol over tcp port 1935.

The protocol enables to encode Action Message Format (AMF) data and the parsing of AMF object is mainly done by the `rtmpt_get_amf_param` or `rtmpt_get_amf_txid` functions. Before parsing AMF param, it is needed to understand its size. Therefore, `rtmpt_get_amf_length` is called to get the item param size. However, it is possible to reach an infinite loop condition within `rtmpt_get_amf_length` by carefully crafting AMF message as follows: The function has a main `while` loop with the following conditions: `while (rv == 0 || depth > 0)` so we need to make sure `rv == 0`. In the first round all variable are initialised to 0.

```
553  static gint
554  rtmpt_get_amf_length(tvbuff_t *tvb, gint offset)
555  {
556          guint8  iObjType;
557          gint    remain  = tvb_reported_length_remaining(tvb, offset);
558          guint32 depth   = 0;
559          gint    itemlen = 0;
560          gint    rv      = 0;
561
562          while (rv == 0 || depth > 0) {
563
564                  if (depth > 0) {
565                          if (remain-rv < 2)
566                                  return remain;
567                          itemlen = tvb_get_ntohs(tvb, offset: offset+rv) + 2;
568                          if (remain-rv<itemlen+1)
569                                  return remain;
570                          rv += itemlen;
571                  }
572
573                  if (remain-rv < 1)
574                          return remain;
575                  iObjType = tvb_get_guint8(tvb, offset: offset+rv);
576
```

We are skipping all these checks because all args are init to 0

The first meaningful line reads the object type: `iObjType = tvb_get_guint8(tvb, offset+rv);` Since we control the data, we can encode a speicifc object type that has a dynamic size. For example - `AMF0_STRING` (+3) or `AMF0_XML` (+5). These types allow to read the item length from the buffer using `tvb_get_ntohl`. However, `itemlen` is defined as a signed int `gint`, which means we can cause an int underflow by providing a size such as `0xfffffffd` (-3) `0xfffffffb` (-5). Then, since `itemlen` will remain zero, the loop will repeat itself because `rv` is affected by the `itemlen`: `rv += itemlen;` and so `rv` will remain zero forever and the loop will continue forerver.

```
case AMF0_STRING:
        if (remain-rv < 3)
                return remain;
        itemlen = tvb_get_ntohs(tvb, offset: offset+rv+1) + 3;
        break;
case AMF0_NULL:
case AMF0_UNDEFINED:
case AMF0_UNSUPPORTED:
        itemlen= 1;
        break;
case AMF0_DATE:
        itemlen = 11;
        break;
case AMF0_LONG_STRING:
case AMF0_XML:
        if (remain-rv < 5)
                return remain;
        itemlen = tvb_get_ntohl(tvb, offset: offset+rv+1) + 5;
        break;
```

itemlen is a signed int, we can underflow it

Relevant functions: `rtmpt_get_amf_txid --> rtmpt_get_amf_length`

Steps to reproduce

Open the provided pcaps with tshark/wireshark poc

What is the current bug behavior?

Wireshark/tshark will be stuck in an endless loop due to an int underflow bug.

What is the expected correct behavior?

Wireshark/tshark shouldn't be stuck in an endless loop.

Sample capture file

Attached poc

Relevant logs and/or screenshots

0:00 / 0:17

poc

Build information

```
Version 3.6.0 (v3.6.0-0-g3a34e44d02c9)
```

Edited 10 months ago by Sharon Brizinov

To upload designs, you'll need to enable LFS and have an admin enable hashed storage. More information

Tasks 0

No tasks are currently assigned. Use tasks to break down this issue into smaller parts.

Linked items 0

Link issues together to show that they're related or that one is blocking others. Learn more.

Related merge requests 3

rtmpt: limit the number of iterations in rtmpt_get_amf_length().
!5660

rtmpt: limit the number of iterations in rtmpt_get_amf_length().
!6166

rtmpt: limit the number of iterations in rtmpt_get_amf_length().
!6167

When these merge requests are accepted, this issue will be closed automatically.

Activity

Sharon Brizinov changed title from **RTMPT dissector excessive memory and CPU consumption - denial of service** to **RTMPT dissector inf loop - 100% cpu - denial of service** 10 months ago

Sharon Brizinov changed the description 10 months ago ·

Uli Heilmeier added lib wireshark scoped label 10 months ago

Dario Lombardo @crondaemon · 10 months ago Developer

Thanks for the outstanding report! I will look into it.

Dario Lombardo assigned to @crondaemon 10 months ago

Dario Lombardo mentioned in merge request !5660 (merged) 10 months ago

Dario Lombardo @crondaemon · 10 months ago Developer

@sean007 I pushed a merge request for this. Can you test it?

Sharon Brizinov @sean007 · 10 months ago Author Contributor

@crondaemon thanks for the quick action! I added there a couple of comments.

Dario Lombardo closed via commit 24403a9a 10 months ago

Sharon Brizinov @sean007 · 10 months ago Author Contributor

@crondaemon can you please add `crash` label to this issue?

Edited by Sharon Brizinov 10 months ago

Sharon Brizinov @sean007 · 10 months ago Author Contributor

sorry for bumping again - @crondaemon / @uhei can you please mark this as crash?

Please register or sign in to reply

Dario Lombardo added crash label 10 months ago

Gerald Combs made the issue visible to everyone 9 months ago

Gerald Combs mentioned in merge request !6166 (merged) 9 months ago

Gerald Combs mentioned in merge request !6167 (merged) 9 months ago

Gerald Combs @geraldcombs · 9 months ago Owner

It looks like this is a true infinite loop and not bounded by gui.max_tree_items, so I plan on requesting a separate CVE.

Dario Lombardo mentioned in commit b36df114 9 months ago

Dario Lombardo mentioned in commit 0bf7ccb2 9 months ago

Gerald Combs @geraldcombs · 9 months ago Owner

This was assigned CVE-2022-0586.

Please register or sign in to reply