Why Does Your Data Leak?

Uncovering the Data Leakage in Cloud from Mobile Apps

Chaoshun Zuo, Zhiqiang Lin, and Yinqian Zhang

Department of Computer Science and Engineering
The Ohio State University

IEEE S&P 2019

Recent News Headlines about Cloud **Data Leakage**

Recent News Headlines about Cloud **Data Leakage**

Verizon data of 6 million users leaked online

by Selena Larson @selenalarson

July 12, 2017: 4:14 PM ET

Cyber-Safe

Pentagon exposed some of its data on Amazon server

by Selena Larson @selenalarson

November 17, 2017: 12:03 PM ET

Recent News Headlines about Cloud **Data Leakage**

Cyber-Safe

Pentagon exposed some of its data on Amazon server

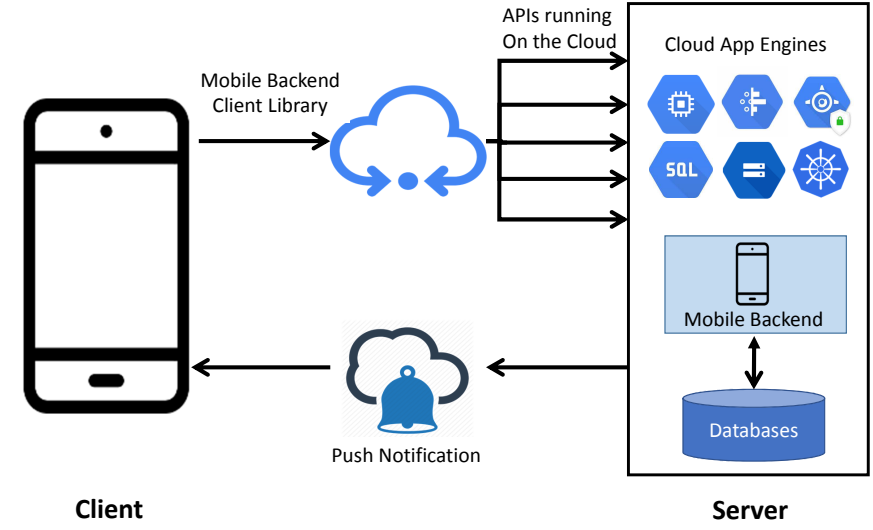
by Selena Larson @selenalarson

November 17, 2017: 12:03 PM ET

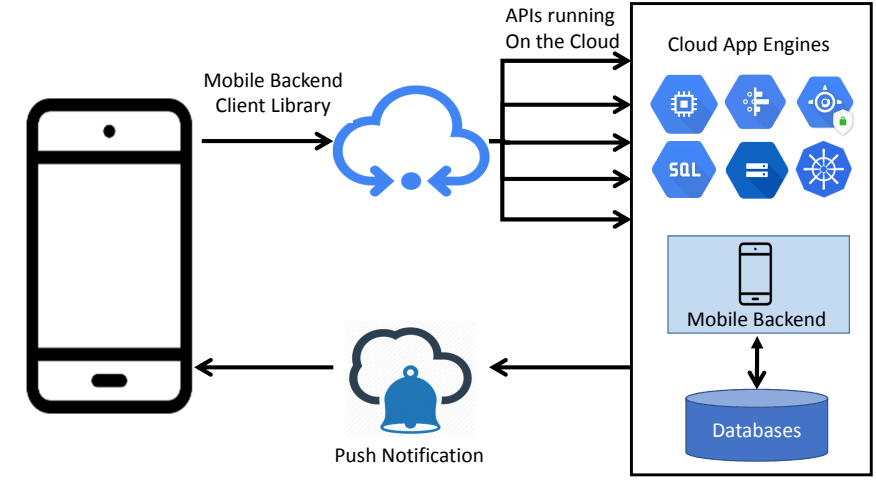
Recommend 9

Recent News Headlines about Cloud **Data Leakage**

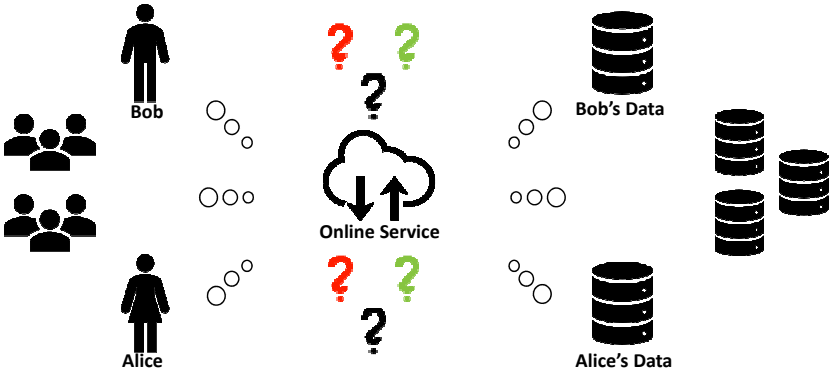
The Mobile Backend as a Service (mBaaS) Clouds



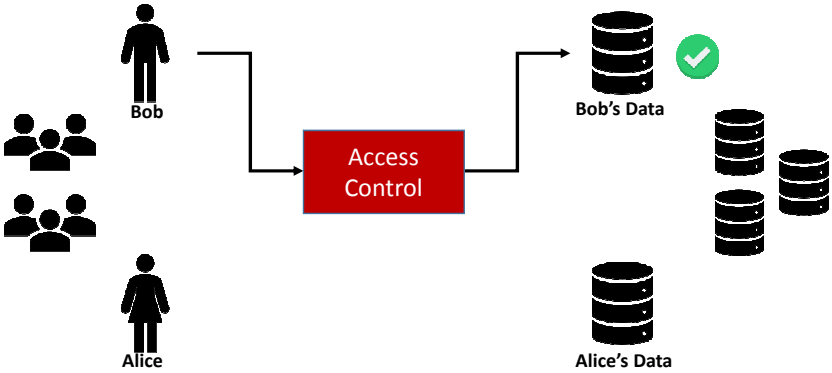
The Mobile Backend as a Service (mBaaS) Clouds



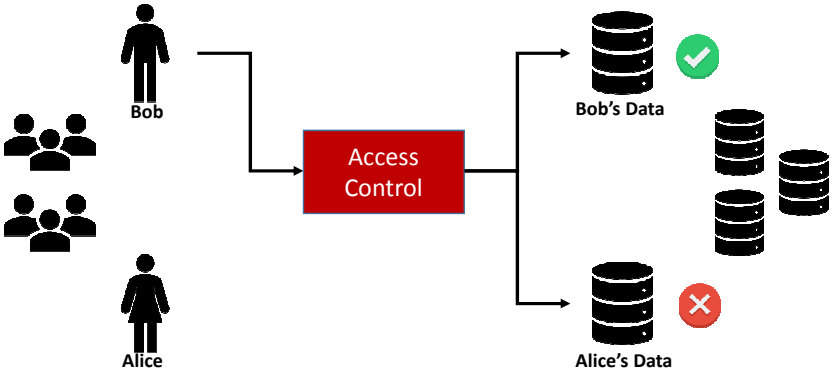
Data Leakage is Essentially an Access Control Problem



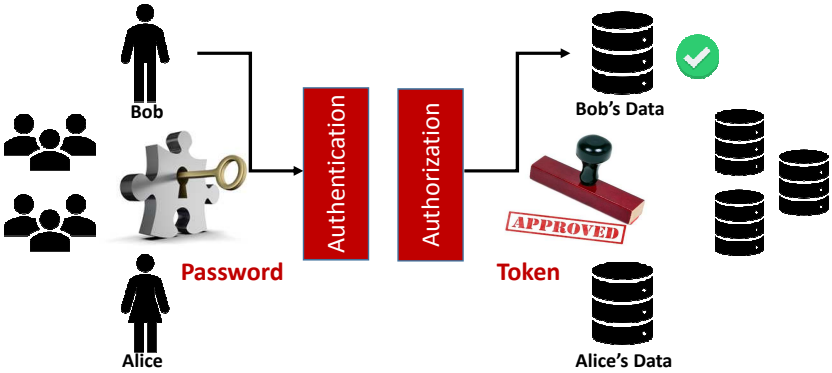
Data Leakage is Essentially an Access Control Problem



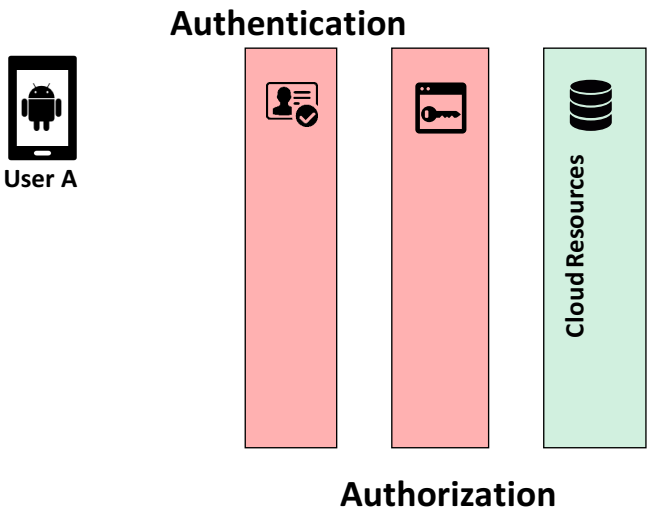
Data Leakage is Essentially an Access Control Problem



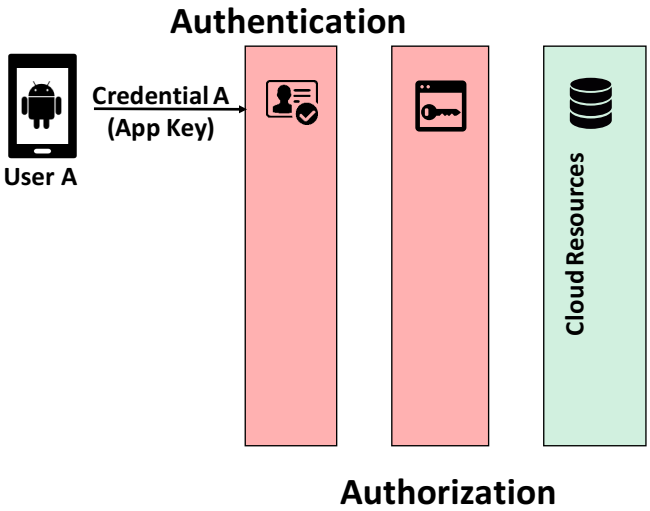
Data Leakage is Essentially an Access Control Problem



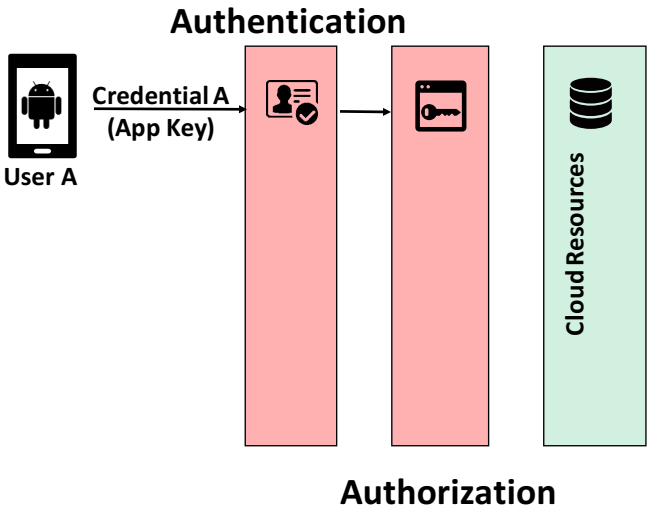
How Do Mobile Apps and mBaaS Cloud Communicate



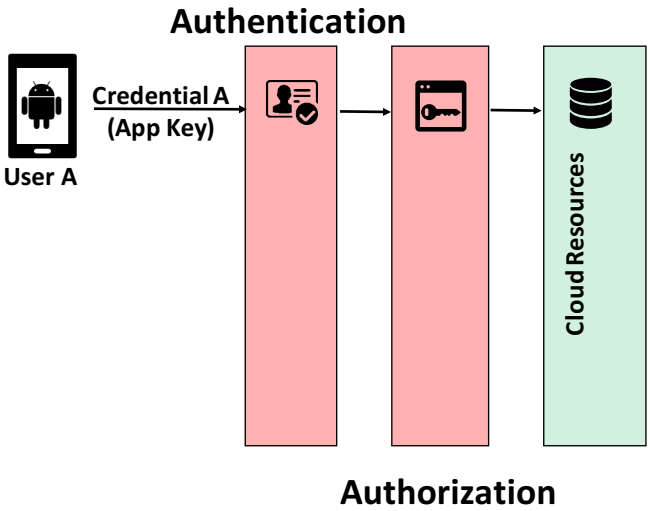
How Do Mobile Apps and mBaaS Cloud Communicate



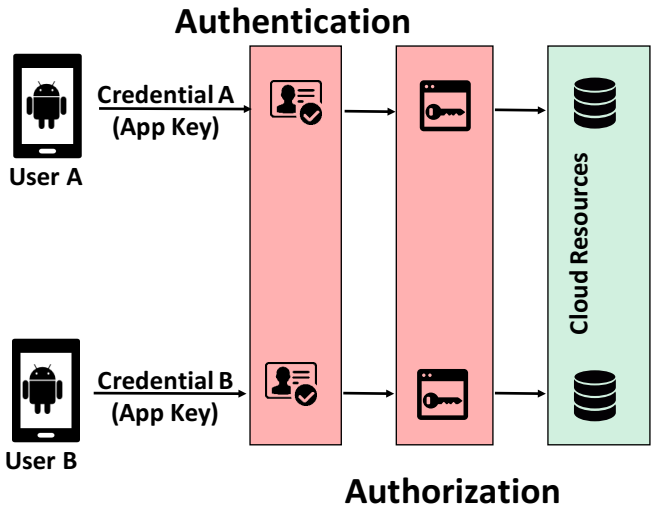
How Do Mobile Apps and mBaaS Cloud Communicate



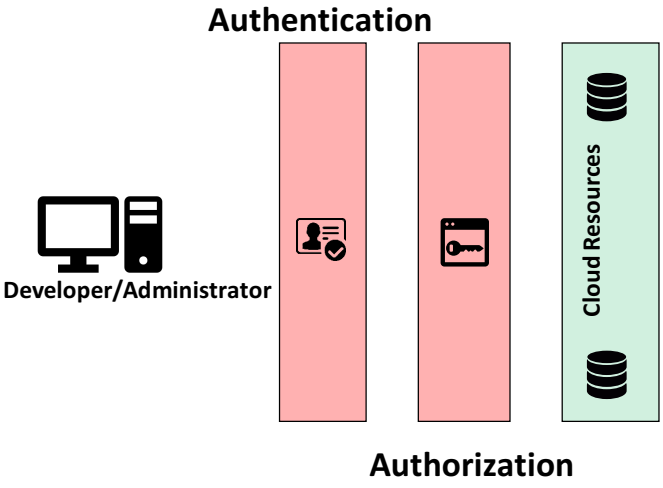
How Do Mobile Apps and mBaaS Cloud Communicate



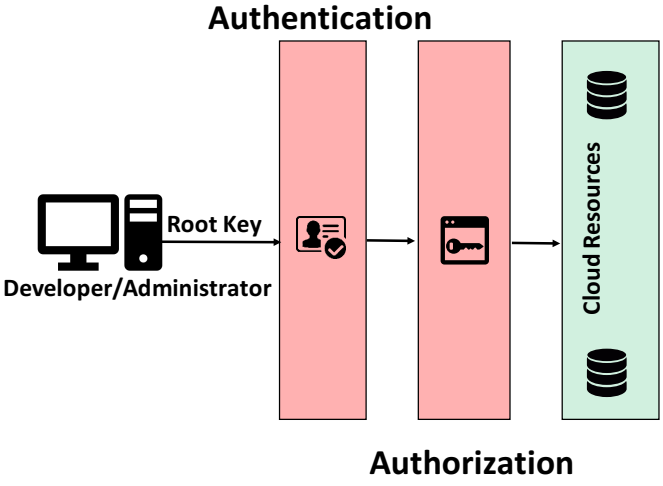
How Do Mobile Apps and mBaaS Cloud Communicate



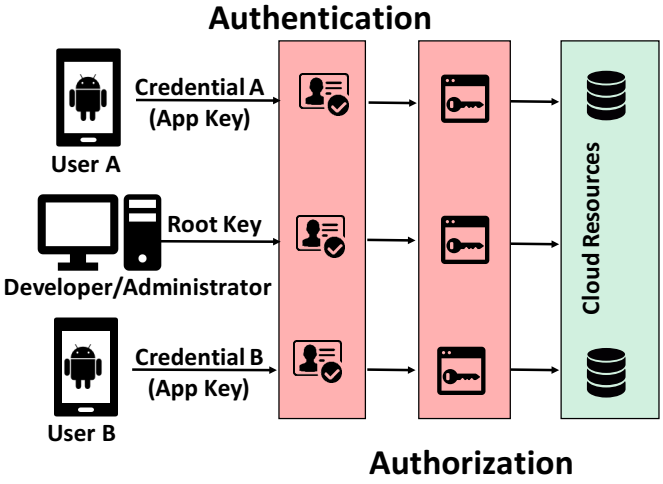
How Do Mobile Apps and mBaaS Cloud Communicate



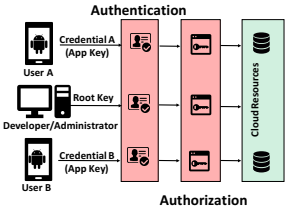
How Do Mobile Apps and mBaaS Cloud Communicate



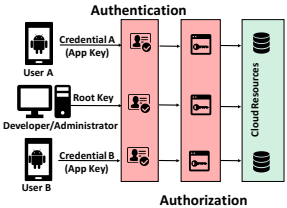
How Do Mobile Apps and mBaaS Cloud Communicate



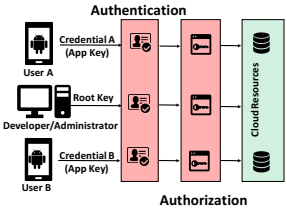
Our Discovery



Our Discovery



Our Discovery

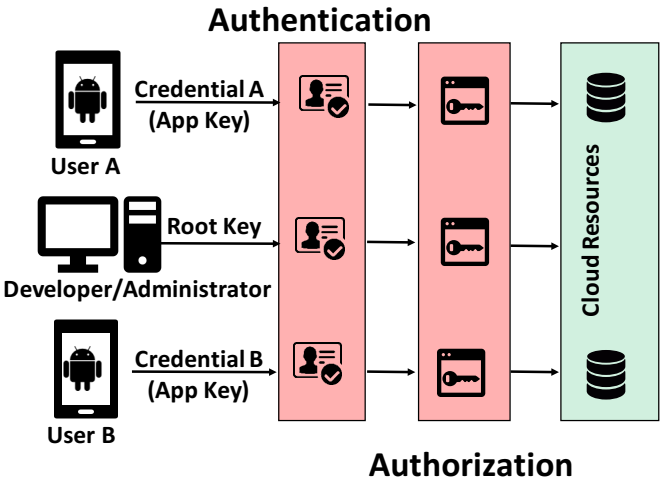


AWS Firebase Azure

The Root Causes of the Cloud Data Leakage

1. Misuse of Various Keys in **Authentication**
   - Microsoft Azure Storage
   - Microsoft Azure Notification Hubs
   - Amazon AWS
2. Misconfiguration of User Permissions in **Authorization**
   - Google Firebase
   - Amazon AWS

Misuse of Various Keys in Authentication



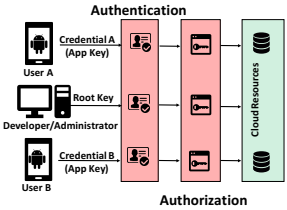
Misuse of Root Keys in Microsoft Azure

Service	Key Type	Example
Azure Storage	Account Key	DefaultEndpointsProtocol=https; AccountName=*;AccountKey=*
	SAS	https://*.blob.core.windows.net/* ?sv=*&st=*&se=*&sr=b& sp=rw&sip=*&spr=https&sig=*
Notification Hub	Listening Key	Endpoint=sb://*.servicebus.windows.net/; SharedAccessKeyName= DefaultListenSharedAccessSignature; SharedAccessKey=*
	Full Access Key	Endpoint=sb://*.servicebus.windows.net/; SharedAccessKeyName= DefaultFullSharedAccessSignature; SharedAccessKey=*

Misuse of Root Keys in Microsoft Azure

Service	Key Type	Example
Azure Storage	Account Key	DefaultEndpointsProtocol=https; AccountName=*;AccountKey=*
	SAS	https://*.blob.core.windows.net/* ?sv=*&st=*&se=*&sr=b& sp=rw&sip=*&spr=https&sig=*
Notification Hub	Listening Key	Endpoint=sb://*.servicebus.windows.net/; SharedAccessKeyName= DefaultListenSharedAccessSignature; SharedAccessKey=*
	Full Access Key	Endpoint=sb://*.servicebus.windows.net/; SharedAccessKeyName= DefaultFullSharedAccessSignature; SharedAccessKey=*

Our Discovery

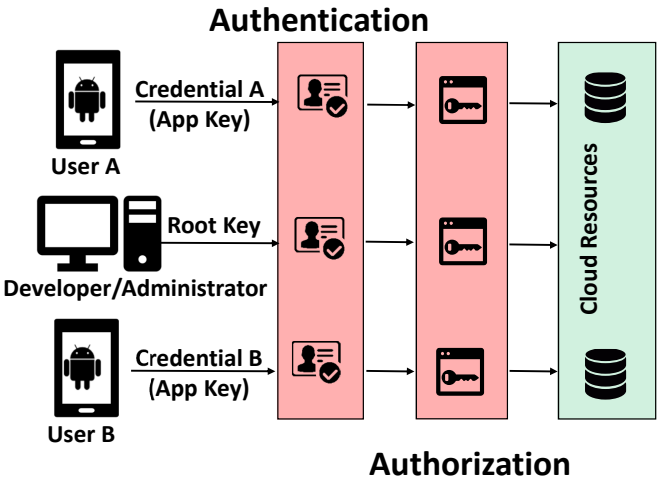


AWS Firebase Azure

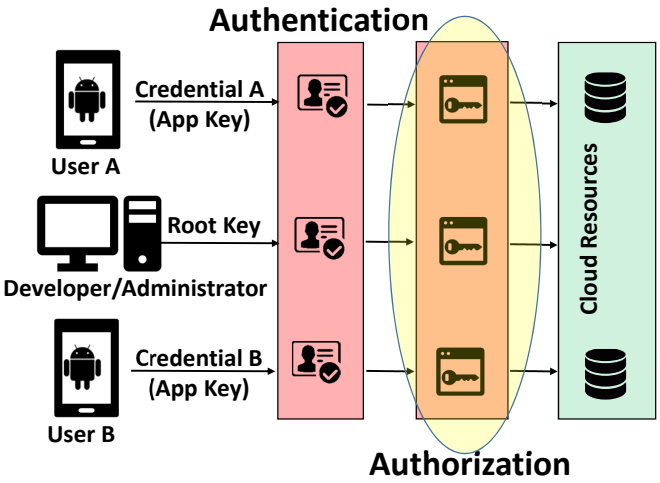
The Root Causes of the Cloud Data Leakage

1. ~~Misuse of Various Keys in **Authentication**~~
   - ~~Microsoft Azure Storage~~
   - ~~Microsoft Azure Notification Hubs~~
   - ~~Amazon AWS~~
2. Misconfiguration of User Permissions in **Authorization**
   - Google Firebase
   - Amazon AWS

Misconfiguration of User Permissions in **Authorization**



Misconfiguration of User Permissions in **Authorization**



Misconfiguration of User Permissions in Google Firebase

```
{
  "rules": {
    "users": {
      "$uid": {
        ".read": "$uid === auth.uid",
        ".write": "$uid === auth.uid"
      }
    }
  }
}
```

Figure: A Correct Firebase Authorization Rule

Misconfiguration of User Permissions in Google Firebase

```
{
  "rules": {
    "users": {
      "$uid": {
        ".read": "$uid === auth.uid",
        ".write": "$uid === auth.uid"
      }
    }
  }
}
```

Figure: A Correct Firebase Authorization Rule

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

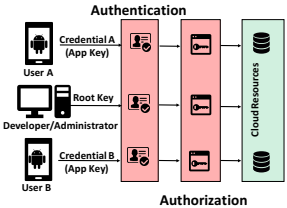
(a)

```
{
  "rules": {
    ".read": "auth != null",
    ".write": "auth != null"
  }
}
```

(b)

Figure: Two Misconfigured Firebase Authorization Rules

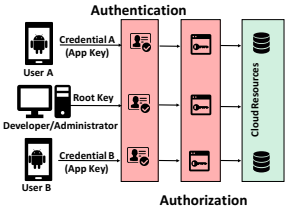
Problem Statement

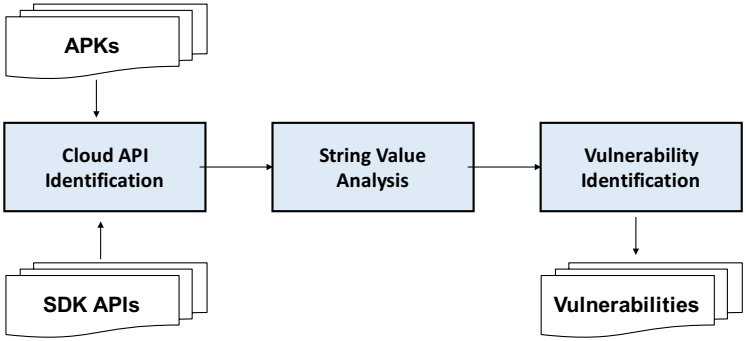


How to automatically detect the cloud leakage at scale

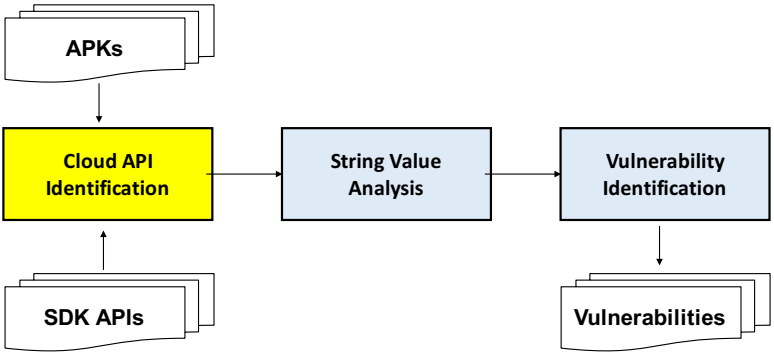
1. How to systematically identify various keys used by mobile apps (**Cloud APIs**)
2. How to identify the relevant key strings that are used by mobile apps (**String Analysis**)
3. How to design an obfuscation-resilient approach to identify cloud APIs and key strings of our interest (**Obfuscation-Resilient**)
4. How to determine the vulnerability without leaking sensitive information in the cloud (**Vulnerability Confirmation**)

Introducing LeakScope





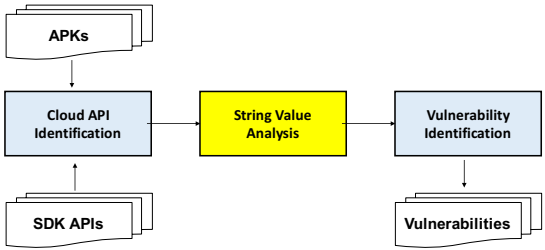
Cloud API Identification



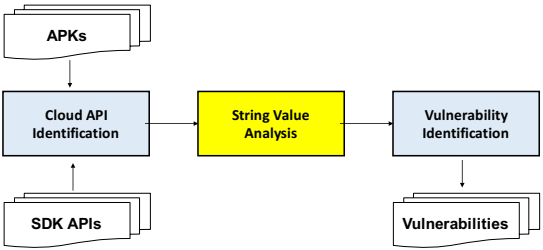
Cloud API Identification

Cloud Service	APIs	Definition	Indexes of The String Parameters of Our Interest
AWS	1*	TransferUtility: TransferObserver downloadUpload(String, String, File)	0
	2*	AmazonS3Client: void S3objectAccess(String, String, ...)	0
	3	CognitoCredentialsProvider: void <init>(String,String,String,String,...)	1
	4	BasicAWSCredentials: void <init>(String,String)	0,1
Azure	5	MobileServiceClient: void <init>(String,Context)	0
	6	MobileServiceClient: void <init>(String,String,Context)	0,1
	7	NotificationHub: void <init>(String,String,Context)	1
	8	CloudStorageAccount: CloudStorageAccount parse(String)	0
Firebase	9	FirebaseOptions: void <init>(String,String,String,String,String,String,String)	0,1,2,5
	10	FirebaseOptions: void <init>(String,String,String,String,String,String)	0,1,2,5

String Value Analysis

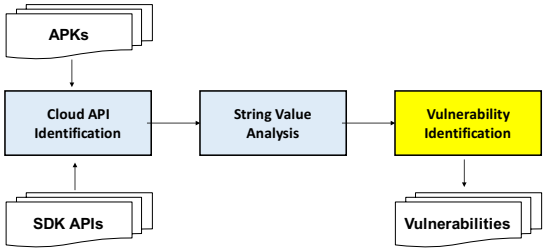


String Value Analysis

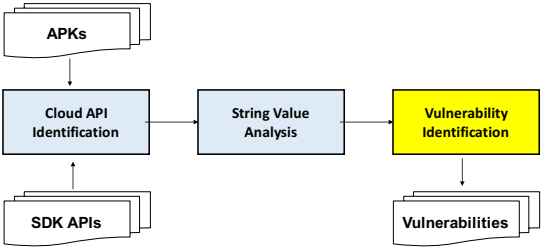


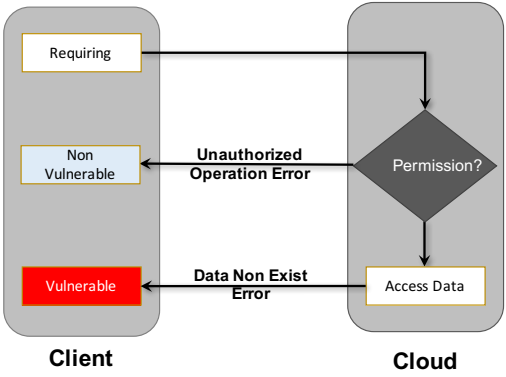

```
1 package com.appname
2 public class ImagesHelper {
3   private final String storageAccountKey;
4   private final String storageAccountName;
5
6   private ImagesHelper(Context arg3) {
7     int v0 = 2131099713;
8     int v1 = 2131099712;
9     this.storageAccountName =
10        this.getResources().getString(v0);
11    this.storageAccountKey =
12        this.getResources().getString(v1);
13  }
14
15  public void downloadImages(Callback arg5,
16        OnDownloadImagesUpdateListener arg6) {
17    StringBuilder v0 = new StringBuilder();
18    v0.append("DefaultEndpointsProtocol=http;AccountName=");
19    v0.append(this.storageAccountName);
20    v0.append(";AccountKey=");
21    v0.append(this.storageAccountKey);
22    String v1 = v0.toString();
23    CloudStorageAccount v7 = CloudStorageAccount.parse(v1);
24 ...
```

Vulnerability Identification



Vulnerability Identification





Distributions of the Testing Apps

	Total		Non-Obfuscated		Obfuscated	
	#Apps	%	#Apps	%	#Apps	%
w/ Cloud API	107,081	-	85,357	79.71	21,724	20.29
w/ AWS only	4,799	4.48	4,548	5.33	251	1.16
w/ Azure only	899	0.84	720	0.84	179	0.82
w/ Firebase only	99,186	92.63	78,475	91.94	20,711	95.34
w/ AWS & Azure	3	0.00	2	0.00	1	0.00
w/ AWS & Firebase	1,973	1.84	1,427	1.67	546	2.51
w/ Azure & Firebase	210	0.20	174	0.20	36	0.17
w/ Three Services	11	0.01	11	0.01	0	0.00

Distributions of the Testing Apps

	Total		Non-Obfuscated		Obfuscated	
	#Apps	%	#Apps	%	#Apps	%
w/ Cloud API	107,081	-	85,357	79.71	21,724	20.29
w/ AWS only	4,799	4.48	4,548	5.33	251	1.16
w/ Azure only	899	0.84	720	0.84	179	0.82
w/ Firebase only	99,186	92.63	78,475	91.94	20,711	95.34
w/ AWS & Azure	3	0.00	2	0.00	1	0.00
w/ AWS & Firebase	1,973	1.84	1,427	1.67	546	2.51
w/ Azure & Firebase	210	0.20	174	0.20	36	0.17
w/ Three Services	11	0.01	11	0.01	0	0.00

Result of Cloud API Identification & String Value Analysis

			Non-Obfuscated				Obfuscated			
	String Parameter Name	APIs	#API-Call	#APP	#Resolved Str.	%	#API-Call	#APP	#Resolved Str.	%
AWS	bucketName	1*	2,460	1,229	2,190	89.02	398	1,229	321	80.65
	bucketName	2*	2,069	1,703	2,045	98.84	444	439	442	99.55
	identityPoolId	3	3,458	3,458	3,315	95.86	291	291	266	91.41
	accessKey	4	3,280	1,769	2,650	80.79	277	203	199	71.84
	secretKey	4	3,280	1,769	2,646	80.67	277	203	197	71.12
Azure	appURL	5	185	39	185	100.00	11	4	11	100.00
	appURL	6	824	316	817	99.15	32	21	32	100.00
	appKey	6	824	316	809	98.18	32	21	31	96.88
	connectionString	7	700	513	643	91.86	207	189	200	96.62
	connectionString	8	345	97	303	87.83	29	21	22	75.86
Firebase	google_app_id	9	2,378	1,228	2,222	93.44	935	908	934	99.89
	google_api_key	9	2,378	1,228	2,230	93.78	935	908	927	99.14
	firebase_database_url	9	2,378	1,228	2,039	85.74	935	908	882	94.33
	google_storage_bucket	9	2,378	1,228	2,050	86.21	935	908	882	94.33
	google_app_id	10	154,664	78,859	143,735	92.93	20,723	20,385	20,657	99.68
	google_api_key	10	154,664	78,859	137,589	88.96	20,723	20,385	20,199	97.47
	firebase_database_url	10	154,664	78,859	118,786	76.80	20,723	20,385	18,077	87.23
	google_storage_bucket	10	154,664	78,859	119,606	77.33	20,723	20,385	18,041	87.06

Statistics of The Detected Vulnerabilities

	The Root Cause	Non-Obfuscated #Apps	Non-Obfuscated %	Obfuscated #Apps	Obfuscated %
Azure	Account Key Misuse	85	9.37	18	8.33
	Full Access Key Misuse	101	11.14	12	5.56
AWS	Root key Misuse	477	7.97	92	11.53
	"Open" S3 Storage	916	15.30	195	24.44
Firebase	"Open" Database	5,166	6.45	1,214	5.70
	No Permission Check	6,855	8.56	2,168	10.18

Statistics of The Detected Vulnerabilities

	The Root Cause	Non-Obfuscated #Apps	Non-Obfuscated %	Obfuscated #Apps	Obfuscated %
Azure	Account Key Misuse	85	9.37	18	8.33
	Full Access Key Misuse	101	11.14	12	5.56
AWS	Root key Misuse	477	7.97	92	11.53
	"Open" S3 Storage	916	15.30	195	24.44
Firebase	"Open" Database	5,166	6.45	1,214	5.70
	No Permission Check	6,855	8.56	2,168	10.18

Statistics of The Detected Vulnerabilities

#Downloads	# Non-Vulnerable Apps				# Vulnerable Apps			
	Azure	AWS	Firebase	Obfuscated%	Azure	AWS	Firebase	Obfuscated%
$1,000,000,000 - 5,000,000,000$	0	0	1	100.00	0	0	0	0.00
$500,000,000 - 1,000,000,000$	0	0	3	66.67	0	0	0	0.00
$100,000,000 - 500,000,000$	0	1	35	58.33	0	1	9	50.00
$50,000,000 - 100,000,000$	0	4	67	45.07	0	2	12	71.43
$10,000,000 - 50,000,000$	2	35	480	47.78	1	4	75	50.00
$5,000,000 - 10,000,000$	3	32	467	37.85	1	6	66	38.36
$1,000,000 - 5,000,000$	16	136	2,405	32.15	2	21	369	30.10
$500,000 - 1,000,000$	10	105	1,823	29.36	1	29	260	28.28
$100,000 - 500,000$	65	356	6,987	26.01	14	66	1,026	26.13
$50,000 - 100,000$	42	249	4,608	25.52	11	50	695	25.13
$10,000 - 50,000$	167	679	12,868	24.85	21	174	1,862	21.88
$5,000 - 10,000$	82	369	6,090	24.05	11	100	770	23.61
$1,000 - 5,000$	272	976	15,920	21.42	40	248	1,977	20.66
$0 - 1,000$	464	3,844	49,626	15.92	111	754	6,402	20.30

Table: The Number of Apps that Have Used the Cloud APIs in Each of The Accumulated Download Category.

Statistics of The Detected Vulnerabilities

#Downloads	# Non-Vulnerable Apps				# Vulnerable Apps			
	Azure	AWS	Firebase	Obfuscated%	Azure	AWS	Firebase	Obfuscated%
$1,000,000,000 - 5,000,000,000$	0	0	1	100.00	0	0	0	0.00
$500,000,000 - 1,000,000,000$	0	0	3	66.67	0	0	0	0.00
$100,000,000 - 500,000,000$	0	1	35	58.33	0	1	9	50.00
$50,000,000 - 100,000,000$	0	4	67	45.07	0	2	12	71.43
$10,000,000 - 50,000,000$	2	35	480	47.78	1	4	75	50.00
$5,000,000 - 10,000,000$	3	32	467	37.85	1	6	66	38.36
$1,000,000 - 5,000,000$	16	136	2,405	32.15	2	21	369	30.10
$500,000 - 1,000,000$	10	105	1,823	29.36	1	29	260	28.28
$100,000 - 500,000$	65	356	6,987	26.01	14	66	1,026	26.13
$50,000 - 100,000$	42	249	4,608	25.52	11	50	695	25.13
$10,000 - 50,000$	167	679	12,868	24.85	21	174	1,862	21.88
$5,000 - 10,000$	82	369	6,090	24.05	11	100	770	23.61
$1,000 - 5,000$	272	976	15,920	21.42	40	248	1,977	20.66
$0 - 1,000$	464	3,844	49,626	15.92	111	754	6,402	20.30

Table: The Number of Apps that Have Used the Cloud APIs in Each of The Accumulated Download Category.

Engaging with the Cloud Providers

Disclosed all the vulnerabilities we have identified. Cloud providers further notified the app developers.

1. **Microsoft** immediately corrected the wrong documentation
2. **Google** plans to provide more user-friendly SDKs when configuring user permissions in authorization.
3. **Amazon** added new permission checks with its S3 storage in November 2017 (two weeks before we disclosed our details to them)

Disclaimer on the use of account key after we disclosed the vulnerability

Disclaimer on the use of account key Browse files

master (#65)

seguler committed on Dec 21, 2017 1 parent 191f088 commit d90c3a49312e77c2cc911c8f55a37be9947454e4

Showing 1 changed file with 7 additions and 0 deletions. Unified Split

7 microsoft-azure-storage-samples/src/com/microsoft/azure/storage/samples/MainActivity.java View

```
@@ -24,6 +24,13 @@
 24  24      * MODIFY THIS!
 25  25      *
 26  26      * Stores the storage connection string.
+   27      * Only use Shared Key authentication (Account Key) for testing purposes!
+   28      * Your account name and account key, which give full read/write access to the associated Storage account,
+   29      * will be distributed to every person that downloads your app.
+   30      * This is not a good practice as you risk having your key compromised by untrusted clients.
+   31      * Please consult following documents to understand and use Shared Access Signatures instead.
+   32      * https://docs.microsoft.com/en-us/rest/api/storageservices/delegating-access-with-a-shared-access-signature
+   33      * https://docs.microsoft.com/en-us/azure/storage/common/storage-dotnet-shared-access-signature-part-1
 27  34      */
 28  35     public static final String storageConnectionString = "DefaultEndpointsProtocol=https;"
 29  36             + "AccountName=[MY_ACCOUNT_NAME];"
```

Google's Update

1. The big additions on Google's side are tools for **local emulation** and **writing tests** against the database products including their security rules, which they expect to have a marked improvement on the ability of customers to test and validate security rules.
2. Additionally, they **have alerting for customers** (sent every few weeks) for anyone using the Realtime Database or Cloud Firestore with open rules.
3. They're exploring more options, but those are a start.

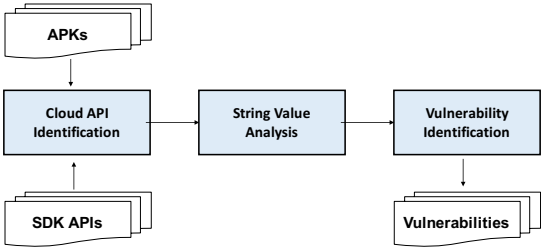
Related Work

1. **Protocol Reverse Engineering**. A large body of research focusing on protocol reverse engineering [Bed, MLK+06, CKW07, CS07, WMKK08, LJXZ08, MWKK09, CPKS09]
2. **Dynamic Analysis**. Monkey [mon17] automatically executes and randomly navigates an app. AppsPlayground [RCE13] and SMV-Hunter [SSG+14] more intelligent. A3E [AN13], a targeted exploration of mobile apps.
   DynoDroid [MTN13] instruments the Android framework and uses `adb` to monitor UI interaction and generate UI events.

Related Work

1. **Mobile App Vulnerability Discovery**.
   - Client Side: TaintDroid [EGC+10], PiOS [EKKV11], CHEX [LLW+12], SMV-Hunter [SSG+14].
   - Server Side: AutoForge [ZWWL16], SmartGen [ZL17], AuthScope [ZZL17].
2. **Misconfiguration Vulnerability Identification**: FIREMAN [YMS+06], ConfErr [KUC08], ConfAid [AF10], SPEX [XZH+13].

LeakScope



LeakScope

- A static analysis to identify server side data leakage vulnerabilities
- It performs cloud API identification, string value analysis to identify the vulnerabilities

Experimental Result w/ 100K apps

- 15,098 apps' cloud servers are vulnerable
- 200 Azure, 1,600 AWS, 13,200 Firebase
- Responsible disclosures were made to the cloud providers

Source code of LeakScopehas been made available at https://github.com/OSUSecLab/LeakScope

Future Works

1. We only scratched the tip of the iceberg of the security of cloud based backend – mBaaS cloud backend.
2. What about their backend software stack (e.g., VMs, operating systems, network stacks)?
3. What about other vulnerabilities (e.g., SQL injection, XSS, XXE)?

Future Works

1. We only scratched the tip of the iceberg of the security of cloud based backend – mBaaS cloud backend.
2. What about their backend software stack (e.g., VMs, operating systems, network stacks)?
3. What about other vulnerabilities (e.g., SQL injection, XSS, XXE)?

"**The Betrayal At Cloud City: An Empirical Analysis Of Cloud-Based Mobile Backends**". Omar Alrawi, Chaoshun Zuo, Ruian Duan, Ranjita Kasturi, Zhiqiang Lin, Brendan Saltaformaggio. In *USENIX Security*, August 2019.

Thank You

Why Does Your Data Leak?

Uncovering the Data Leakage in Cloud from Mobile Apps

Chaoshun Zuo, Zhiqiang Lin, and Yinqian Zhang

Department of Computer Science and Engineering
The Ohio State University

IEEE S&P 2019

References I

Mona Attariyan and Jason Flinn, *Automating configuration troubleshooting with dynamic information flow analysis*, Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (Vancouver, BC, Canada), OSDI'10, 2010, pp. 237–250.

Tanzirul Azim and Iulian Neamtiu, *Targeted and depth-first exploration for systematic testing of android apps*, Proceedings of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages & Applications (New York, NY, USA), OOPSLA '13, ACM, 2013, pp. 641–660.

Marshall Beddoe, *The protocol informatics project*, http://www.4tphi.net/~awalters/PI/PI.html.

Weidong Cui, Jayanthkumar Kannan, and Helen J. Wang, *Discoverer: Automatic protocol reverse engineering from network traces*, Proceedings of the 16th USENIX Security Symposium (Security'07) (Boston, MA), August 2007.

Juan Caballero, Pongsin Poosankam, Christian Kreibich, and Dawn Song, *Dispatcher: Enabling active botnet infiltration using automatic protocol reverse-engineering*, Proceedings of the 16th ACM Conference on Computer and and Communications Security (CCS'09) (Chicago, Illinois, USA), 2009, pp. 621–634.

Juan Caballero and Dawn Song, *Polyglot: Automatic extraction of protocol format using dynamic binary analysis*, Proceedings of the 14th ACM Conference on Computer and and Communications Security (CCS'07) (Alexandria, Virginia, USA), 2007, pp. 317–329.

W. Enck, P. Gilbert, B.G. Chun, L.P. Cox, J. Jung, P. McDaniel, and A.N. Sheth, *TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones*, OSDI, 2010.

M. Egele, C. Kruegel, E. Kirda, and G. Vigna, *Pios: Detecting privacy leaks in ios applications*, NDSS, 2011.

References II

Lorenzo Keller, Prasang Upadhyaya, and George Candea, *Conferr: A tool for assessing resilience to human configuration errors*, Dependable Systems and Networks With FTCS and DCC, 2008. DSN 2008. IEEE International Conference on, IEEE, 2008, pp. 157–166.

Zhiqiang Lin, Xuxian Jiang, Dongyan Xu, and Xiangyu Zhang, *Automatic protocol format reverse engineering through context-aware monitored execution*, Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08) (San Diego, CA), February 2008.

Long Lu, Zhichun Li, Zhenyu Wu, Wenke Lee, and Guofei Jiang, *Chex: statically vetting android apps for component hijacking vulnerabilities*, Proceedings of the 2012 ACM conference on Computer and communications security, ACM, 2012, pp. 229–240.

Justin Ma, Kirill Levchenko, Christian Kreibich, Stefan Savage, and Geoffrey M. Voelker, *Unexpected means of protocol inference*, Proceedings of the 6th ACM SIGCOMM on Internet measurement (IMC'06) (Rio de Janeriro, Brazil), ACM Press, 2006, pp. 313–326.

Ui/application exerciser monkey, `https://developer.android.com/tools/help/monkey.html`, 2017.

Aravind Machiry, Rohan Tahiliani, and Mayur Naik, *Dynodroid: An input generation system for android apps*, Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering, ACM, 2013, pp. 224–234.

Paolo Milani Comparetti, Gilbert Wondracek, Christopher Kruegel, and Engin Kirda, *Prospex: Protocol Specification Extraction*, IEEE Symposium on Security & Privacy (Oakland, CA), 2009, pp. 110–125.

Vaibhav Rastogi, Yan Chen, and William Enck, *Appsplayground: Automatic security analysis of smartphone applications*, Proceedings of the Third ACM Conference on Data and Application Security and Privacy (New York, NY, USA), CODASPY '13, ACM, 2013, pp. 209–220.

References III

David Sounthiraraj, Justin Sahs, Garrett Greenwood, Zhiqiang Lin, and Latifur Khan, *Smv-hunter: Large scale, automated detection of ssl/tls man-in-the-middle vulnerabilities in android apps*, Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS'14) (San Diego, CA), February 2014.

Gilbert Wondracek, Paolo Milani, Christopher Kruegel, and Engin Kirda, *Automatic network protocol analysis*, Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08) (San Diego, CA), February 2008.

Tianyin Xu, Jiaqi Zhang, Peng Huang, Jing Zheng, Tianwei Sheng, Ding Yuan, Yuanyuan Zhou, and Shankar Pasupathy, *Do not blame users for misconfigurations*, Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles (Farminton, Pennsylvania), SOSP '13, 2013, pp. 244–259.

Lihua Yuan, Jianning Mai, Zhendong Su, Hao Chen, Chen-Nee Chuah, and Prasant Mohapatra, *Fireman: A toolkit for firewall modeling and analysis*, Proceedings of the 2006 IEEE Symposium on Security and Privacy, SP'06, 2006, pp. 199–213.

Chaoshun Zuo and Zhiqiang Lin, *Exposing server urls of mobile apps with selective symbolic execution*, Proceedings of the 26th World Wide Web Conference (Perth, Australia), April 2017.

Chaoshun Zuo, Wubing Wang, Rui Wang, and Zhiqiang Lin, *Automatic forgery of cryptographically consistent messages to identify security vulnerabilities in mobile services*, Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS'16) (San Diego, CA), February 2016.

Chaoshun Zuo, Qingchuan Zhao, and Zhiqiang Lin, *Authscope: Towards automatic discovery of vulnerable authorizations in online services*, Proceedings of the 24th ACM Conference on Computer and Communications Security (CCS'17) (Dallas, TX), November 2017.

Backup

	App Name	App Description and Functionality	Obfuscated?	Data in Database/Storage	Privacy Sensitive?
AWS	A1	Sending messages with multiple fancy features	✓	User Photos	✓
	A2	Editing user photos with magical enhancements	✓	User Photos	✓
	A3	Editing user photos with featured specialties	✓	User Photos; Posted Pictures	✓
	A4	Allowing users to organize and upload photos	✗	User Uploaded Pictures	✓
	A5	Helping users in planning and booking trips	✓	User Photos	✓
	A6	A game app to build and design attractive hotels	✗	User Backups	✓
	A7	A game app to express revenges on game NPCs	✗	Premium Plug-ins	✗
Azure	A10	Helping users to start a diet and control weight	✓	User Photos; Posted Pictures	✓
	A11	Calculating and tracking calories for human health	✗	User Photos	✓
	A12	Showing fertility status from correspondent kits	✗	User Uploaded Pictures	✓
	A13	Helping users to easily play a popular game	✗	Configurations about the Game	✗
	A14	A real time translation tool, for calls, chats, etc.	✗	User Photos; Chat History	✓
	A15	Showing images of nations' commemorative coins	✓	Coins Images	✗
	A16	A convenient tool to take notes with rich content	✓	User Uploaded Pictures	✓
	A17	A convenient tool for users to schedule a taxi	✗	Driver Photos	✓
	A18	Allowing users to buy/renew general insurances	✗	Inspection Videos	✓
	A19	Providing accurate local weather forecast	✓	Device Info (IMEI, etc.)	✓
Firebase	A20	Editing and enhancing users photos and selfies	✗	User Info (①④); User Private Messages	✓
	A21	Allowing users to guess information about music	✓	Music Details	✗
	A22	Allowing users to sell and buy multiple products	✗	User Info (②④); Transactions	✓
	Photo Collage	Creating photo collage with personal photos	✓	User Info (②③)	✓
	A23	Helping users to translate and learn languages	✓	User Info (①); Quiz Data	✓
	A24	Editing user photos with effects for cartoon avatar	✗	User Info (①); User Pictures	✓
	A25	Help users to learn how to draw human bodies	✓	User Info (①②③); User Pictures	✓
	A26	An offline bible learning app with texts and audios	✗	User Info (①③④)	✓
	A27	Music platform for hiphop mixtapes and musics	✗	User Info (①②③); Play List	✓
	A28	Helping users to learn drawing different things	✓	User Info (①②③); User Pictures	✓

Symbol ① denotes the user name, ② the user ID, ③ the user email, and ④ the user token.