

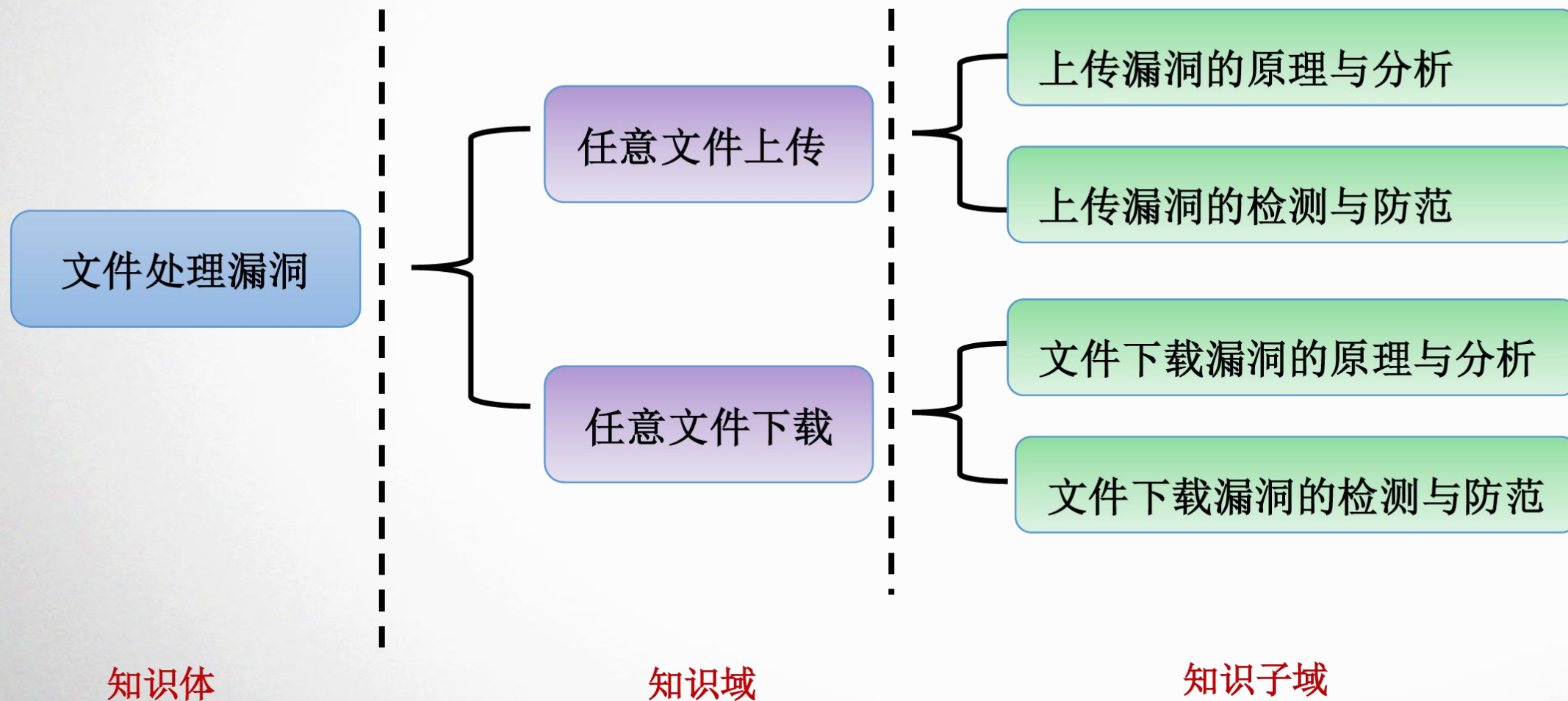


# CISP-PTE

## Web 安全基础(5) – 文件处理漏洞

主讲：

# 文件处理漏洞



# 任意文件上传

# 任意文件上传

---

- 通过本知识域，我们会：
  - 上传漏洞的原理
    - 了解任意文件上传漏洞产生的原因
    - 了解服务端语言对上传文件类型限制方法
  - 上传漏洞的检测与防范
    - 了解任意文件上传漏洞的危害
    - 掌握上传漏洞的检测思路和修复方法

# 任意文件上传漏洞产生的原因

- Web应用程序在处理用户上传的文件操作时，如果用户上传文件的路径、文件名、扩展名成为用户可控数据，就会导致直接上传脚本木马到web服务器上，直接控制web服务器。
- 文件上传时检查不严。
  - 没有进行文件格式检查。
  - 在客户端进行了格式检查 – 很容易绕过
  - 一些应用虽然在服务器端进行了不严格的黑 / 白名单检查
    - 忽略大小写，将.php改为.Php即可绕过检查
    - 忽略了%00截断符， xxx.php%00.jpg保存时变成xxx.php
  - 只对文件类型（Content-Type）进行检查
- 文件上传后修改文件名时处理不当 - 允许用户修改文件名后缀



# HTTP请求 – POST (续)

---

## ➤ 混合模式

➤ 有文件上传时常用的方法。可以接受同时提交不同类型的数据

➤ 表单中，可以把类型更改为file就可以上传文件

`<input type="file" name="file" id="file" />`

➤ 类型后面一般会跟boundary来告知数据区域分隔符

➤ 每个数据都可以单独说明数据类型

➤ 获取文件时，可以使用相应参数：

➤ PHP代码如下：

`$_FILES["file"]["name"]` - 文件名

`$_FILES["file"]["type"]` - 类型

`$_FILES["file"]["size"]` - 文件大小

`$_FILES["file"]["tmp_name"]` - 临时文件路径

# 服务端语言对上传文件类型限制方法

---

- 对文件类型进行限制 (Content-Type)
  - 此类设置，很容易被绕过。直接使用burpsuite等软件更改为允许的类型
- 对文件名后缀进行限制 (黑名单, 白名单)
  - 可能使用%00等截断方式绕过
  - 使用大小写方式绕过黑名单

# 任意文件上传漏洞危害

---

- 上传任意文件，可能会造成
  - 文件是一个webshell，可以任意执行系统命令
  - 与后台数据库链接，任意执行数据库命令
  - 把本服务器当作跳板，访问局域网内任意服务器
- 任意文件上传漏洞可能会让用户完全控制本机，直接获取登陆权限。



# 任意文件上传漏洞检测方法 - 检测代码

---

## ➤ 检测点:

- 检测所有可以上传漏洞的页面

## ➤ 检测方法:

- 是使用什么方式来限制文件类型
  - 后缀名
  - Content-Type
- 是否对文件名进行重命名
- 是否允许用户重命名文件名

# 任意文件上传漏洞检测方法 - 检测功能

---

- 检测点：
  - 检测所有可以上传漏洞的页面
- 上传检测方法：
  - 尝试上传多后缀文件，并查看时候能被截断
    - a.php.jpg
  - 尝试上传文件，与Content-Type不符
  - 尝试更改文件名
- 检测方法
  - 是否能执行之前上传漏洞里的代码

# 如何防御任意文件上传漏洞

---

- 文件上传的目录设置为不可执行
- 判断文件类型
  - 不仅仅是Content-Type 来进行判断，也需要对后缀名进行判断
  - 使用白名单策略
  - 对于图片处理，使用resize（或类似）函数来破坏源代码
- 使用随机数来存储文件
  - 防止.php.jpg等多后缀引起的漏洞
- 单独设置文件服务器的域名
  - 这个服务器，不能运行任何动态网页
  - 只能静态访问HTML、图片等

# 任意文件下载

# 任意文件下载

---

- 通过本知识域，我们会：
  - 文件下载漏洞的原理
    - 了解什么是文件下载漏洞
    - 掌握通过文件下载漏洞读取服务端文件的方法
  - 任意文件下载漏洞的检测与防护
    - 掌握能够通过代码审计和测试找到文件下载漏洞
    - 掌握修复文件下载漏洞的方法

# 文件下载漏洞概念

---

- 一些网站由于业务需求，往往需要提供文件查看或文件下载功能，但若对用户查看或下载的文件不做限制，则恶意用户就能够查看或下载任意敏感文件，这就是文件查看与下载漏洞
- 下载服务器任意文件，如脚本代码、服务及系统配置文件等 可用得到的代码进一步代码审计，得到更多可利用漏洞



# 通过文件下载漏洞读取服务端文件的方法

## ➤ 查找漏洞点:

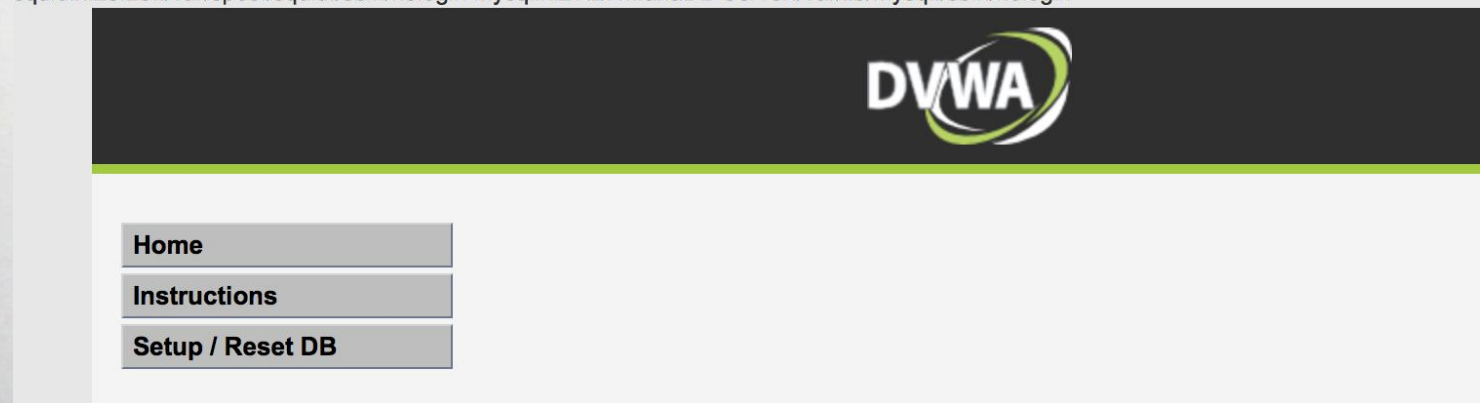
➤ <http://mydvwa.com/dvwa/vulnerabilities/fi/?page=file3.php>

## ➤ 把文件更改为要查看的文件

➤ <http://mydvwa.com/dvwa/vulnerabilities/fi/?page=/etc/passwd>

## ➤ 返回:

```
root:x:0:0:root:/root:/bin/bash bin:x:1:1:bin:/bin:/sbin/nologin daemon:x:2:2:daemon:/sbin:/sbin/nologin adm:x:3:4:adm:/var/adm:/sbin/nologin lp:x:
sync:x:5:0:sync:/bin:/bin/sync shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown halt:x:7:0:halt:/sbin:/sbin/halt mail:x:8:12:mail:/var/spool/mail:/sbin
operator:x:11:0:operator:/root:/sbin/nologin games:x:12:100:games:/usr/games:/sbin/nologin ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin nobody:x:
autoipd:x:170:170:Avahi IPv4LL Stack:/var/lib/avahi-autoipd:/sbin/nologin dbus:x:81:81:System message bus:./sbin/nologin polkitd:x:999:998:U:
tss:x:59:59:Account used by the trousers package to sandbox the tcsd daemon:/dev/null:/sbin/nologin postfix:x:89:89:/var/spool/postfix:/sbin/nol
SSH:/var/empty/sshd:/sbin/nologin nginx:x:998:997:nginx user:/var/cache/nginx:/sbin/nologin apache:x:48:48:Apache:/usr/share/httpd:/sbin/nolo
squid:x:23:23:/var/spool/squid:/sbin/nologin mysql:x:27:27:MariaDB Server:/var/lib/mysql:/sbin/nologin
```



# 如何查找任意文件下载漏洞

---

## ➤ 查找传入文件名的参数

- 导入文件等参数，要是直接输入文件名，就有可能有注入点

- 注意如下几个参数名：

- RealPath, FilePath, filepath, Path, path, inputFile, url, urls, Lang, dis, data, readfile, filep, src, menu, META-INF, WEB-INF

## ➤ 代码中如何查找漏洞：

- PHP为例，有如下等代码，就有可能存在任意文件下载漏洞

- readfile

- fopen

- file\_get\_contents

# 敏感文件路径 - Window

---

## ➤ Windows:

- C:\boot.ini //查看系统版
- C:\Windows\System32\inetsrv\MetaBase.xml //IIS配置文
- C:\Windows\repair\sam //存储系统初次安装的密码
- C:\Program Files\mysql\my.ini //Mysql配置
- C:\Program Files\mysql\data\mysql\user.MYD //Mysql root
- C:\Windows\php.ini //php配置信息
- C:\Windows\my.ini //Mysql配置信息

# 敏感文件路径 - Linux

---

## ➤ Linux:

- /root/.ssh/authorized\_keys
- /root/.ssh/id\_rsa
- /root/.ssh/id\_rsa.keystore
- /root/.ssh/known\_hosts
- /etc/passwd /etc/shadow
- /etc/my.cnf
- /etc/httpd/conf/httpd.conf
- /root/.bash\_history
- /root/.mysql\_history /proc/self/fd/fd[0-9]\*(文件标识符)
- /proc/mounts
- /porc/config.gz

# 修复文件下载漏洞

---

## ➤ PHP为例:

- 过滤.(点), 使用户在url中不能回溯上级目录
- 正则严格判断用户输入参数的格式
- php.ini配置open\_basedir限定文件访问范围