IOT设备中NTP服务的RCE漏洞分析

mosin / 2018-02-08 12:54:00 / 浏览数 4074 技术文章 技术文章 顶(0) 踩(0)

概述：
在对某些厂家的IOT网关设备进行检测时，发现了一个RCE漏洞，这个漏洞存在于大多数网关路由器设备中，漏洞点位于更新ntp中。这些漏洞的利用是需要条件的，那就是
当然，如果你想测试这个漏洞，最好找以前的版本（光猫、路由器）测试，现在最新的版本大多数都已修复，如果你恰好遇到存在此漏洞的设备，那么祝君好运!

漏洞分析
在我们传入参数之前，httpd程序（http服务器）会对传过来的URL进行CGI解析，随后选择调用sntp程序并传入参数。怎么调用这个不是我们要分析的重点（虽然这个也是
在SNTP程序文件中，它首先获取参数的传入

随后把参数传入了ntpdate文件，bin/ntpdate ntpserver地址

在获取了ntpserver地址后，进行了系统命令调用

第一次system执行调用是初始化ntpdate程序（清理关闭干扰程序），随后system执行nptdate程序，获取当前ntp服务器的时间，加之写入配置文件。

漏洞利用

我们来看下漏洞利用点，漏洞发生在设备的时间设定功能上

因为这个漏洞是隐式RCE，所以没有返回，我们只能进系统进行验证。
根据以上分析，我们后台监控看下

可以看到，sntp成功的调用了ntpdate程序来获取时间，而且ntpserver服务器的参数是我们可以控制的。
这里我们需要输入分隔符";"，这样，我们就能够执行多行命令了。
我们向tmp目录写入test.txt文件

后台监控看下是否利用成功

在这里，我们还是来看下前端代码吧，因为代码太长，所以选择主要函数讲解。
在我们提交保存按钮后，会调用btnApply()函数，我们跟进

```
function btnApply() {
 var loc = 'sntpcfg.cgi?ntp_enabled=';
 with( document.forms[0] ) {
   if( ntpEnabled.checked ) {
     loc += '1&ntpServer1=';
     if( ntpServer1.selectedIndex == ntpServers.length ) {
       if( ntpServerOther1.value.length == 0 ) { // == Other
         alert('■■■■■■■■■"■■"■■■■"■■"■■■■');
         return;
       } else {
         loc += ntpServerOther1.value;
       }
     } else {
       loc += ntpServer1[ntpServer1.selectedIndex].value;
     }

     loc += '&ntpServer2=';
     if( ntpServer2.selectedIndex == ntpServers.length+1 ) {
       if( ntpServerOther2.value.length == 0 ) { // == Other
         alert('■■■■■■■■■"■■"■■■■"■■"■■■■');
         return;
       } else {
         loc += ntpServerOther2.value;
       }
     } else {
       if( ntpServer2.selectedIndex > 0 )
         loc += ntpServer2[ntpServer2.selectedIndex].value;
     }

     loc += '&ntpServer3=';
     if( ntpServer3.selectedIndex == ntpServers.length+1 ) {
       if( ntpServerOther3.value.length == 0 ) { // == Other
         alert('■■■■■■■■■"■■"■■■■"■■"■■■■');
         return;
```

```
      } else {
        loc += ntpServerOther3.value;
      }
    } else {
      if( ntpServer3.selectedIndex > 0 )
        loc += ntpServer3[ntpServer3.selectedIndex].value;
    }
    loc += '&ntpServer4=';
    if( ntpServer4.selectedIndex == ntpServers.length+1 ) {
      if( ntpServerOther4.value.length == 0 ) { // == Other
        alert('■■■■■■■■“■■”■■■“■■”■■■');
        return;
      } else {
        loc += ntpServerOther4.value;
      }
    } else {
      if( ntpServer4.selectedIndex > 0 )
        loc += ntpServer4[ntpServer4.selectedIndex].value;
    }
    loc += '&ntpServer5=';
    if( ntpServer5.selectedIndex == ntpServers.length+1 ) {
      if( ntpServerOther5.value.length == 0 ) { // == Other
        alert('■■■■■■■■“■■”■■■“■■”■■■');
        return;
      } else {
        loc += ntpServerOther5.value;
      }
    } else {
      if( ntpServer5.selectedIndex > 0 )
        loc += ntpServer5[ntpServer5.selectedIndex].value;
    }

    loc += '&timezone_offset=' + cboTimeZone[cboTimeZone.selectedIndex].value;
    loc += '&timezone=' + getTimeZoneName(cboTimeZone.selectedIndex);
    loc += '&ntpWan=' + ntpWan.value;
    loc += '&use_dst=0';

    var ntpIntervalVal = parseInt(ntpInterval.value);
    if(isNaN(ntpIntervalVal) || ntpIntervalVal < 3600 || ntpIntervalVal > 604800){
        alert('■■■■■■■3600-604800');
        return;
    }
     loc += '&ntpInterval=' + ntpIntervalVal;
  } else {
    loc += '0';
  }
 }
 loc += '&sessionKey=' + sessionKey;
 var code = 'location="' + loc + '"';
 eval(code);
}
```

函数先对是否开启自动更新时间进行判断，随后进入操作。
在我们设置完参数后，函数会把我们提交过来的参数经过几个步骤的参数拼接，然后用eval()函数进行提交操作，最后httpd调用sntp程序对参数进行操作。

我们可以看到，在一系列的操作中，没有任何的函数对我们提交的参数进行过滤和拦截，最后此漏洞的发生还是在过滤不严上面。
下面我们来看下华为和中兴对NTP漏洞的防范，他们的代码一样。。。
在我们提交ntpserver地址后，会对我们自定义的值进行检查和过滤,由isTValidName()函数进行,只有合规的参数才能进入后面操作

```
if(isTValidName(ntpServerOther1.value) == false)
{
  AlertEx('■■■SNTP■■■■■■■■■');
  return;
}
Form.addParameter('NTPServer1',ntpServerOther1.value)
```

我们来跟进
isTValidName()函数，可以看到，对大多数的特殊字符进行了检测，这些字符正是我们执行命令需要的，有了这个安全函数的检测，漏洞自然就不存在了。

```
function isTValidName(name) {
  var i = 0;
  var unsafeString = "\"<>%\\^[]`\+\$\,='#&:;*/{} \t";
  for ( i = 0; i < name.length; i++ ) {
     for( j = 0; j < unsafeString.length; j++)
       if ( (name.charAt(i)) == unsafeString.charAt(j) )
        return false;
  }
 return true;
}
```

点击收藏 | 1 关注 | 1

上一篇：Java反序列化漏洞-玄铁重剑之C... 下一篇：Java反序列化漏洞从入门到深入

1. 2 条回复

阿尔百思科 2018-02-08 14:13:03

因为前端JS 加过滤所以漏洞不存在这个我不是很懂。。。

0 回复Ta

mosin 2018-02-08 15:25:51

@1683183204547782
是这样的，前端JS过滤，我们可以抓包改包对吧，问题是，有些设备是有多重检测的，比如中兴这个检测，在提交了URL过来之后，是还会再一次的对各参数进行检测，

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板