代码审计 - Lua 扩展 Nginx 导致的 RCE

0x00 前言

应朋友之邀，分享一个去年遇到的 Lua RCE 案例。

0x01 简介

在这个案例中，该应用使用了 lua 扩展 nginx 功能，在调用 io.popen 处理上传包时没有对参数进行过滤，从而导致命令注入。

0x02 代码分析

分析 nginx 配置文件

```
nginx/conf/app.conf
```

定位到

```
location /sdk/file/uploadfile {
  more_set_headers 'Access-Control-Allow-Origin: *';
  client_max_body_size 50M;
  content_by_lua_file 'lualib/uploadfile.lua';
}
```

- lualib/uploadfile.lua

```
args = ngx.req.get_uri_args()
room_id = args["room_id"]
user_id = args["user_id"]
ngx.log(ngx.ERR, room_id, "/", user_id)
if not room_id or room_id == ngx.null or not user_id or user_id == ngx.null then
  local t = { ret = 10001, msg = "args room_id or user_id required" }
  ngx.say(cjson.encode(t))
  return
```

```
end

if not form then
  ngx.log(ngx.ERR, err)
  local t = { ret = 10910, msg = "new upload fail. msg:"..err, test = path }
  --ngx.say("upload new fail")
  ngx.say(cjson.encode(t))
  return
end
form:set_timeout(0)
local filename

function get_filename(res)
  local filename = ngx.re.match(res, '(.+)filename="(.+)"(.*)')
  if filename then
    return filename[2]
  end
end

local osfilepath = create_user_dir(room_id, user_id)
```

需要满足的条件

- room_id、user_id不为空
- 为上传表单

满足条件后会调用到 create_user_dir 方法

```
local function create_user_dir(room_id, user_id)
  local t = io.popen("pwd")
  local pwd = t:read("*all")
  pwd = string.sub(pwd, 1, -2)

  local stat_dir = pwd .. "/nginx/static/download/resources/web/tmp"
  local cmd = io.popen("ls "..stat_dir)
  if not cmd:read("*a") or cmd:read("*a") == "" then
    os.execute("mkdir ".. stat_dir)
    ngx.log(ngx.ERR, stat_dir)
  end

  stat_dir = stat_dir .. "/" .. room_id
  cmd = io.popen("ls "..stat_dir)

  ...
end
```

拼接参数 room_id，然后调用到命令执行 sink io.popen 从而导致命令注入。

0x03 小结

漏洞原理很简单，但是这种漏洞场景还是蛮常见的。

LEAVE A REPLY