README.md

Open5gs - A memory leak in PFCP protocol processing crashes UPF causing DoS

Recently, we discovered a logic vulnerability that may cause Open5gs UPF to crash during a code audit of Open5gs Ver2.4.11. The specific causes of the vulnerability are as follows:

Vulnerability description

When processing PFCP packet, a memory leak in UPF `src/upf/pfcp-path.c` from open5gs causing a DoS vulnerability.

UPF pfcp-path

Function `pfcp_recv_cb` from `src/upf/pfcp-path.c` will be called when receiving pfcp connection.

> src/upf/pfcp-path.c

```
static void pfcp_recv_cb(short when, ogs_socket_t fd, void *data)
{
```

```
    ...
```

`pfcp_node` will be allocated by calling `ogs_pfcp_node_add`.

> src/upf/pfcp-path.c

```
    node = ogs_pfcp_node_find(&ogs_pfcp_self()->pfcp_peer_list, &from);
    if (!node) {
        node = ogs_pfcp_node_add(&ogs_pfcp_self()->pfcp_peer_list, &from);
        ogs_assert(node);

        node->sock = data;
        pfcp_node_fsm_init(node, false);
    }
    ...
```

`pfcp_node` is allocated from `ogs_pfcp_node_pool` and appended to `pfcp_peer_list` in `ogs_pfcp_node_add`.

> lib/pfcp/context.c

```
ogs_pfcp_node_t *ogs_pfcp_node_new(ogs_sockaddr_t *sa_list)
{
    ogs_pfcp_node_t *node = NULL;

    ogs_assert(sa_list);

    ogs_pool_alloc(&ogs_pfcp_node_pool, &node);
    ogs_assert(node);
    memset(node, 0, sizeof(ogs_pfcp_node_t));

    node->sa_list = sa_list;

    ogs_list_init(&node->local_list);
    ogs_list_init(&node->remote_list);

    ogs_list_init(&node->gtpu_resource_list);

    return node;
}


ogs_pfcp_node_t *ogs_pfcp_node_add(
        ogs_list_t *list, ogs_sockaddr_t *addr)
{
    ogs_pfcp_node_t *node = NULL;
```

```
    ogs_sockaddr_t *new = NULL;

    ogs_assert(list);
    ogs_assert(addr);

    ogs_assert(OGS_OK == ogs_copyaddrinfo(&new, addr));
    node = ogs_pfcp_node_new(new);

    ogs_assert(node);
    memcpy(&node->addr, new, sizeof node->addr);

    ogs_list_add(list, node);

    return node;
}
```

Instead of freeing the nodes after using or encountering an error, these nodes are freed only after the termination of UPF by calling function `ogs_pfcp_context_final` .

So making more than 64 pfcp connections will crash the UPF causing DoS.

ogs_pfcp_node_pool

The size of `ogs_pfcp_node_pool` is defined as 64.

> lib/app/ogs-context.c

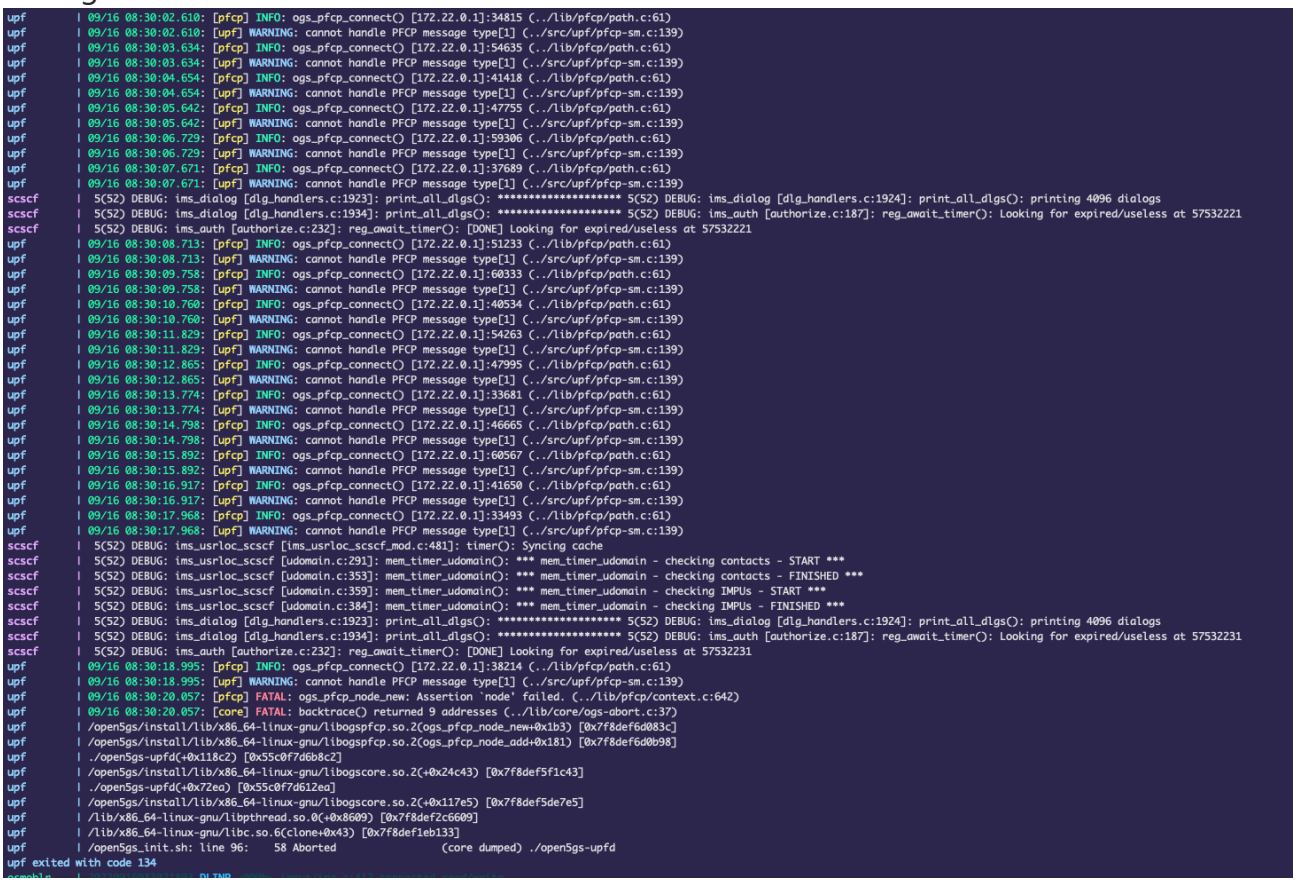
```
#define MAX_NUM_OF_UE               1024    /* Num of UEs */
#define MAX_NUM_OF_PEER             64      /* Num of Peer */

    self.max.ue = MAX_NUM_OF_UE;
    self.max.peer = MAX_NUM_OF_PEER;



static void recalculate_pool_size(void)
{
    ...
    self.pool.nf = self.max.peer;
    ...
}
```

> lib/pfcp/context.c

```
ogs_pool_init(&ogs_pfcp_node_pool, ogs_app()->pool.nf);
```

POC

The vulnerability can be triggered simply by sending more than 64 invalid pfcp packets through different sockets.



Upadate

We have reported this vulnerability to the vendor through email at 19 Sep 2022, but this bug has not been fixed yet.

Acknowledgment

Credit to @ToughRunner,@HenryzhaoH,@leonW7 from Shanghai Jiao Tong University.

Releases

No releases published

Packages

No packages published