(In-)Security of Backend-as-a-Service

Siegfried Rasthofer (TU Darmstadt / CASED)
Steven Arzt (TU Darmstadt / CASED)
Robert Hahn (TU Darmstadt)
Max Kolhagen (TU Darmstadt)
Eric Bodden (Fraunhofer SIT / TU Darmstadt)

#Whoami

Siegfried Rasthofer

- 3rd year PhD-Student at TU Darmstadt
- Research interest in static-/dynamic code analyses
- AOSP exploits, App security vulnerabilities
- Talks at academic as well as industry conferences

Steven Arzt

- 3rd year PhD-Student at TU Darmstadt
- Maintainer of the Soot and FlowDroid frameworks
- Works on static program analysis
- Likes to look for vulnerabilities

TECHNISCHE UNIVERSITÄT DARMSTADT

SECURE SOFTWARE ENGINEERING GROUP

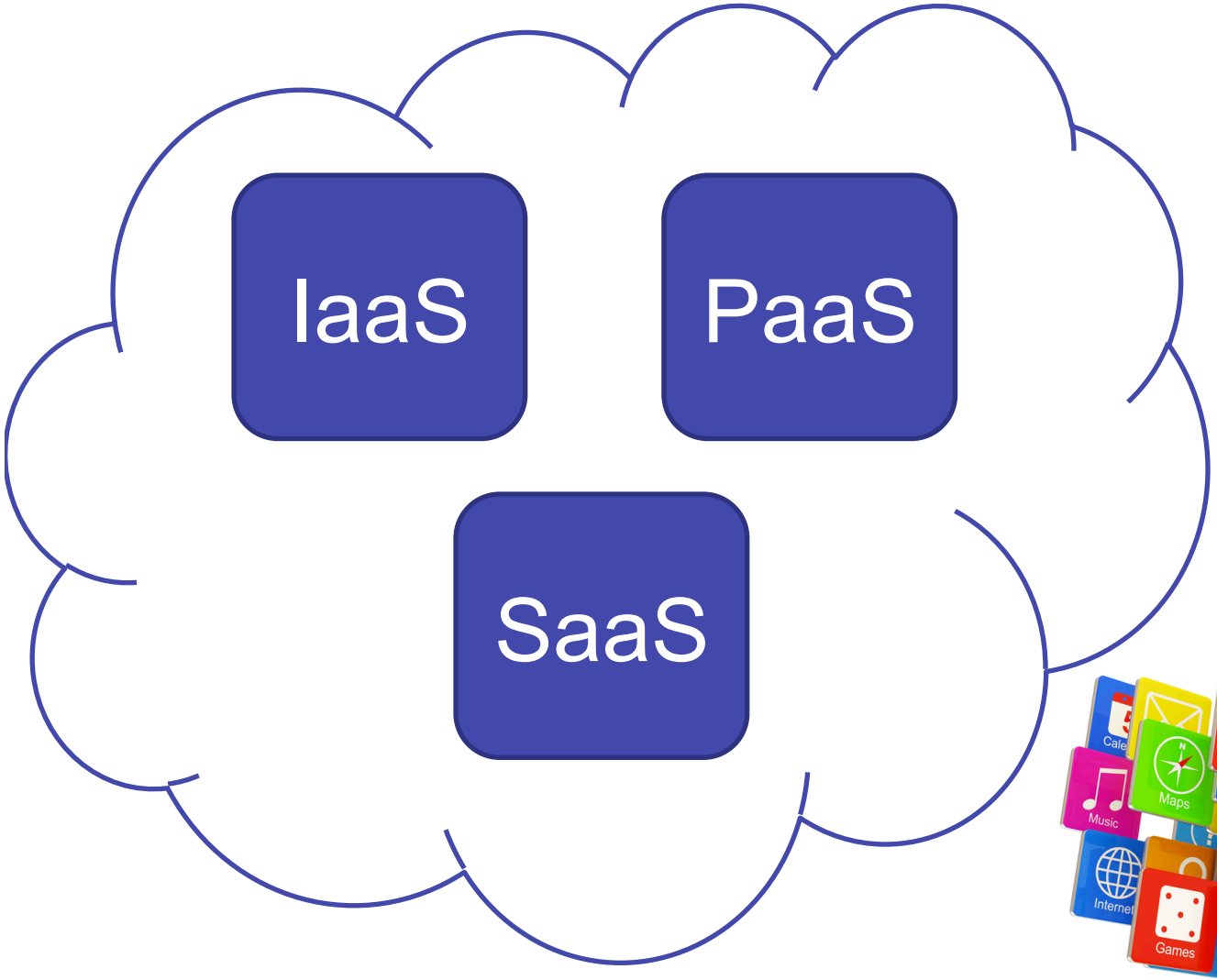
Fraunhofer SIT

Access to 56 Mio data...

Remote code execution...

Full VM access...

... with ease



??

TECHNISCHE UNIVERSITÄT DARMSTADT

SECURE SOFTWARE ENGINEERING GROUP

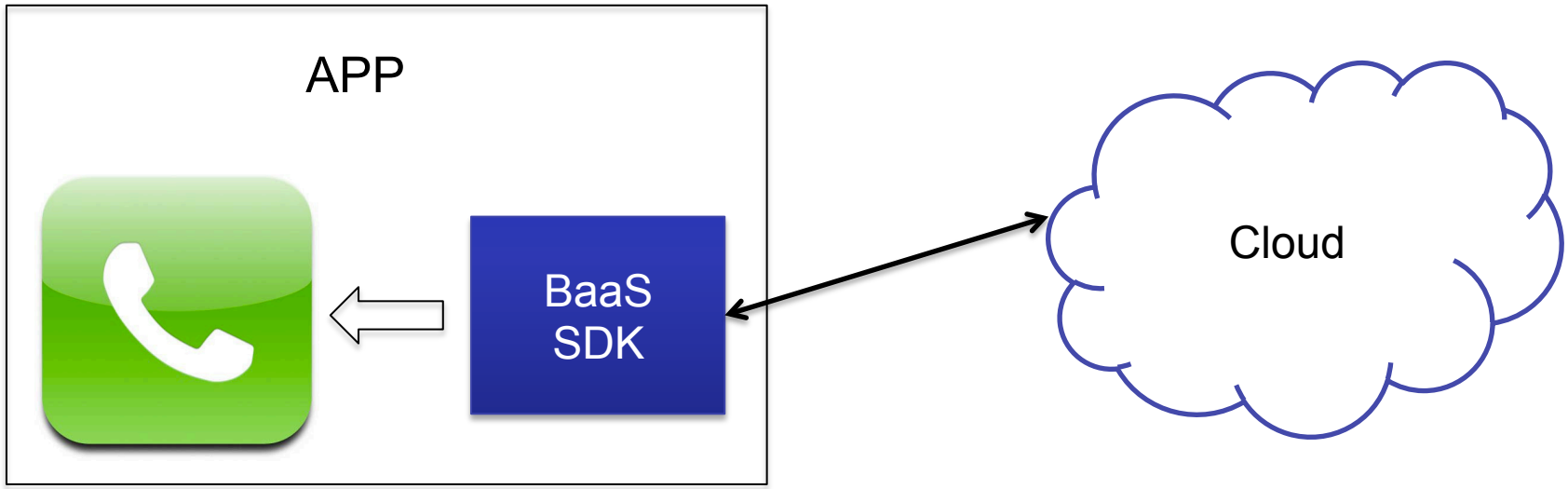
Fraunhofer SIT


BaaS

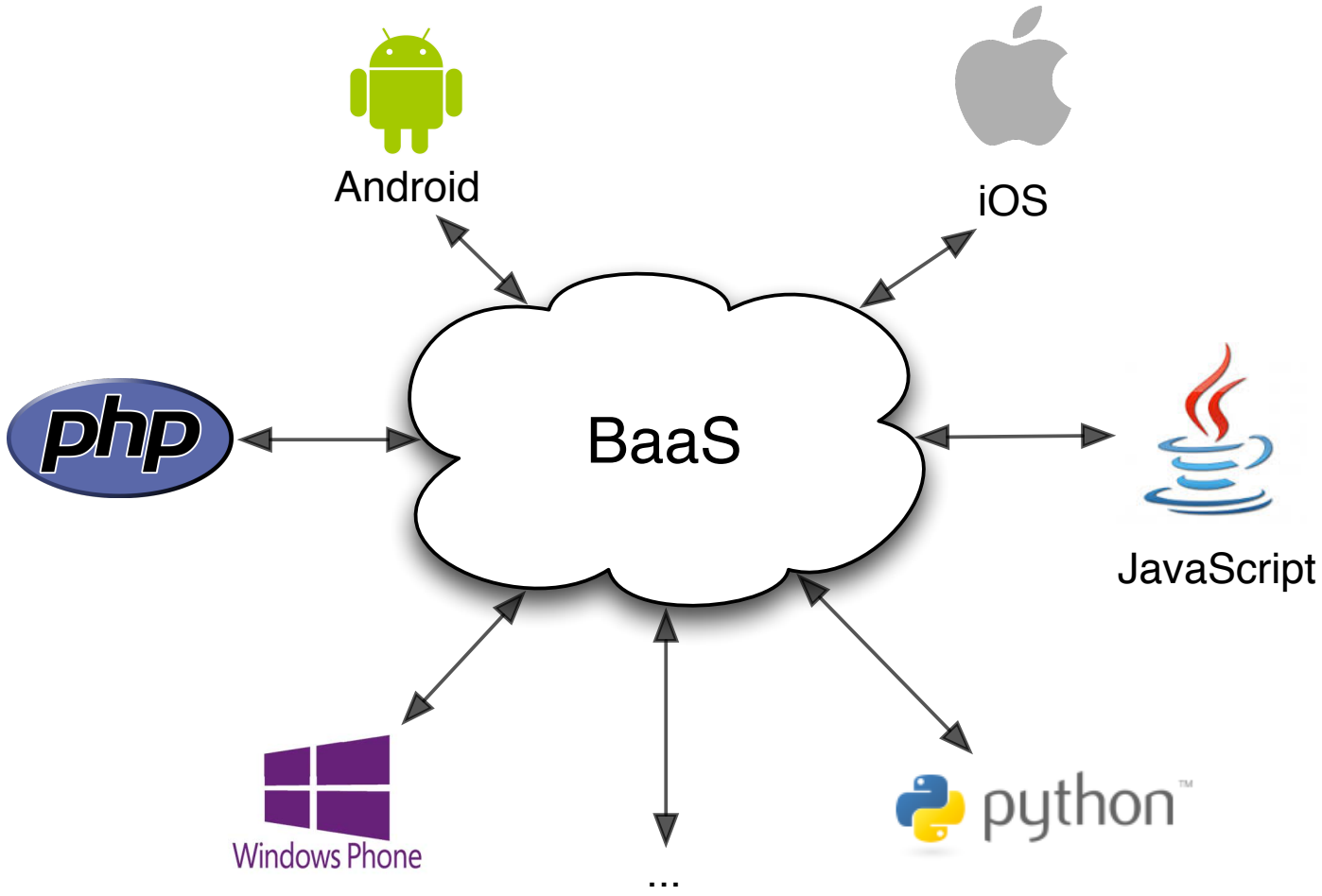
Agenda

- Introducing BaaS
- Security Analysis
- Findings
- Countermeasures
- The Wishlist
- Conclusion

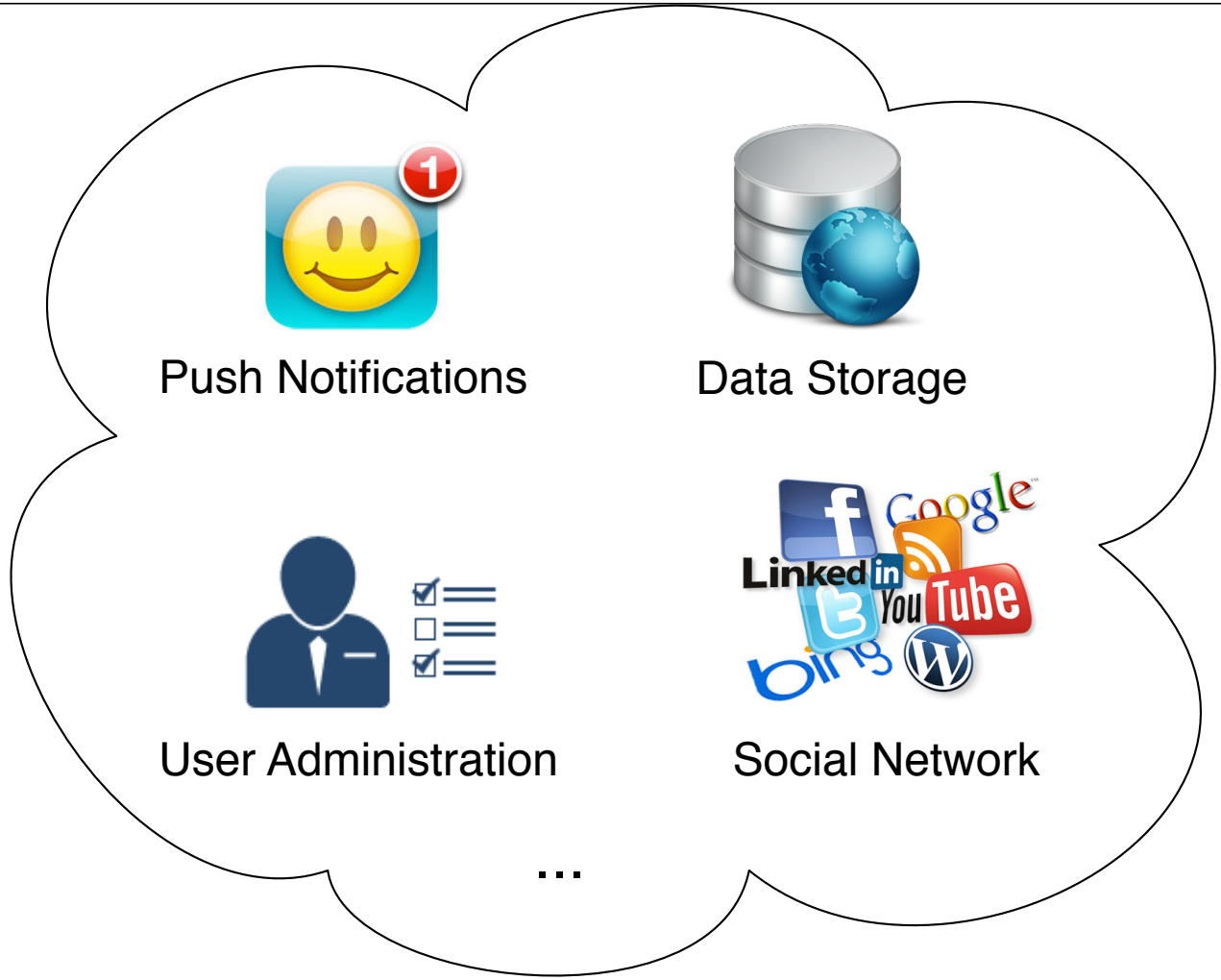
Backend-as-a-Service (1)



Backend-as-a-Service (2)



Backend-as-a-Service (3)




Parse
The Cloud Application Platform
cocoafish
StackMob
kinvey
Security?
cloudmine
amazon
web services™
BAASBOX
OPEN SOURCE BACKEND AS A SERVICE
mobeelizer


Parse
The Cloud Application Platform

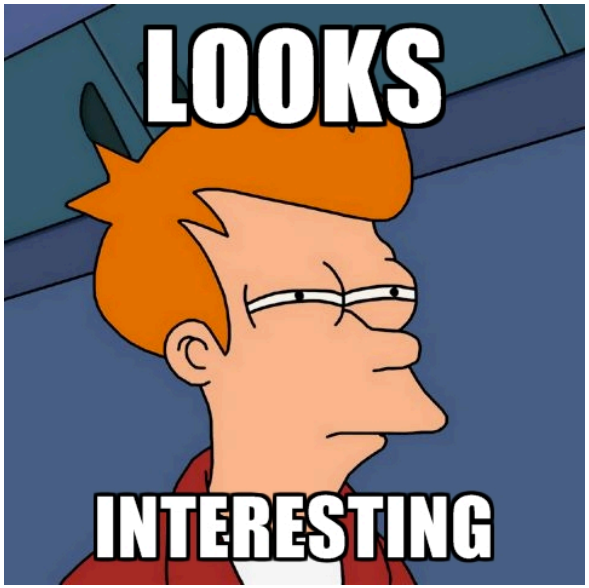

amazon
web services™

BaaS SDK

Amazon Tutorial – Files in S3 Buckets

DB connection

```
AmazonS3Client s3Client = new AmazonS3Client( new
BasicAWSCredentials("MY_ACCESS_KEY_ID", "MY_SECRET_KEY") );
```



*"When you access AWS programmatically, you use an access key to **verify your identity and the identity of your applications**. An access key consists of an access key ID and a secret access key.*

Anyone who has your access key has the same level of access to your AWS resources that you do."

Source: http://docs.aws.amazon.com/

BaaS SDK

Amazon Tutorial – Files in S3 Buckets

DB write

```
PutObjectRequest por = new PutObjectRequest( BUCKET_NAME, OBJECT_NAME,
File( filePath) );
s3Client.putObject( por );
```

DB read

```
GetObjectRequest request = new GetObjectRequest(BUCKET_NAME, OBJECT_NAME);
S3Object object = s3Client.getObject(request);
S3ObjectInputStream objectContent = object.getObjectContent();
IOUtils.copy(objectContent, new FileOutputStream("D://upload//test.jpg"));
```

BaaS SDK

Parse example

DB connection

```
Parse.initialize(this, "ApplicationID", "ClientKey");
```

DB write

```
ParseObject testObject = new ParseObject("TestTable");
testObject.put("foo", "bar");
testObject.saveInBackground();
```

DB read

```
final ParseQuery<ParseObject> userQuery = ParseQuery.getQuery("TestTable");
userQuery.findInBackground(new FindCallback<ParseObject>() {
@Override
public void done(List<ParseObject> parseObject, ParseException e) {
    if (e == null) {
        //foo : bar
    }
}
```

TECHNISCHE UNIVERSITÄT DARMSTADT

Let’s go for it

SECURITY ANALYSIS

TECHNISCHE UNIVERSITÄT DARMSTADT

SECURE SOFTWARE ENGINEERING GROUP

Fraunhofer SIT

```
initialize("AppID", "ClientKey")

getQuery("TestTable")

ParseObject("Users")

...
```

=

FUN???

Static

- Lot‘s of tools available:
  - Soot
  - Dex2jar + grep
  - Dare/Ded + grep
  - Smali + grep
  - ...

```
initialize(v1, dec("yicqco44"))

getQuery(v3)

ParseObject("Users")

...
```

=

FUN???

Static + Dynamic

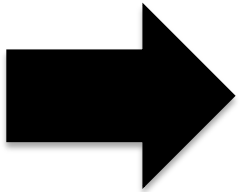
- Constant string propagation
- Function hooking
- Bytecode instrumentation
- HTTP(S) interception
- *Harvester tool*

Pre-Analysis (1)

+

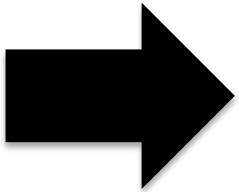
Code Inspect

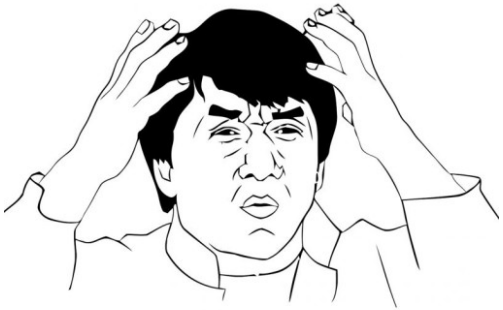
```
public void onCreate()  {
  java.lang.String $S1, $S2;
  $S1 = "[redacted]buKTe7e";
  $S2 = "[redacted]98BF7e3";
  staticinvoke <Parse: void initialize(Context, String, String)>(this, $S1, $S2);
 }
```



- App ID
- Client Key

Parse REST API





TECHNISCHE UNIVERSITÄT DARMSTADT

SECURE SOFTWARE ENGINEERING GROUP

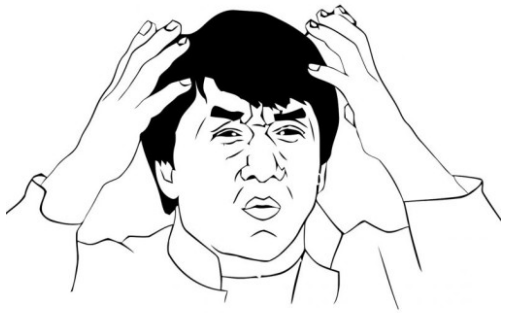
Fraunhofer SIT

Pre-Analysis (2)

Q:

[...]"***The App-Secret key should be kept private*** - *but when releasing the app they can be reversed by some guys. I want to know what is the best thing to* ***encrypt, obfuscate*** *or whatever to make this secure.*"[...]

(Source: stackoverflow.com)



R:

"*Few ideas, in my opinion only first one gives some guarantee:*

1. ***Keep your secrets on some server on internet****, and when needed just grab them and use.*
2. ***Put your secrets in jni code***
3. ***use obfuscator***

4. Put your secret key as ***last pixels of one of your image*** *in assets* "

(Source: stackoverflow.com)

Pre-Analysis result:

Only a few developers apply “security by obscurity“.

The rest doesn‘t even use obfuscation.

… let’s get ready for a mass-analysis

Mass-Analysis – What do we need?

ToDo	How?
BaaS-identification	• Package-name • Heurisitc (obufscation)
Keys extraction	• *Static*: grep, constant string propagation • *Dynamic*: function hooking, bytecode instrumentation, traffic interception, etc. • *Hybrid*: HARVESTER (tool)
Table-name/bucket-name extraction	• *Static*: grep, constant string propagation • *Dynamic*: function hooking, bytecode instrumentation, traffic interception, etc. • *Hybrid*: HARVESTER (tool)
Data extraction	Rest API + Python

HARVESTER

Combines static code analysis (backward slicing) with bytecode manipulation and dynamic code execution

```
initialize(v1, dec("yicqco44"))
```

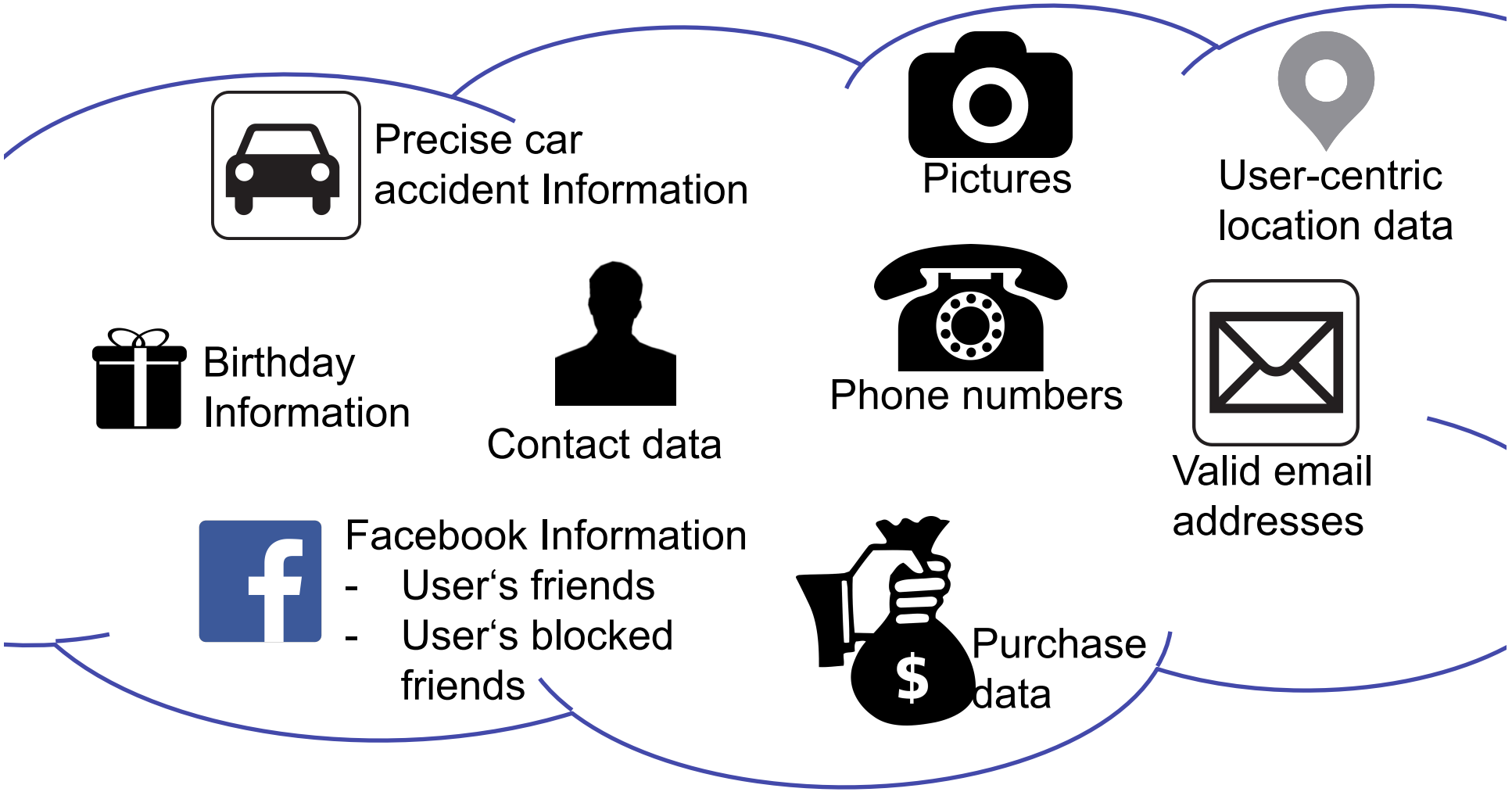
```
initialize("AppID", "ClientID")
```

Harvesting Runtime Data in Android Applications for Identifying Malware and Enhancing Code Analysis
Siegfried Rasthofer, Steven Arzt, Marc Miltenberger, Eric Bodden
Technical Report, February 2015.

So … how bad is it?

OUR FINDINGS

Findings Parse



Findings Parse (2)

Intercepted SMS messages

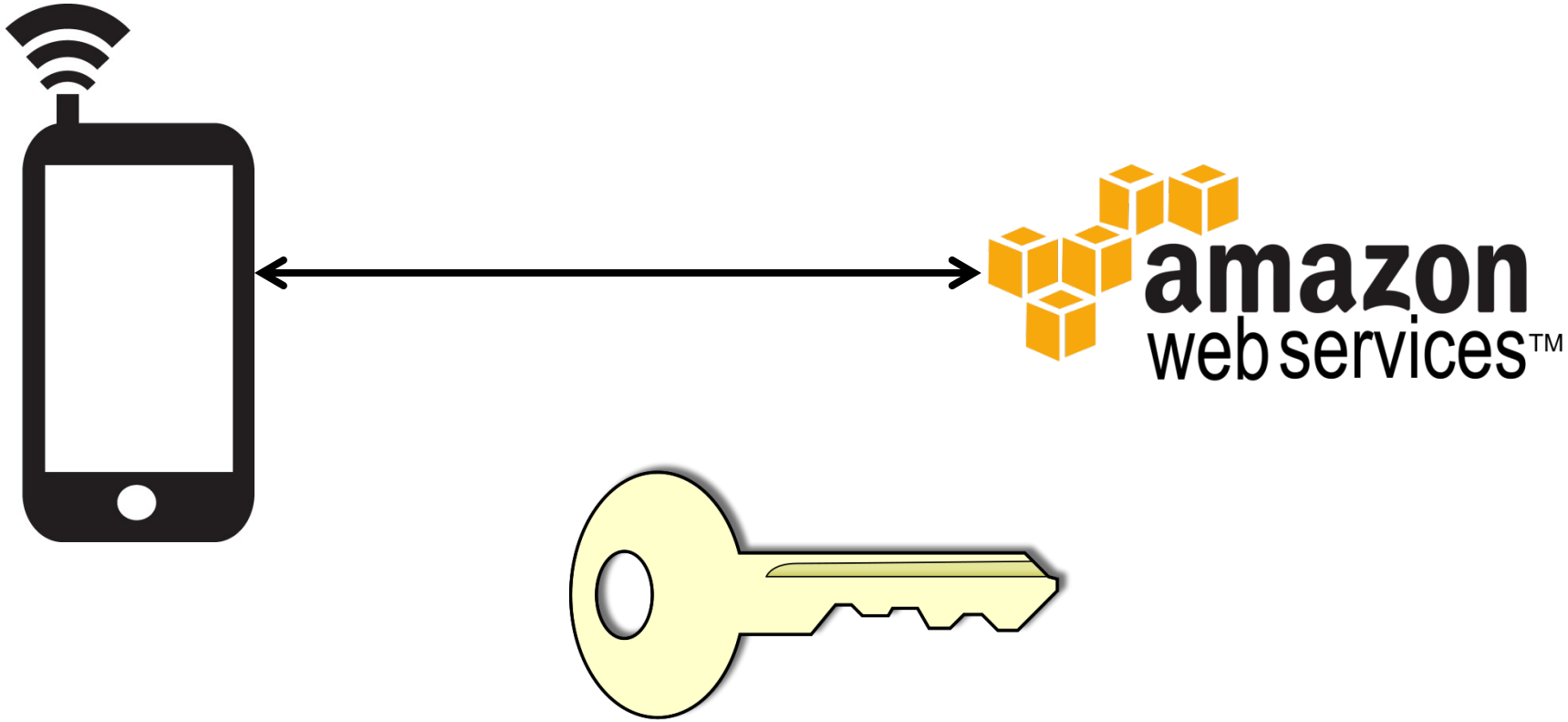
C&C tasks

Leaked data

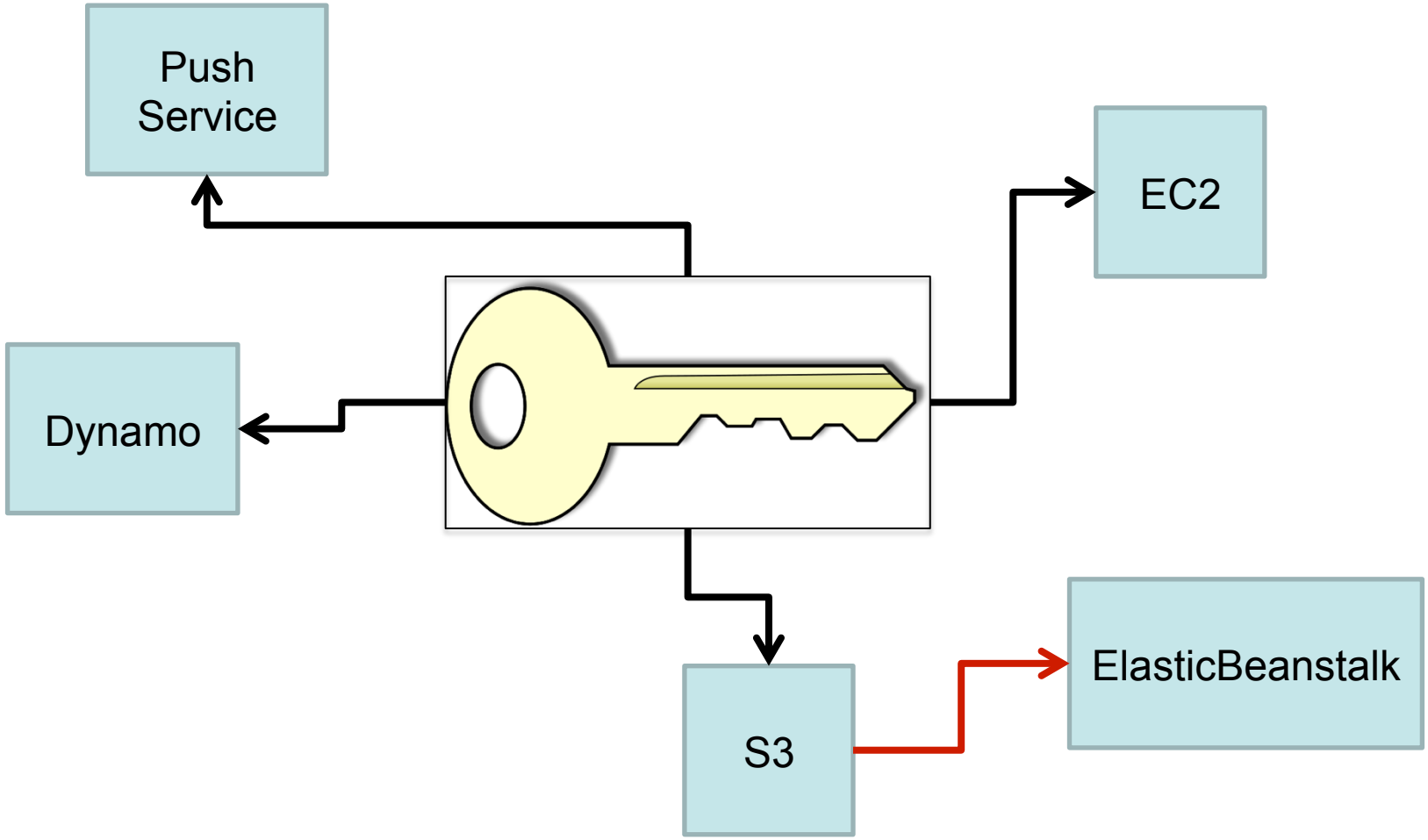
C&C commands

We know what you did this summer: Android Banking Trojan exposing its sins in the cloud
Siegfried Rasthofer, Eric Bodden, Carlos Castillo, Alex Hinchliffe
VirusBulletin 2015, AVAR 2015

Findings Amazon (1)



Findings Amazon (2)



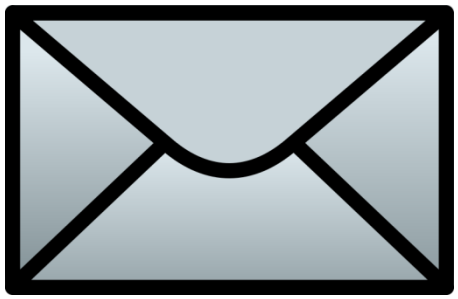
Findings Amazon (3)

Server Backups

Baby Growth Data

Photos

Findings Amazon (4)



Private Messages

Lottery Data

Web Page Content

Responsible Disclosure Process – Parse (Facebook)

	Access to ~56 Mio Data
28th April 2015	Contacted Facebook with a few samples
5th Mai 2015	Facebook verified it and asked for more data
18th Mai 2015	We provided all information
20th Mai 2015	Facebook forwarded everything to Parse (we assume they contacted the developers)
12th Nov 2015	Still access to ~56 Mio Data

How can we get it right?

COUNTERMEASURES

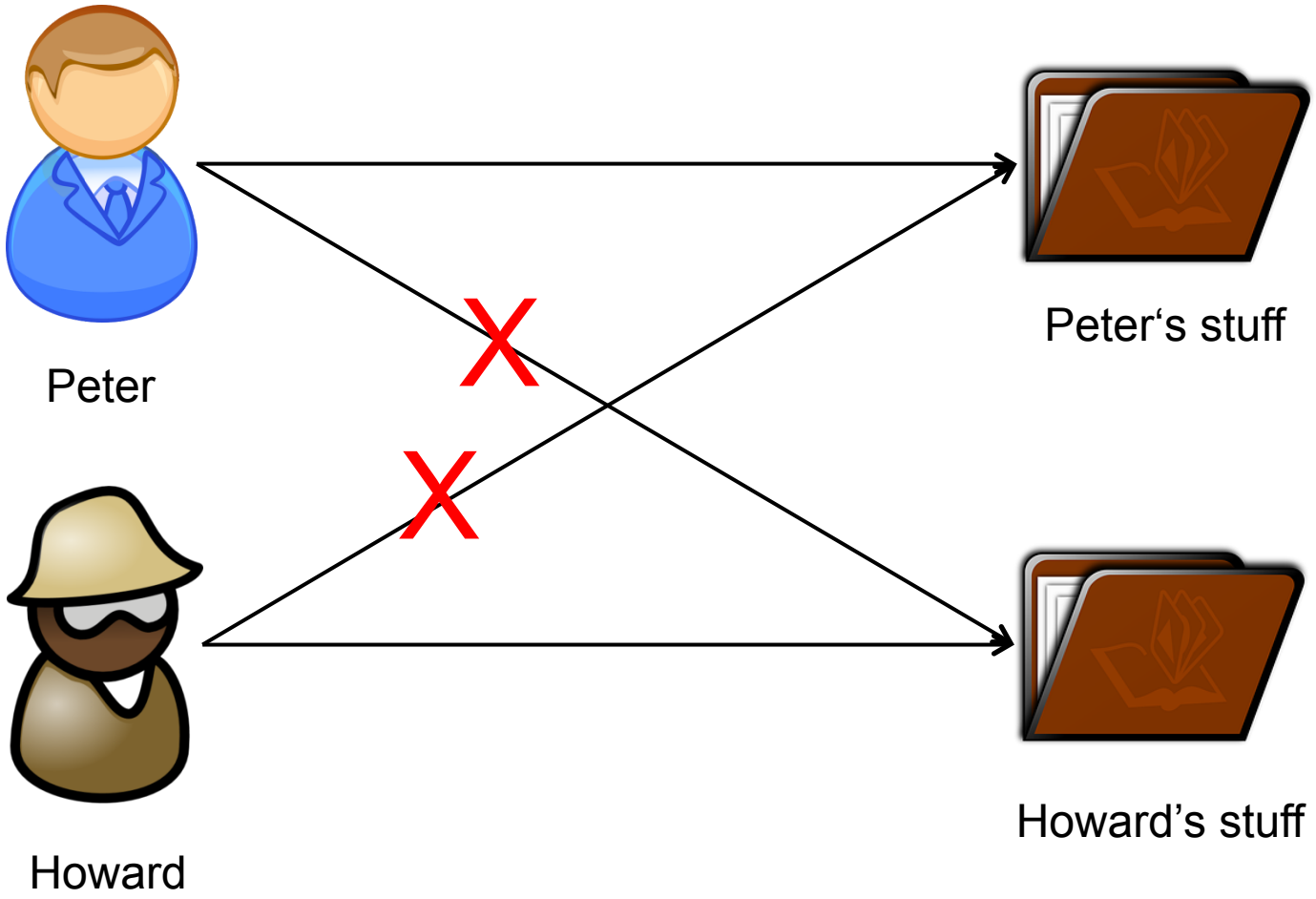
Basic Countermeasures

- Use ACLs to limit power of credentials
  - Lest privilege principle
- Get the credentials out of the app
  - Server-side semantics
  - Expiring tokens
- Check for weird behavior
  - Server-side monitoring
  - Server-side integrity checks

Amazon Cognito (1)

- Identity management
  - Authenticate app users
  - ACLs on objects (folders in S3 buckets, etc.)
- Profile management
  - Associate additional data with accounts
- Automatic synchronization
  - Well, it's online anyways

Amazon Cognito (2)



Amazon Cognito (3)

- What if you don‘t have accounts?

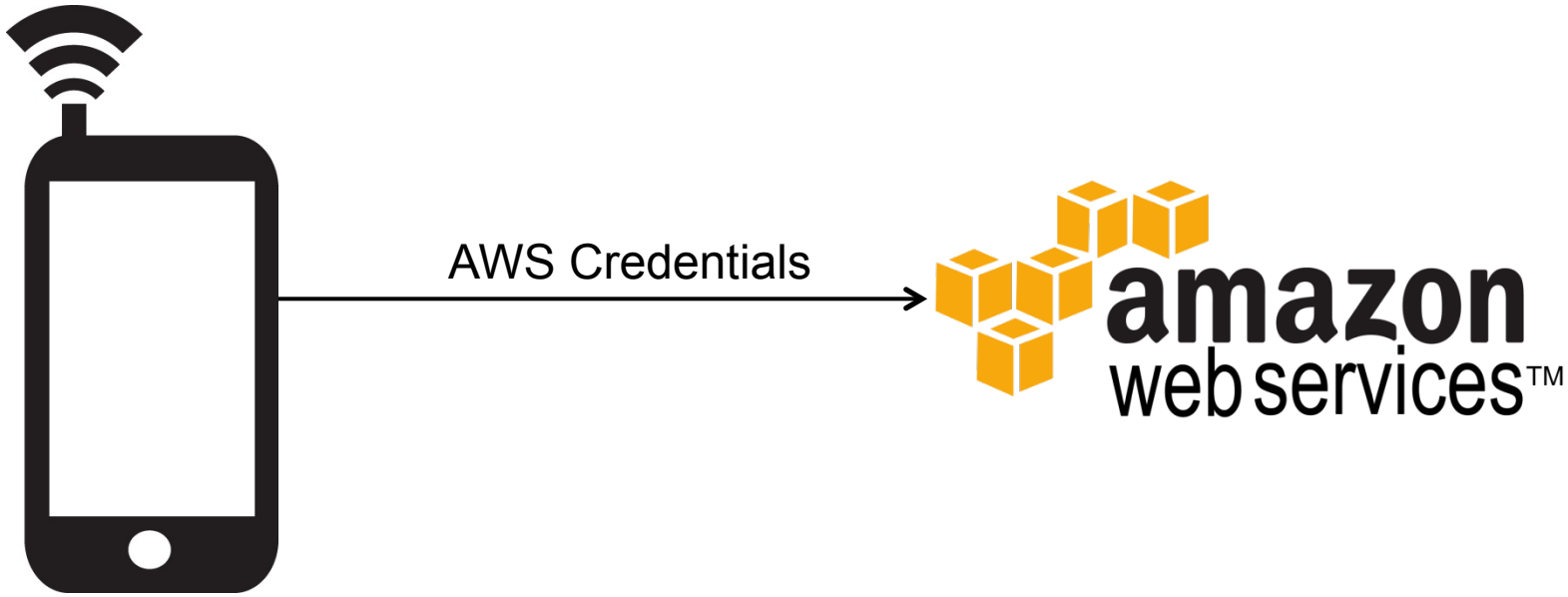
- Goal: Protect AWS credentials for shared backend
  - Can’t isolate users
  - Can still apply least-privilege principle

- Can be a bit confusing
  - AWS Identity and Access Management (IAM)
  - Tons of nomenclature
  - Tons of different web pages with chunks of info

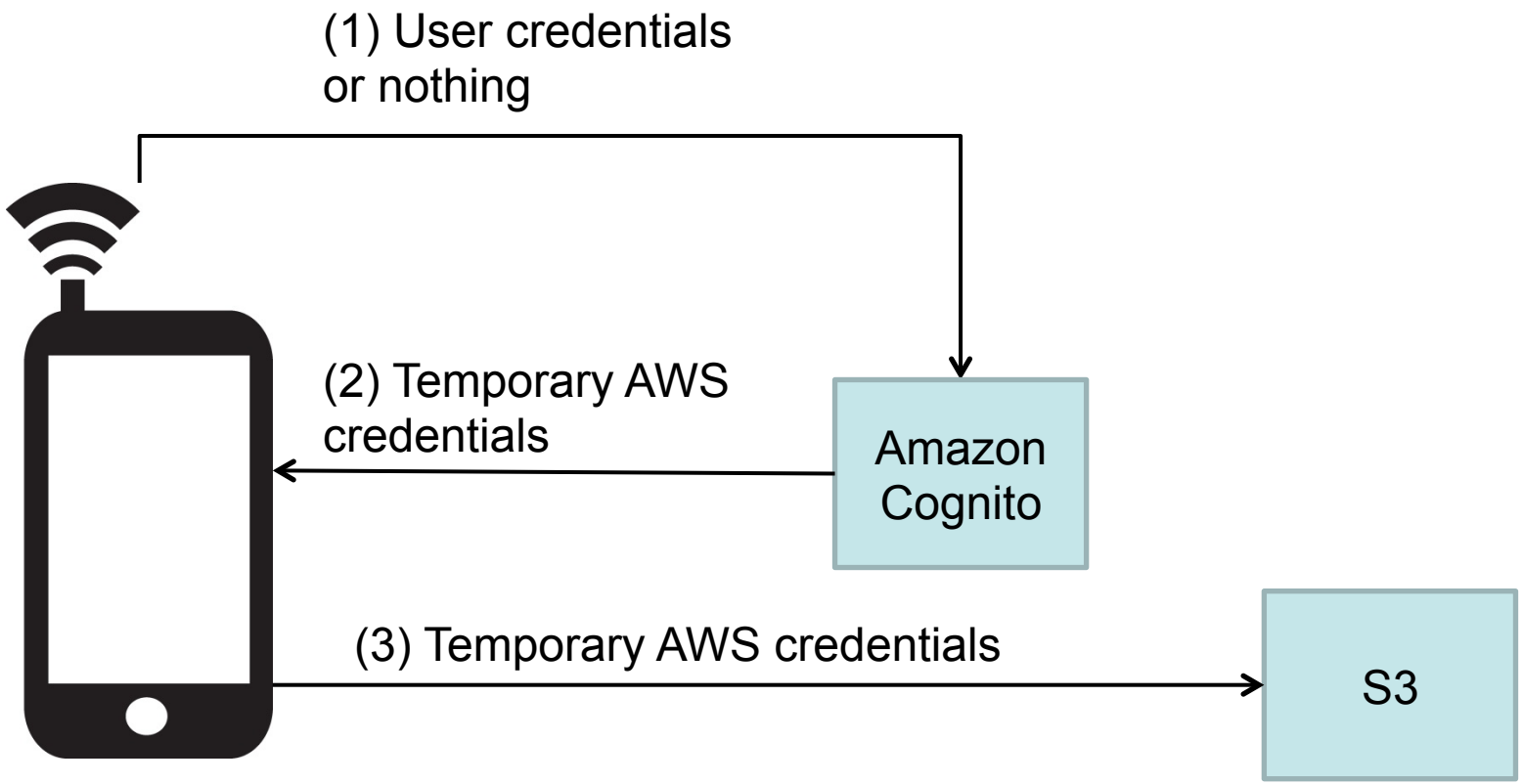
Amazon Cognito (4)

Note: If you created your identity pool before February 2015, you will need to reassociate your roles with your identity pool in order to use this constructor without the roles as parameters. To do so, open the Amazon Cognito Console, select your identity pool, click **Edit Identity Pool**, specify your authenticated and unauthenticated roles, and save the changes.

Amazon Cognito (5)



Amazon Cognito (6)



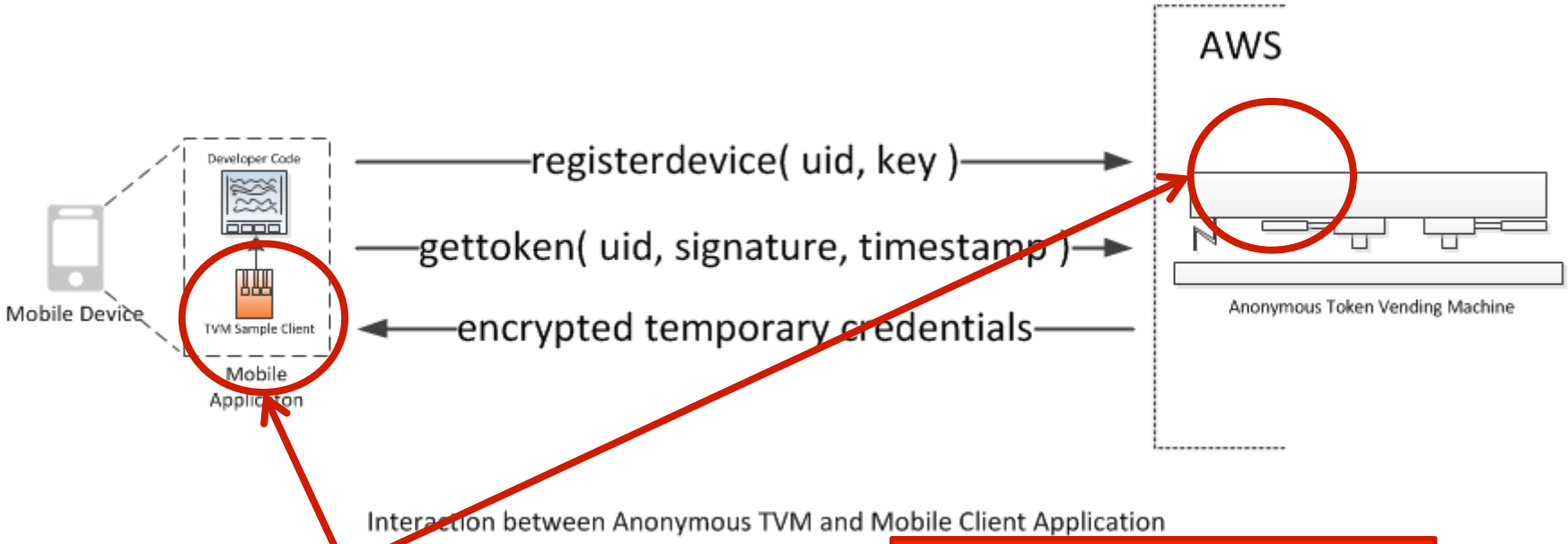
Amazon Cognito (7)

- Overkill for simple apps, but neat

- It’s not popular in the real world
  - Putting root credentials is easier
  - Tutorials (StackOverflow etc.) use root credentials
  - Cognito hasn’t been around that long

- Let‘s turn back the clock
  - Out of the dusty shades of history, there comes…

Amazon Token Vending Machine (1)



Sample available, final implementation is on you

Needs hosting. Tomcat, Elasticbeanstalk anyone?

TECHNISCHE UNIVERSITÄT DARMSTADT

SECURE SOFTWARE ENGINEERING GROUP

Fraunhofer SIT

Amazon Token Vending Machine (2)

Although you will need to use your AWS account credentials to deploy the TVM, we recommend that you do not run the TVM under your AWS account. Instead, create an IAM user and configure the TVM to use the credentials of this IAM user, which we will call the *TVM user*.

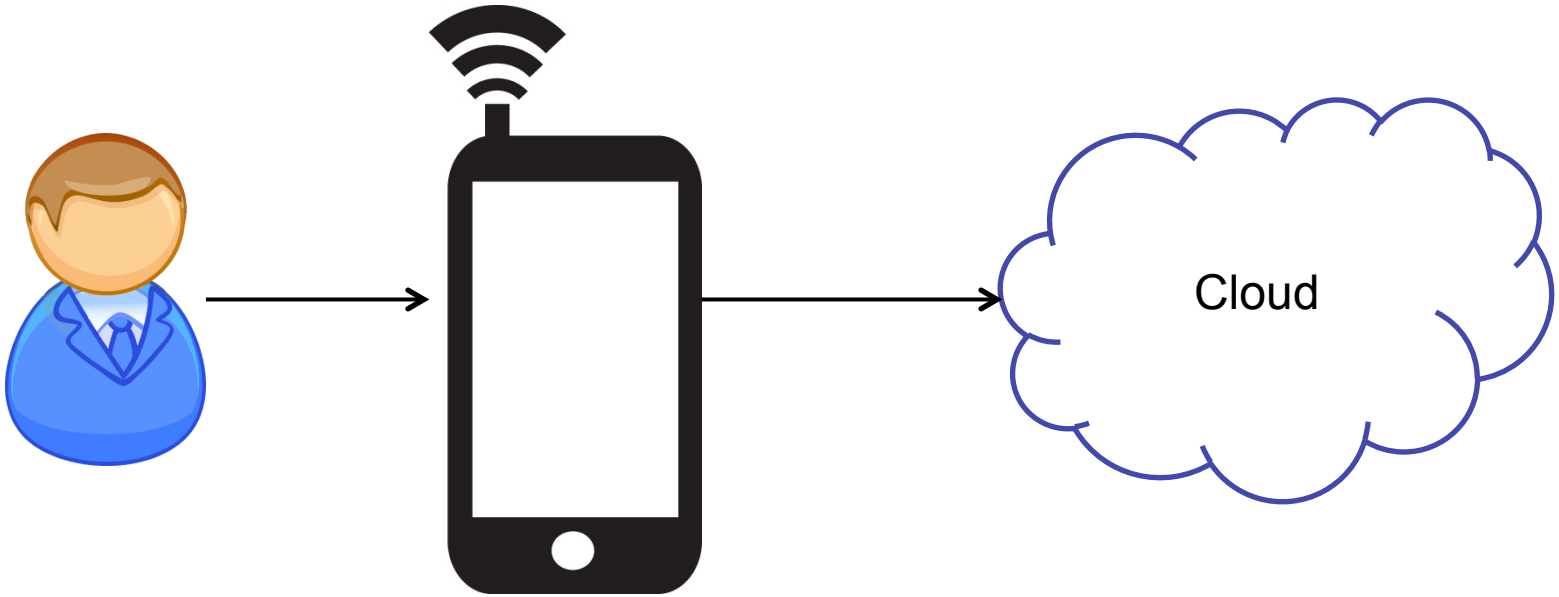
So, we have S3, TVM, IAM, Elastic Beanstalk

Amazon Token Vending Machine (3)

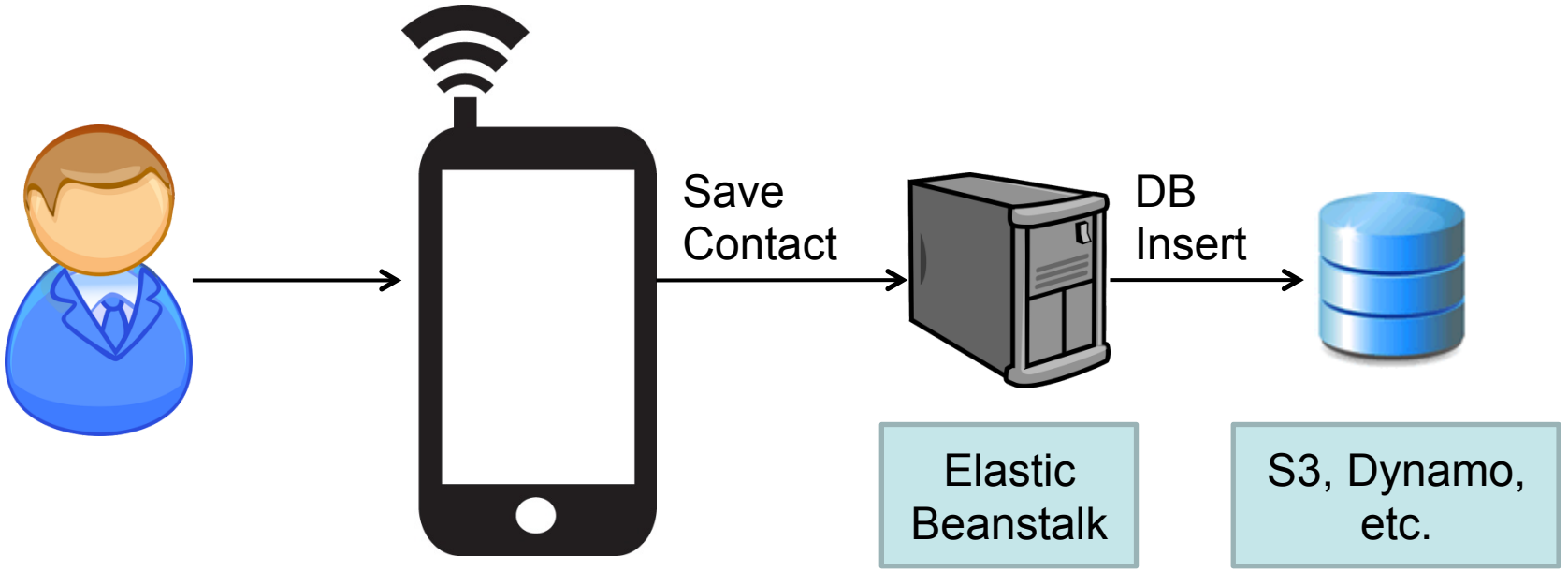
- What if I want ACLs?
- Identity TVM samples do exist, but…

You would need to modify the provided samples in order to implement these user-specific policy objects. For more information about policy objects, see the Identity and Access Management (IAM) documentation

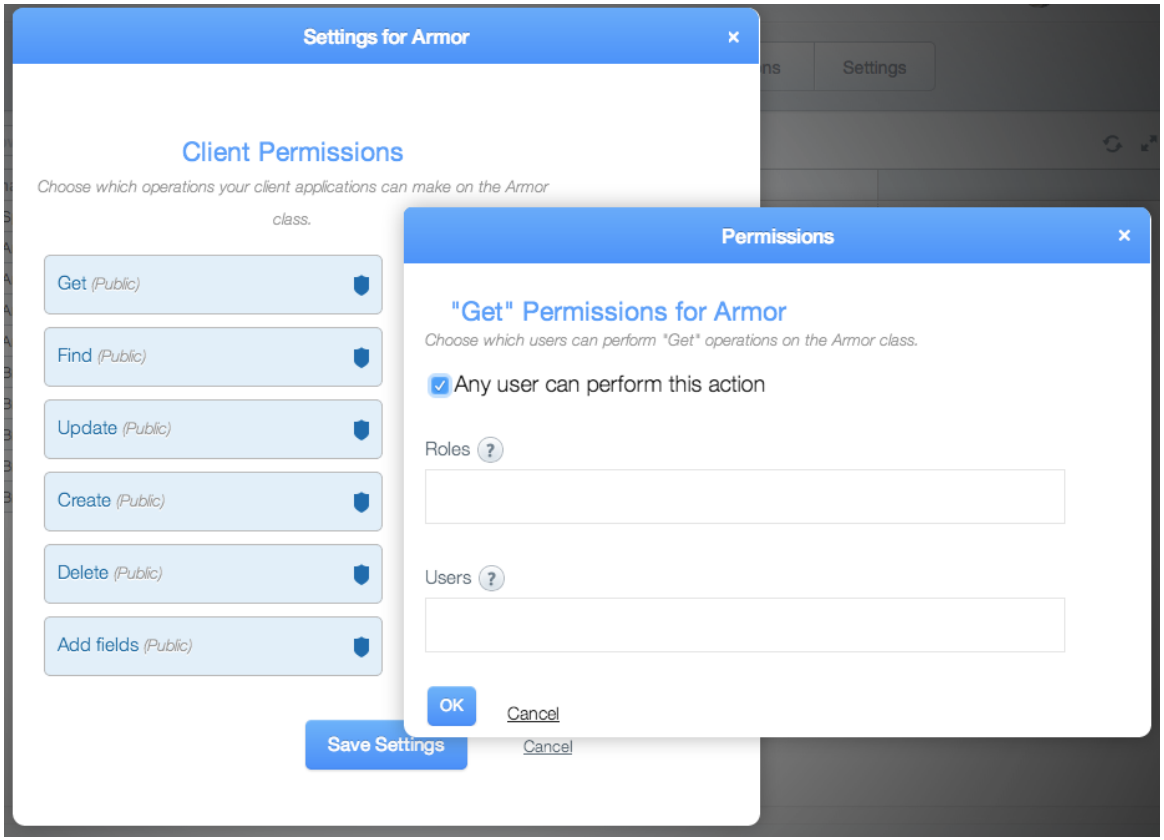
Amazon Middleware (1)



Amazon Middleware (2)

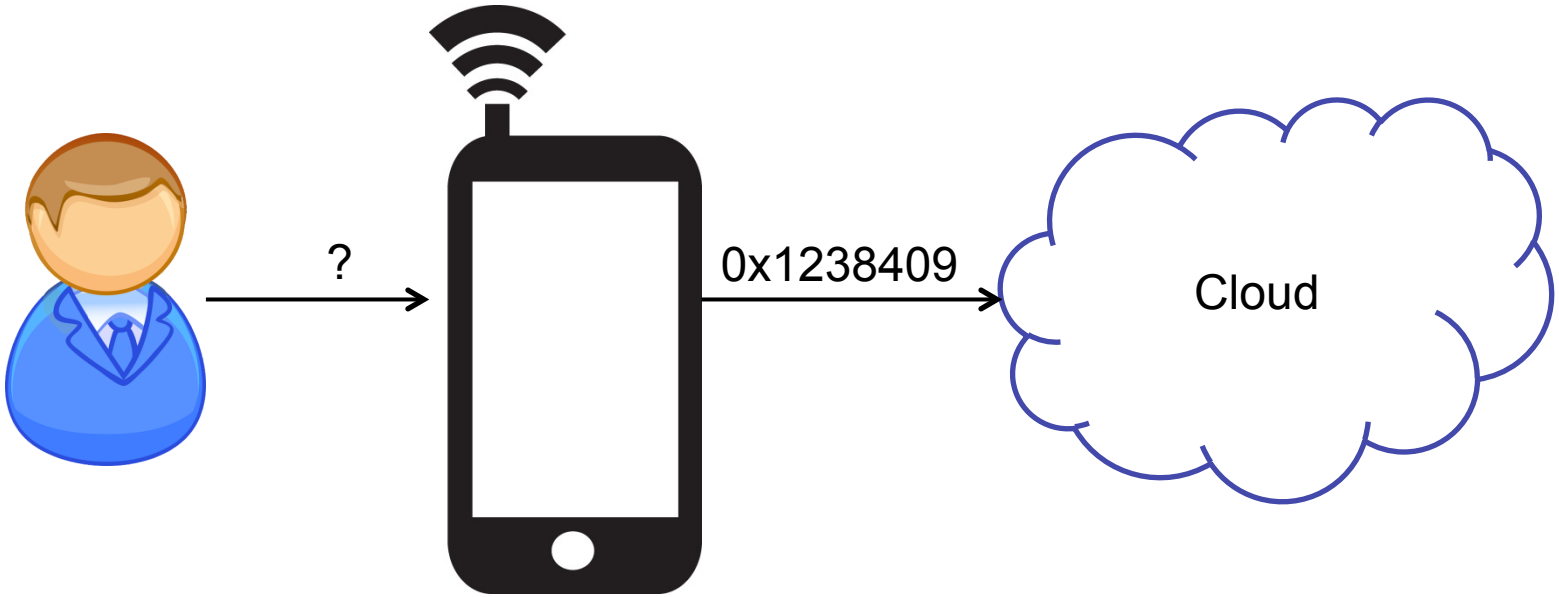


Parse.com ACLs (1)



Source: http://blog.parse.com/learn/engineering/parse-security-ii-class-hysteria/

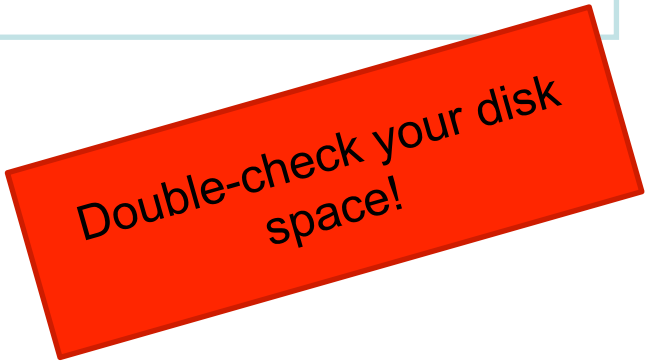
Parse.com ACLs (2)



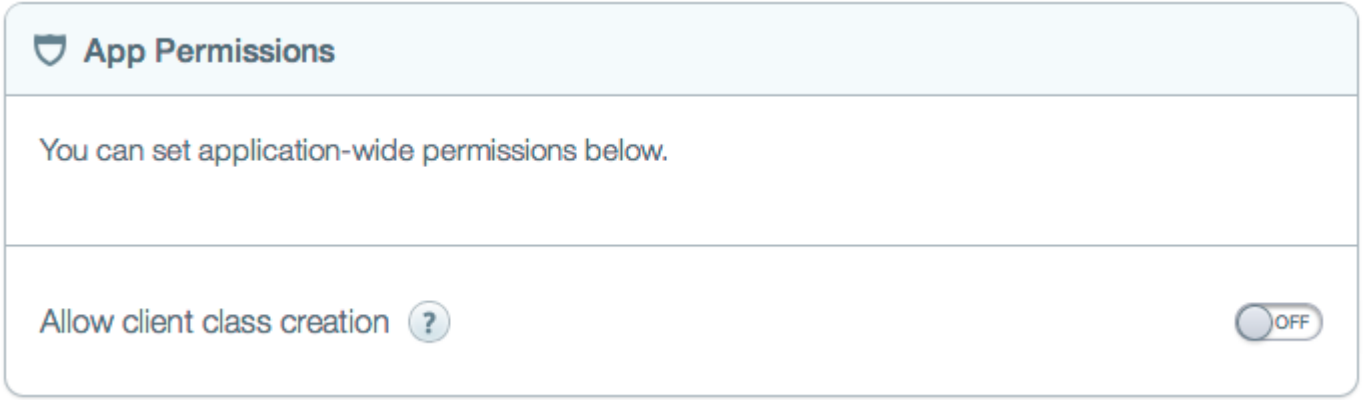
http://blog.parse.com/announcements/protect-user-data-with-new-parse-features/

Parse.com ACLs (3)

Anonymous users are special, however, in that once logged out, the user cannot be recovered – a new user will need to be created, and the original user (and its associated data) will be orphaned.



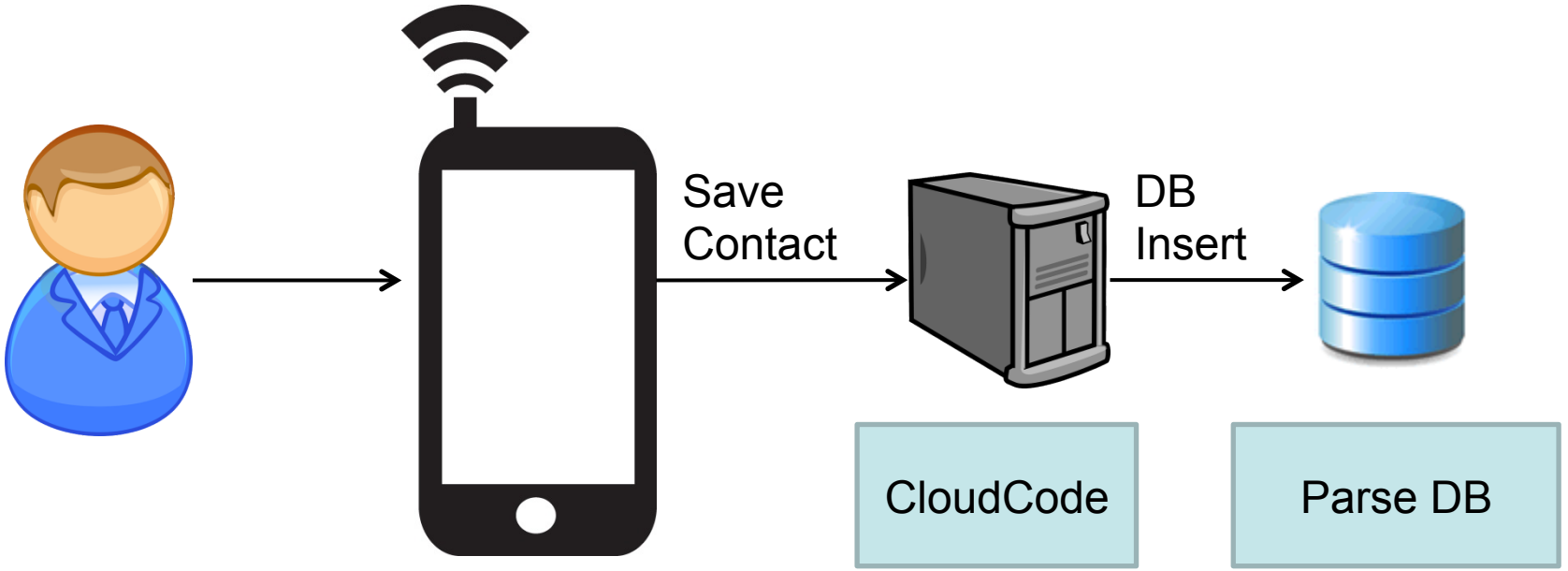
Parse.com Global Settings



Get this wrong and offer free disk space to anyone!

Source: http://blog.parse.com/learn/engineering/parse-security-ii-class-hysteria/

Parse Middleware



What now?

THE WISHLIST

What shall change?

Improved Documentation

Checks and Alerts

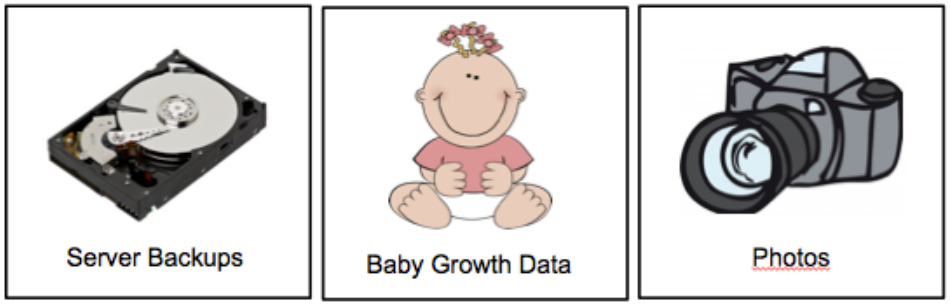
Legal Framework

Takeaway Messages

- Security in the cloud doesn‘t come for free
- Attacks are free, effortless, and simple
- Mitigation techniques exist
  - ➢ People must care about them
  - ➢ Secure your apps now – we’re there!

Mass-Analysis – What do we need?

ToDo	How?
Parse-Identification	• Package-name • Heurisitc (obufscation)
Keys Extraction	• *Static*: grep, constant string propagation • *Dynamic*: function hooking, bytecode instrumentation, traffic interception, etc. • *Hybrid*: HARVESTER (tool)
Table-name Extraction	• *Static*: grep, constant string propagation • *Dynamic*: function hooking, bytecode instrumentation, traffic interception, etc. • *Hybrid*: HARVESTER (tool)
Data Extraction	Rest API + Python



How can we get it right?

COUNTERMEASURES

All is lost? Not quite.

THE WISHLIST

Siegfried Rasthofer
Secure Software Engineering Group
Email: siegfried.rasthofer@cased.de
Twitter: @CodeInspect

Steven Arzt
Secure Software Engineering Group
Email: steven.arzt@cased.de

Blog: http://sse-blog.ec-spride.de
Website: http://sse.ec-spride.de

TECHNISCHE UNIVERSITÄT DARMSTADT
SECURE SOFTWARE ENGINEERING GROUP
Fraunhofer SIT