Game to Dethrone: A Least Privilege CTF

Wenjing Wu
Department of Computer Science
Portland State University
Portland, Oregon, US
wwu@pdx.edu

Wu-chang Feng
Department of Computer Science
Portland State University
Portland, Oregon, US
wuchang@pdx.edu

***Abstract*—Identity and Access Management (IAM) misconfiguration is one of the most critical threats to the security of cloud environments. As more infrastructure is being migrated to the cloud, the importance of following the principle of least privilege (PoLP) to mitigate security risks has significantly increased. Unfortunately, the mechanisms provided for doing so in the cloud are complex and substantially different than traditional legacy infrastructure. As a result, the number of practitioners that know how to secure cloud projects is insufficient compared to the number of cloud projects being deployed. To address the issue, this paper describes a Least Privilege CTF, a series of exercises that allows practitioners to practice applying least privileges on cloud deployments that are easily deployed with minimal cost.**

***Index Terms*—Cloud security, CTF, least-privileges, IAM**

I. Introduction

ue to the agility, scalability, and potential cost savings that cloud computing offers, businesses are shifting their workloads to public cloud providers [4]. Unfortunately, complicated fine-grained permissions and novel role-based access control mechanisms employed by cloud providers make the practice of security more difficult [17]. As a result, studies [2], [9] reveal that nearly 80% of companies have experienced at least one cloud data breach, with one of the top threats coming from improper IAM configuration [3], [10], [11].

In order to mitigate security risks, it is essential that cloud deployments practice the principle of least privilege (PoLP) [14] to ensure that users and applications only have the minimum privileges needed to perform their intended tasks. Unfortunately, doing so is difficult as cloud systems have become complex. As a result, misconfiguration is common and when discovered, very few of the issues are fixed. A recent study on DevOps security found that only 4% of security issues identified in production are dealt with after development [5]. Compounding this problem is the fact that developers often have limited experience with PoLP and are often introduced to cloud services using guides and walk-throughs that de-emphasize good security practices in favor of fast and frictionless adoption of the service.

While applying PoLP is critical to cloud security, there are few exercises that allow developers to practice it. Many cloud-based "capture-the-flag" exercises allow one to instead practice identifying vulnerabilities in cloud deployments and then practice exploiting them to achieve a goal. Such offensive-minded CTFs [12], [13], [15], [16], [18] show users how to leverage overprovisioned permissions to compromise a project, but do not afford opportunities for practitioners to practice reducing them. In an attempt to address this, this paper describes the design, implementation, and deployment of a set of exercises that allow practitioners to apply the principle of least privileges on Google Cloud Platform (GCP). Results from an initial study show that our Least Privilege CTF can effectively train developers to follow PoLP.

II. Google Cloud IAM

For cloud-based infrastructure, developers typically use a platform's Identity and Access Management (IAM) service to implement authorization rules and policies that govern access to cloud resources. IAM allows one to perform access control at a highly granular level with regard to who is allowed access and to what resource the access is allowed. When configured properly, IAM can provide protection for all resources and data via permissions and policies that are enforced on users and applications [1].

A. Basics

In its instantiation on Google Cloud Platform, there are three main components for implementing IAM: members, roles, and policies [7]. Members provide the "Identity" (e.g. authentication) component while roles and policies provide the "Access Management" (e.g. authorization) component. Once the identity of a user or system is verified, an access management process is executed that determines the privileges afforded to that user or system when accessing resources. Each is described below:

- *Members:* A member represents an authenticated identity. Such identities can be a Google user account (`userid@gmail.com`), a service account created for applications (`1234...@....gserviceaccount.com`), a Google group (`groupname@googlegroups.com`), or an identity domain (`...@example.com`). While the identities of the first three member types are email addresses, an identity domain is typically a domain name for an organization.
- *Roles:* A role is a collection of permissions where each permission specifies access to and operations allowed on a particular resource. Some examples of resources are projects, virtual machine instances, and storage buckets. Due to the sheer number of resources in a cloud project,

TABLE I
EXAMPLE PREDEFINED IAM ROLES FOR CLOUD STORAGE

Role	Permissions
Storage Object Viewer (roles/storage.objectViewer)	resourcemanager.projects.get resourcemanager.projects.list storage.objects.get storage.objects.list
Storage Object Admin (roles/storage.objectAdmin)	resourcemanager.projects.get resourcemanager.projects.list storage.objects.*
Storage Admin (roles/storage.admin)	firebase.projects.get resourcemanager.projects.get resourcemanager.projects.list storage.buckets.* storage.objects.*

there are thousands of permissions that can be specified. While one could assign individual permissions to each member, doing so is cumbersome. As a result, permissions are typically grouped and assigned to roles, which are then attached to members. There are three types of roles in Google Cloud IAM [8]: primitive, predefined and custom. Primitive (also known as Basic) roles are coarse, project-level roles that are managed by GCP and include project "Owner", "Editor", and "Viewer" roles. Such roles preceded the current version of IAM that supports fine-grained permissions and should be avoided due to the inability to reduce privileges in them appropriately. Predefined roles are also maintained by GCP and consist of collections of commonly grouped permissions for a specific service. Predefined roles specify permissions at a finer granularity than primitive roles and are made available to projects for convenience. Table I shows a number of the predefined roles for cloud storage. Note that each cloud platform resource and service such as Compute Engine, App Engine, and Cloud Functions have a similar set of pre-dedefined roles. Finally, there are instances when predefined roles do not exactly express an appropriate least privilege configuration for a project. In such cases, users can create custom roles and specify the exact set of permissions allowed. This allows one to apply the principle of least privileges at its finest granularity, but is the most complicated to set up, requiring the user to understand what individual permissions allow access to.

- *Policies:* A policy is a collection of bindings that bind members to roles and thus give the members access to the resources specified in each role. Multiple members can be bound to a particular role [7].

B. An example applying least-privileges

Consider a scenario in which two users need to list all of the objects in a storage bucket and view all of their contents. To begin with, we can define a member that is a Google group in which both users are added. Then, for a policy, we can attach a primitive role of "Viewer" that will enable the users to view all parts of the project (not just its buckets). The primitive role "Viewer", while having a limited set of permissions compared

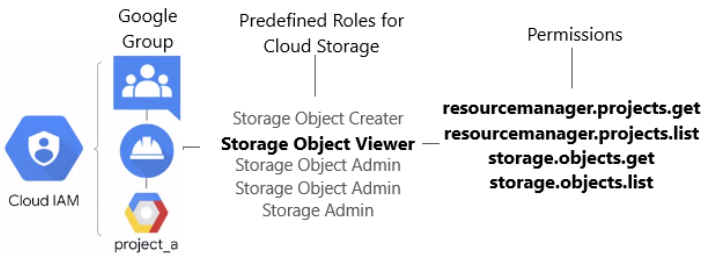

Fig. 1. Policy - Predefined Role

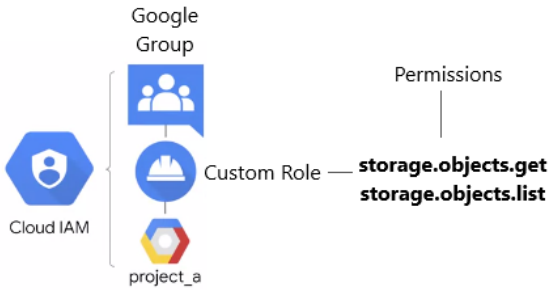

Fig. 2. Policy - Custom Role

to the primitive role "Owner", still has excessive permissions associated with it for the usage the two users require.

To address this, we might instead examine predefined roles defined for cloud storage as shown in Table I and attach a predefined role of "Storage Admin" (`roles/storage.admin`) with its associated permissions. Such a setting would allow the operations (`storage.objects.*`). However, this also has excessive permissions. In looking at other predefined roles in the table and their associated permissions with Cloud Storage, we notice that "Storage Object Viewer" has sufficient permissions to list objects in a bucket (`storage.objects.list`) and to retrieve an object's contents (`storage.objects.get`). Thus, binding "Storage Object Viewer" to the Google group would further reduce privileges as shown in Figure 1. In looking at the "Storage Object Viewer" predefined role, however, we notice that it provides two additional permissions that are not required by the two users. To apply the PoLP at its finest granularity, as Figure 2 shows, we can instead create a custom role and attach only the two permissions required, provide the custom role to a policy with the least-privilege setting and add the policy to the group.

III. LEAST PRIVILEGE CTF

To help train the application of PoLP towards cloud projects, the Least Privilege CTF consists of a set of exercises in which players attempt to reduce the privileges of a policy attached to a member using the different processes described previously.

A. Goals

The CTF has been designed with several goals in mind including:

- Level service account pd1-f-access-979621621703-sa is bound with an over-privileged **Predefined Role** to list an object(txt file) in a Cloud Storage Bucket.
- Your goal is to find a **Predefined Role** with minimum permissions but still enough to list the object.
- Use the check function to view role and permissions of pd1-f-access-979621621703-sa.
- Go to google cloud console IAM&Admin -> IAM -> MEMBERS, click the pencil icon to edit service acount pd1-f-access-979621621703-sa in the table.
- Replace the current role with the new one you found. To keep the process simple, do not leave the service account with empty role.
  Because if all roles are removed, service account will disappear from the IAM memeber table (can be added back via granting access).
- Refresh the check function to validate your answer.
  Hint: Look for a **Predefined Cloud Storage Role** in google doc.
- Note that we are not going to use the Predefined Lagacy Role
- You can submit a new request to this access function by pressing ENTER in the browser's address bar and see if sufficient permissions are granted to level SA.
  (The change may take a couple submits to show up.)

Bucket:

- pd1-bucket-979621621703 : secret_pd1.txt

Source code of function that checks what roles bind with pd1-f-access-979621621703 :

```
name = f'{RESOURCE_PREFIX}-bucket-{NONCE}'
request = storage_api.objects().list(bucket=name).execute()["items"][0]
bucket = name + ' :  ' + request["name"]
```

Fig. 3. Access Function - Access level pd1(PredefinedRole-Storage)

- *Scaffolded*: Each level should incrementally build on prior levels in order to support consistent player progression.
- *Consistent, straight-forward gameplay*: Exercises should provide explicit and detailed instructions that enable all players to complete. Goals for each level and level gameplay should be consistent in order to ensure players focus on the specific concepts and skills being targeted.
- *Immediate feedback*: Players should be able to receive results on their progress in each level as soon as possible as well as be given reasons why solution attempts might be failing.
- *Easily deployed*: Exercises should be easy to deploy and at minimal cost.
- *Extensible*: Developers should be able to easily add additional exercises to follow the evolution of the security mechanisms supported on the cloud platform.

B. Implementation

1) Scaffolded conceptual progression: Perhaps the most important aspect of an exercise is the progression of levels that players go through. Towards this end, the CTF follows the progression of concepts outlined in Section II. Initial levels expose players to the coarse, project-level, primitive roles that are prevalently used in "quickstart" walkthroughs for convenience, but are often overprovisioned. Players then progress to a variety of levels that show them how predefined roles can be applied to reduce the privileges from primitive roles. Finally, players are then asked to create custom roles to generate an exact set of permissions that a particular access pattern requires. There are currently 11 levels in the CTF. The levels are designed in a scaffolded style with incrementally increasing difficulties with the final level requiring players to employ custom and predefined roles with least-privilege settings across multiple project services including the Cloud Vision API, the Cloud Datastore service, and the Cloud Storage service.

2) Gameplay: To keep players focused on the conceptual tasks and to minimize the amount of time navigating the mechanics of an exercise, each level has the same structure in order to give players a familiar pattern from which to reduce privileges. Moreover, the mechanism for solving a level is exactly the mechanism one would perform in an actual project when reducing privileges. Specifically, each level contains two functions, an access function and a check function. Players begin by visiting the access function as shown in Figure 3, which has an overprovisioned set of permissions. The function

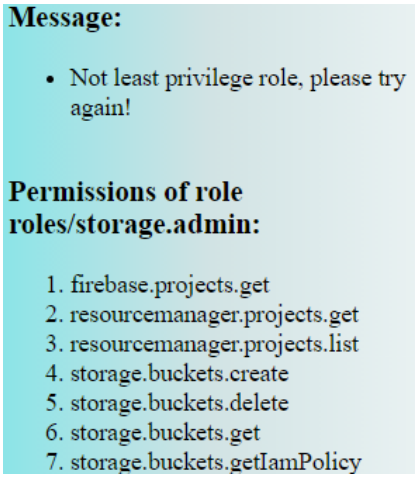

Fig. 4. Check Function - Check privileges of level pd1(PredefinedRole-Storage)

Score Board

Level	Score
PrimitiveRole-Project	0 / 10
PredefinedRole-Storage	0 / 10
PredefinedRole-Compute	0 / 10
PredefinedRole-Logging	0 / 10
PredefinedRole-Datastore	0 / 10
PredefinedRole-Vision	0 / 10
CustomRole-Project	0 / 10
CustomRole-Storage	0 / 10
CustomRole-Compute	0 / 10
CustomRole-Logging	0 / 10
CustomdRole-Vision	0 / 10
Sum / Total	0 / 110

Fig. 5. Scoreboard Function

delivers a web page that reveals its source code and the privileges it has been afforded. It also provides detailed instructions and links to material that can help solve the level. As the figure shows, the access function code requires permissions to list objects in a bucket, but has been granted a role that has given it additional privileges. From this, the level asks the users to visit the service account attached to the access function and change its role to a predefined role with least privileges. The access function contains a link to a check function that players can then visit after changing the role attached to the access function's service account. The check function lists the current roles and permissions that are attached to the access function and displays whether or not it has been set with least privileges. Figure 4 shows the corresponding output of the check function for the access function shown previously. As the figure shows, the predefined role that is currently assigned is `roles/storage.admin` and the check function fails since it is still overprovisioned for what the access function requires.

Similar to Jeopardy-style CTF exercises, the Least Privilege CTF also provides a summary scoreboard for the user to see which levels have been solved in order to allow them to track their progress easily and get immediate feedback on solutions. Figure 5 shows the scoreboard function. To make it easier for players to focus on the specific goal of each level, the type of role and the service that a level targets are included in the level name and joined by a dash (first column of Figure 5).

3) Deployment: Several cloud-based CTFs require individual levels to be brought up and down with explicit commands. For example, in both Cloud Goat and Thunder CTF, players create and destroy levels individually in order to play them. To streamline gameplay, the Least Privilege CTF deploys its entire set of levels with a single command allowing players to avoid the mechanics of running the CTF as much as possible. This is done via the platform's Cloud Deployment Manager service, an infrastructure-as-code solution that allows one to programmatically instantiate cloud resources in a consistent and repeatable manner. Using this service, all service accounts, roles, policies, access functions, check functions, and scoreboard functions for the CTF are instantiated all at once using a single command and YAML specification file at the beginning of the CTF. The launch time for all 11 levels takes around 4.5 minutes.

Figure 6 provides a more detailed illustration of how the CTF is deployed. When deployed, service accounts for both the level functions and the check functions are created. Then, for service accounts attached to level functions, roles with excessive permissions are bound to them and players are tasked with updating them to a least privilege setting. Each level function has a corresponding check function that validates whether the binding role(s) in the access function match(es) the correct answer. To perform this checking operation, a role with permissions to list and get roles and IAM policies is attached to the service account associated with the check function. The scoreboard function uses a similar approach in order to check role permissions across the CTF levels and calculate scores. The winning condition for the CTF is met when PoLP has been applied successfully to grant a minimum set of privileges across all level functions.

4) Serverless operation: To reduce costs, the CTF is predominantly implemented using Cloud Functions, a serverless function platform. For the CTF, all access and check functions are implemented as Cloud Functions. There are several advantages in this approach. First, Cloud Functions require no server management, which simplifies both development and deployment as one does not have to manage the operational server infrastructure. Second, serverless platforms scale the number of resources deployed based on usage and projects are charged only per function invocation. As a result, when the CTF is not actively in use, the infrastructure to handle its functions is brought down and the project incurs no charges. The infrastructure for each function is then instantly spun up upon the player's next access to it. The Cloud Function

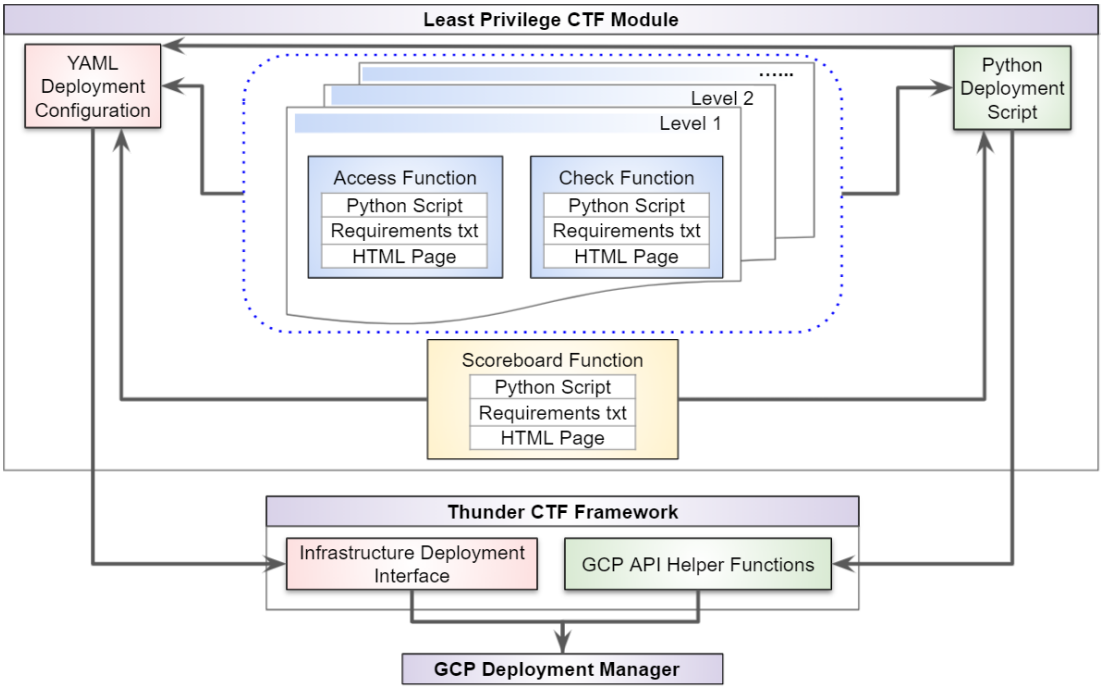

Fig. 6. Overall Structure

service [6] provides a free tier of 2 million invocations per month, which ensures that the exercise is free to use. Third, the use of individual serverless functions for the CTF allows one to easily implement levels, since each function can be assigned a unique service account with a primitive, predefined, or custom role attached to it, naturally aligning it with the conceptual content the CTF is attempting to teach. The separation also allows levels to be independently developed and provides isolation between them, making it easier to develop additional levels. As a result of its largely serverless approach, the CTF consumes a minimal number of resources with a single Compute Engine VM (for the privileges related to Compute Engine roles), a storage bucket, and the aforementioned Cloud Functions being utilized.

5) Extensibility: The Least Privilege CTF is built on the Thunder CTF framework, which implements a scaffolded, scenario-based CTF for practicing cloud security skills on GCP. Each Thunder CTF level includes a YAML development configuration, a Python deployment script and an HTML template that generates level hints [19]. The modular design and structure streamline the process of adding new levels. The Least Privilege CTF inherits the modular design from Thunder CTF framework with some modifications. In the Least Privilege CTF, each level consists of a pair of Cloud Functions triggered by HTTP requests. Each level's resources is independent of other levels, allowing them to be inserted and removed without impacting each other. Level designers first create the access function that serves as the landing page of the level. The access function is written in Python and uses Jinja to construct the landing page containing the level's instructions and hints. The function also contains code that interacts with other cloud services through APIs. To do so successfully, the service account associated with the access function is given an initial (overprovisioned) role. Players can invoke the access function in a web browser to see the level instructions and its functionality. The designer then creates the check function that examines the privileges associated with the service account attached to the access function. This is the main exercise. Players must determine whether the permissions associated with the service account are set in a least privilege manner. If not, then they are tasked with going to the IAM settings for the project and reducing its privileges. The check function, also written as a Cloud Function with an HTTP trigger, outputs a web page that indicates whether the privileges have been reduced appropriately and the level has been solved. The general URL format of the levels' access endpoints follow the pattern: (e.g. https://REGION-PROJECT_ID.cloudfunctions.net/FUNCTION_NAME).

IV. Evaluation

We deployed the Least Privilege CTF occurred in our Fall 2020 offering of Portland State University's CS 430/530 Internet, Web, and Cloud Systems course with graduate and upper-division undergraduate students. There were 60 students in the course. The first half of the 10-week course covers key concepts in networking, operating systems, web development, and databases before transitioning to their use in cloud computing environments. One lecture on Google Cloud Identity and Access Management was initially given and the Least Privilege CTF was then assigned to students at the beginning of the 5th week. The due date of the CTF was set at the end

TABLE II
HELPFULNESS RATINGS OF LEAST PRIVILEGE CTF (1=VERY UNHELPFUL, 2=SOMEWHAT UNHELPFUL, 3=NEITHER HELPFUL NOR UNHELPFUL, 4=SOMEWHAT HELPFUL, 5=VERY HELPFUL)

Question	1	2	3	4	5	Mean rating
Q1	1	3	1	9	21	4.31
Q2	0	1	4	13	17	4.31
Q3	3	3	4	8	17	3.91

of the 5th week, leaving about a week for students to finish the levels.

To assess the effectiveness of the CTF, we surveyed students at the beginning of the 10th week in order to determine how well the CTF helped students learn about cloud security issues related to reducing privileges via IAM. The questions included in the survey included:

- Q1: Rate the exercise for helping to understand least-privilege access control issues in the cloud.
- Q2: Rate the exercise for helping to develop skills for applying the principle of least-privilege access control in the cloud.
- Q3: Rate the scaffolding of levels in the exercise for helping to quickly learn about least-privilege access control in the cloud.

Of the 60 students in the class, 36 responded to the survey. Table II shows the results. As the table shows, students felt that the lecture material and CTF exercises were both helpful for learning Cloud IAM and the principle of least privilege, while students found the scaffolded levels and instructions very helpful as a learning aid, validating our design.

V. CONCLUSION

Using Identity and Access Management to implement the principle of least-privilege is imperative in cloud deployments. Unfortunately, it can be difficult for practitioners to understand and apply it. Towards this end, we have designed and implemented a Least Privilege CTF, a set of exercises focused on applying the principle of least privilege in Google Cloud IAM. The CTF is publicly available [20] and results from an initial offering of the CTF in a cloud course shows promising results.

ACKNOWLEDGMENTS

This material is supported by the National Science Foundation under Grant No. 1821841. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] S. A. Almulla and C. Y. Yeun, "Cloud computing security management," in *2010 Second International Conference on Engineering System Management and Applications*, 2010, pp. 1–7.

[2] Ermetic, "Top identity and data access risks idc cloud security survey highlights," Ermetic, IDC, Tech. Rep., 2020. [Online]. Available: https://l.ermetic.com/wp-idc-survey-results?utm_campaign=IDC%20Survey%20Highlights&utm_source=Press%20release

[3] G. Fawkes, "Report: Travel Reservations Platform Leaks US Government Personnel Data," vpnMentor, Sep. 02, 2020. [Online]. [Online]. Available: https://www.vpnmentor.com/blog/us-travel-military-leak/

[4] Flexera, "2020 Flexera State of the Cloud Report," Flexera, Tech. Rep., 2020. [Online]. Available: https://www.flexera.com/about-us/press-center/flexera-releases-2020-state-of-the-cloud-report.html

[5] T. Foremski, "Devsecops report: Cloud it complexity creates 'immutable' security issues," 2020. [Online]. Available: https://www.zdnet.com/article/report-finds-cloud-it-complexity-creates-immutable-security-issues/

[6] Google Cloud Platform, "GCP Free Tier," 2021, https://cloud.google.com/free.

[7] ——, "Identity and Access Management," 2021. [Online]. Available: https://cloud.google.com/iam/docs/overview

[8] ——, "Understanding Roles," 2021. [Online]. Available: https://cloud.google.com/iam/docs/understanding-roles

[9] IBM Security, "Cost of Data Breach Report 2020 (Highlights)," IBM, Tech. Rep., 2020. [Online]. Available: https://www.ibm.com/security/digital-assets/cost-data-breach-report/{#}/

[10] S. Media, "Cloud Infrastructure IAM Lessons from the Capital One Breach," https://www.scmagazine.com/home/opinion/executive-insight/cloud-infrastructure-iam-lessons-from-the-capital-one-breach/, Dec. 03, 2019. [Online].

[11] L. M. Newman, "Twitter Insiders Allegedly Spied for Saudi Arabia ," WIRED, Nov. 06, 2019. [Onine]. [Online]. Available: https://www.wired.com/story/twitter-insiders-saudi-arabia-spy/

[12] OWASP, "OWASP ServerlessGoat: A Serverless Application Demonstrating Common Serverless Security FLaws," 2021. [Online]. Available: https://github.com/OWASP/Serverless-Goat

[13] Rhino Security Labs, "CloudGoat: The Vulnerable-by-Design AWS Environment," 2021. [Online]. Available: https://github.com/RhinoSecurityLabs/cloudgoat

[14] J. H. Saltzer, "Protection and the Control of Information Sharing in Multics," *Communications of the ACM*, vol. 17, no. 7, pp. 388–402, jul 1974. [Online]. Available: http://portal.acm.org/citation.cfm?doid=361011.361067

[15] Scott Piper, "flAWS," 2021. [Online]. Available: http://flaws.cloud

[16] ——, "flAWS2," 2021. [Online]. Available: http://flaws2.cloud

[17] P. Sharrma, "The state of devsecops report - spring 2020," Accurics, Tech. Rep., 2020. [Online]. Available: https://start.accurics.com/

[18] N. Springer, "Thunder CTF," 2021. [Online]. Available: https://thunder-ctf.cloud/

[19] N. Springer and W.-C. Feng, "Thunder CTF: Learning Cloud Security on a Dime," 2021. [Online]. Available: https://arxiv.org/abs/2107.12566

[20] W. Wu, "Least-Privilege CTF," 2021. [Online]. Available: https://thunder-ctf.cloud/leastprivilege