S3/Blob Manipulation (version 1.0)

**Cloud Service Label: IaaS, PaaS**

Description

By default S3/Blobs are accessible from the public internet if the S3/blob identifier is known and you have the proper credentials. Given this, an adversary with stolen storage credentials can read from existing buckets from anywhere on the Internet and exfiltrate the data stored on them. Malicious actors with more expansive cloud access can stage collected data from other cloud assets within a bucket and then remove contents from the bucket to any external host. They can be also serve out content via HTTPS to allow public consumption of new code.

Although often intended as a temporary solution while working on more challenging aspects of code development, storing code in cloud storage can unintentionally be incorporated into production, leaving consumers of this software vulnerable to code modification when a container or bucket is misconfigured. Adversaries can modify code stored in buckets to implant backdoor software to gain a foothold in what would otherwise be a well-protected enclave. If a storage bucket used for hosting code is ever deleted in the future, adversaries can reclaim the bucket name and without any special privileges stage code for unsuspecting people to invoke. This is especially problematic if web servers are pointing at the repo to fetch some forgotten piece of code.

Examples

|  |  |
| --- | --- |
| **Name** | **Description** |
| Capitol One Data Breach | Using previously obtained credentials attackers accessed sensitive data stored within S3 buckets and exfiltrated it. |
| Hackerone | The Rocket Chat install application was grabbing code from a publicly accessible S3 bucket. When the S3 bucket was deleted, this made the situation even worse. Once a bucket is deleted, anyone with access to AWS can create a bucket of the same name. In this case putting a malicious script within the bucket and waiting for unsuspecting users of the Rocket Chat installation script to fetch and run their payload. |
| Subjack | Open source tool that automates the search for s3 buckets and other cloud resources that are registered in DNS but which no longer exist indicating a possible hijacking opportunity. |
| Scout Suite | Scout Suite is an open source multi-cloud security tool that will check exposed storage APIs for misconfigurations and risks. The tool is compatible with cloud providers like AWS, Azure, GCP, and Alpha versions for Alibaba Cloud and Oracle Cloud Infrastructure. |
| Prowler | Prowler is open source tool on GitHub checks security best practices against AWS CIS Benchmark standards. This includes coverage of IAM, logging, monitoring, networking, CIS Levels 1 and 2, forensics, and even GDPR and HIPAA. Prowler also looks for resources that may or may not be intentionally configured in a way that violates the standards. For example, it will check for an open S3 bucket or a security group with an open port. |
| S3 Inspector | S3 Inspector is an open source tool used to check AWS S3 bucket permissions. The tool will check all account buckets for public access and provide a list of URLs that can be used to access that said public bucket. S3 Inspector can be used as a normal CLI utility or as an AWS Lambda function. |
| DNS Enumeration | An OWASP Bay Area presentation in August 2019 covered the process of performing a port scan on an IP address found in the TXT records of a DNS enumeration process and revealed ssh keys that were used to log into servers hosted with AWS. These private keys were found in a zip file of an open S3 bucket. An RDS database server configuration file was found and username and password hashes were subsequently dumped. |
| S3\_download\_Bucket | Publicly available Pacu module that scans the current account for AWS buckets and prints/stores as much data as it can about each one. With no arguments, this module will enumerate all buckets the account has access to, then prompt you to download all files in the bucket or not. |

Mitigations

|  |  |  |
| --- | --- | --- |
| **Mitigation** | | **Description** |
| Use [Network Traffic](https://attack.mitre.org/mitigations/M1037)Policies | | Create and apply AWS/Azure policies that restrict access to S3 buckets or Azure Blobs from external networks. In Azure you can restrict access to blobs to certain networks only during blob creation. |
|  | AWS | An AWS environment can be configured with network ACLs (access control lists) to allow or deny inbound and outbound traffic. This can be accomplished by accessing Amazon VPC and navigating to either inbound or outbound rules depending on the rule the user wishes to add and they can be added, removed, or edited from that panel. Full details about ACLs and how to add rules in AWS can be found here: **https://docs.aws.amazon.com/vpc/latest/userguide/vpc-network-acls.html.** |
|  | Azure | In Azure storage resources can be tied exclusively to a particular virtual network reducing the chances that it can be accessed externally or from other cloud assets. This can be done multiple ways including the Azure Portal, Azure PowerShell, and Azure CLI (Command Line Interface). Depending on the method used to implement this the procedure can vary, but will include the need to create a security group, create a network security group, associate that network security group with a specific subnet and then create security rules that are associated to the inbound and outbound rules for that subnet. Full details on how to configure this utilizing the various methods can be found below:  Azure PowerShell: **https://docs.microsoft.com/en-us/azure/virtual-network/tutorial-filter-network-traffic-powershell**  Azure CLI: **https://docs.microsoft.com/en-us/azure/virtual-network/tutorial-filter-network-traffic-cli** |
| Audit | | Frequently check permissions on cloud storage to ensure proper permissions are set to deny open or unprivileged access to resources. Consider using automated resource checkers such as CloudSploit or Divvycloud . |
|  | AWS | To perform an audit via AWS it is suggested to review information such as account details (credentials, users, groups, roles, etc), mobile applications, EC2 configurations, policies, and account activity. How to audit these different factors can be found in detail at: **https://docs.aws.amazon.com/general/latest/gr/aws-security-audit-guide.html.** |
|  | Azure | To perform an audit via Azure an administrator can review the audit logs that are recorded under Azure’s monitoring for active directory. The audit logs allow for filtering, as well as looking at users, groups, and enterprise specific information. Full details on how to access this information can be found at: **https://docs.microsoft.com/en-us/azure/active-directory/reports-monitoring/concept-audit-logs.** |
|  | GCP | To perform an audit via GCP the logs can be reviewed. GCP breaks this down into three categories; admin activity, data access, and system events. The audit logs can be viewed a few different ways- the console, API, or gcloud. Full details on how to view these logs, how to export, and for how to configure the retention period can be found here: **https://cloud.google.com/logging/docs/audit.** |
| Don’t Do it | | Don’t store executable code within cloud storage repos. |
| Encrypt Sensitive Information | | Encrypt data stored at rest in cloud storage. Managed encryption keys can be rotated by most providers. At a minimum, ensure an incident response plan to storage breach includes rotating the keys and test for impact on client applications. |
|  | AWS | To encrypt data at rest in an AWS environment first start by configuring the IAM roles and permissions. A policy will need to be created to specify that the data is to be encrypted and then the policy is attached to the instance. Full details on how to accomplish this can be found at: **https://aws.amazon.com/blogs/security/how-to-protect-data-at-rest-with-amazon-ec2-instance-store-encryption/**. |
|  | Azure | To encrypt data at rest in an Azure environment it can be done differently depending on the cloud service the customer is utilizing - normally a check box is presented enabling encryption for data at rest. For SaaS customers this can be enabled or available on each specific service. For PaaS customers there are specific storage and application platforms that are supported. In terms of IaaS customers Azure storage components such as queues and blobs are encrypted by selecting a check box during creation. Azure also allows customers to employ their own encryption in addition to the server side encryption provided by Azure –effectively creating two layers of encryption for all storage assets. Currently this must be done programtically as described here: **https://docs.microsoft.com/en-us/azure/virtual-network/tutorial-filter-network-traffic**  Encrypted compute allows for all managed disks, snapshots, and images to be encrypted utilizing a service managed key. The keys are stored in the Azure key vault. Full details on how to accomplish this can be found at: **https://docs.microsoft.com/en-us/azure/security/fundamentals/encryption-atrest.** |
| Least Privilege | | All access given to users in the cloud environment should be assigned by the necessary privileges needed for team members to complete their job responsibilities. Ensure that temporary access tokens are issued rather than permanent credentials, especially when access is being granted to entities outside of the internal security boundary . |
|  | AWS | Ensure only administrators have access to the S3 Permissions management actions. Grant actions like *List*, *Read*, and *Write* on a by-need basis, with particular attention on the *Write* action. Those with the *Write* action have the ability to delete or put objects into S3 buckets. To implement least privilege in an AWS environment IAM policies will be used. This gives the ability to allow users to perform list, read, write, permissions management, or tagging actions. AWS suggests utilizing *last accessed information* and A*WS CloudTrail event history* to get a better understanding of privileges that might be needed or reduced based on a specific role. Full details can be found at **https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#grant-least-privilege.** |
|  | Azure | Disable anonymous access to blob containers with the following steps:   1. Go to *Storage Accounts* 2. For each storage account, go to *Containers* under *BLOB SERVICE* 3. For each container, click *Access policy* 4. Ensure *Public access level* is set to *Private* |
| Microsoft Blob Storage Security Recommendations | | Microsoft’s article contains recommendations for blob storage related to data protection, identity and access management, networking, logging, and monitoring. Azure Security Center is highly recommended to protect resources within Azure. It also provides recommendations on how to fix vulnerabilities.  Full details for Microsoft’s security recommendations for blob storage can be found at [**https://docs.microsoft.com/en-us/azure/storage/blobs/security-recommendations**](https://docs.microsoft.com/en-us/azure/storage/blobs/security-recommendations)**.** |
| AWS S3 Security Best Practices | | Amazon’s article highlights best practices for S3 like ensuring S3 buckets are not publicly accessible, using IAM roles for applications services as related to S3, implementing encryption of data at rest, data in transit, and many others.  Full details for AWS’s security best practices for S3 can be found at [**https://docs.aws.amazon.com/AmazonS3/latest/dev/security-best-practices.html**](https://docs.aws.amazon.com/AmazonS3/latest/dev/security-best-practices.html)**.** |
| Multi-factor Authentication | | Use multi-factor authentication for user and privileged accounts. Do not manage Cloud portals from machines that perform user email and web browsing tasks. All users should be required to utilize two factor authentication. |
|  | AWS | This can be enforced by first creating a policy that would prohibit actions except those that allow a user to change their password or manage 2FA, then attaching a policy to a group that includes all user accounts where they can be allowed all access if they sign in with 2FA. Once these actions are completed it should be tested to verify the access is given correctly. To see full details on how to complete this view AWS documentation at: **https://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial\_users-self-manage-mfa-and-creds.html.** |
|  | Azure | This can be done by creating a MFA registration policy. It can than be assigned to all users (with the ability to exclude some if need be, but is not recommended). Make sure once the policy is created and added to users that it is then being enforced, once enforced it should be tested for verification. To see full details on how to complete this view Azure documentation at: **https://docs.microsoft.com/en-us/azure/active-directory/identity-protection/howto-identity-protection-configure-mfa-policy.** |
|  | GCP | This can be done by first enabling it on the current account being used by admin to assign the roles, then enable two factor on an instance by instance or project by project basis, then assigning the requirements based on IAM roles and applying it to all users. To see full details on how to complete this view Azure documentation at: **https://cloud.google.com/compute/docs/oslogin/setup-two-factor-authentication.** |
| Restrict File and Directory Permissions | | Users should have limited access to files and directories depending on their need for access. The file and directory permissions should be restricted on the basis of least privilege. |
|  | AWS | To manage the files and directory permissions in AWS, IAM policies can be used. This can be done by utilizing group policies and policy variables. The policy would be created specifying the folder, then the permissions attached to that folder (whether the user has access to list out the objects within the directory, if they have read permissions, if they have write permissions, etc.), lastly the group that it applies to would be specified. The users can that be added and removed from that group as needed. Full details on how this can be done is explained here: **https://aws.amazon.com/blogs/security/writing-iam-policies-grant-access-to-user-specific-folders-in-an-amazon-s3-bucket/.** |
|  | Azure | To manage the files and directory permissions in an Azure environment basic and advanced system defined controls. This will be dependent on the type of system being used (Windows, Linux, etc). The permissions will be set individually or by group using the system commands or controls needed.. Full details on how this can be done is explained here:[**https://docs.microsoft.com/en-us/azure/storage/files/storage-files-identity-ad-ds-configure-permissions**](https://docs.microsoft.com/en-us/azure/storage/files/storage-files-identity-ad-ds-configure-permissions)**.** |

Detection

|  |  |
| --- | --- |
| **Detection** | **Description** |
| Enable S3 Bucket Logging | To enable CloudTrail S3 bucket logging:   1. Navigate to CloudTrail console at Go to the Amazon CloudTrail console 2. Click Trails in the API activity history pane on the left 3. Sign into AWS Management Console and open the S3 console Sign in to the AWS Management Console and open the S3 console 4. Click on a bucket under *All Buckets* 5. Click on *Properties* 6. Under *Bucket:\_<bucket\_name>\_* click *Logging* 7. Ensure *Enabled* is checked |
| Create Log Metric Filters and Alarms for S3 | To create a metric filter and alarm:   1. Create a filter that checks for S3 bucket policy changes and the specific *<cloudtrail\_log\_group\_name>* 2. Create an SNS topic that the alarm will notify 3. Create an SNS subscription to the above topic 4. Create an alarm associated with the filter from step 1 and SNS topic in step 2 |
| Enable Azure Storage Logging | This is used to track how requests made to Azure Storage were authorized. Enabling logs provides visibility into whether a request was anonymous, made with an OAuth2.0 token, a shared key, or shared access signature (SAS). Full Azure Storage analytics logging details can be found at [**https://docs.microsoft.com/en-us/azure/storage/common/storage-analytics-logging?tabs=dotnet**](https://docs.microsoft.com/en-us/azure/storage/common/storage-analytics-logging?tabs=dotnet)**.** |

References

1. https://hackerone.com/reports/399166. Accessed 02/12/2020
2. techcommunity.microsoft.com/t5/azure-sentinel/hunting-for-capital-one-breach-ttps-in-aws-logs-using-azure/ba-p/1014258#. Accessed Feb. 21, 2020
3. <https://github.com/nccgroup/ScoutSuite>. Accessed July 27, 2020.
4. <https://github.com/toniblyx/prowler>. Accessed July 27, 2020.
5. <https://d0.awsstatic.com/whitepapers/compliance/AWS_CIS_Foundations_Benchmark.pdf>. Accessed July 27, 2020.
6. <https://github.com/clario-tech/s3-inspector>. Accessed July 27, 2020.
7. <https://blog.appsecco.com/getting-shell-and-data-access-in-aws-by-chaining-vulnerabilities-7630fa57c7ed>. Accessed July 27, 2020.
8. <https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html>. Accessed July 31, 2020.
9. <https://docs.microsoft.com/en-us/azure/storage/blobs/security-recommendations>. Accessed August 3, 2020.
10. <https://docs.aws.amazon.com/AmazonS3/latest/dev/security-best-practices.html>. Accessed August 3, 2020.
11. <https://docs.microsoft.com/en-us/azure/storage/common/storage-analytics-logging?tabs=dotnet>. Accessed August 3, 2020.