Accountability for the Cloud

Walid Benghabrit,
Ronan-Alexandre Cherrueau,
Jean-Claude Royer &
Mario Südholt

Ascola team
Inria, Mines Nantes, Lina
Journée Cloud, Nantes France

September 19, 2014

Accountability

General Definition

"In ethics and governance, accountability is answerability, blameworthiness, liability, and the expectation of account-giving.

In leadership roles, accountability is the acknowledgment and assumption of responsibility for actions, products, decisions, and policies including the administration, governance, and implementation within the scope of the obligation to report, explain and be answerable for resulting consequences."

Accountability

General Definition – The good part!

"In ethics and governance, accountability is answerability, blameworthiness, liability, and the expectation of account-giving

In leadership roles, accountability is the acknowledgment and assumption of responsibility for actions, products, decisions, and policies including the administration, governance, and implementation within the scope of the obligation to report, explain and be answerable for resulting consequences."

Accountability in Real Life

- Two means: *preventive* & *retrospective* accountability
- A real life example – The bank robbery
  - Bank robber arrives hooded and armed
    - ⇒ Bank security officer doesn't let the robber enter
  - Bank robber arrives with a hidden weapon
    - ⇒ Bank security officer lets the robber enter
  - It's easy to enter in a bank for an holdup! Why everybody doesn't rob a bank?
    - ⇒ Legal risks!

Accountability in Real Life

- Two means: *preventive* & *retrospective* accountability
- A real life example – The bank robbery
  - Bank robber arrives hooded and armed
    ⇒ Bank security officer doesn't let the robber enter
  - Bank robber arrives with a hidden weapon
    ⇒ Bank security officer lets the robber enter
  - It's easy to enter in a bank for an holdup! Why everybody doesn't rob a bank?
    ⇒ Legal risks!

Accountability in Real Life

- Two means: *preventive* & *retrospective* accountability
- A real life example – The bank robbery
  - Bank robber arrives hooded and armed
    - ⇒ Bank security officer doesn't let the robber enter
  - Bank robber arrives with a hidden weapon
    - ⇒ Bank security officer lets the robber enter
  - It's easy to enter in a bank for an holdup! Why everybody doesn't rob a bank?
    - ⇒ Legal risks!

Accountability in Real Life

- Two means: *preventive* & *retrospective* accountability

- A real life example – The bank robbery
  - Bank robber arrives hooded and armed
    - ⇒ Bank security officer doesn't let the robber enter
    - ⇒ *Preventive* accountability that avoids
  - Bank robber arrives with a hidden weapon
    - ⇒ Bank security officer lets the robber enter
  - It's easy to enter in a bank for an holdup! Why everybody doesn't rob a bank?
    - ⇒ Legal risks!

Accountability in Real Life

- Two means: *preventive* & *retrospective* accountability

- A real life example – The bank robbery
  - Bank robber arrives hooded and armed
    - ⇒ Bank security officer doesn't let the robber enter
    - ⇒ *Preventive* accountability that avoids
  - Bank robber arrives with a hidden weapon
    - ⇒ Bank security officer lets the robber enter
  - It's easy to enter in a bank for an holdup! Why everybody doesn't rob a bank?
    - ⇒ Legal risks!

Accountability in Real Life

- Two means: *preventive* & *retrospective* accountability

- A real life example – The bank robbery
  - Bank robber arrives hooded and armed
    - ⇒ Bank security officer doesn't let the robber enter
    - ⇒ *Preventive* accountability that avoids
  - Bank robber arrives with a hidden weapon
    - ⇒ Bank security officer lets the robber enter

  - It's easy to enter in a bank for an holdup! Why everybody doesn't rob a bank?
    - ⇒ Legal risks!

Accountability in Real Life

- Two means: *preventive* & *retrospective* accountability

- A real life example – The bank robbery
  - Bank robber arrives hooded and armed
    - ⇒ Bank security officer doesn't let the robber enter
    - ⇒ *Preventive* accountability that avoids
  - Bank robber arrives with a hidden weapon
    - ⇒ Bank security officer lets the robber enter

  - It's easy to enter in a bank for an holdup! Why everybody doesn't rob a bank?
    - ⇒ Legal risks!

Accountability in Real Life

- Two means: *preventive* & *retrospective* accountability

- A real life example – The bank robbery
  - Bank robber arrives hooded and armed
    - ⇒ Bank security officer doesn't let the robber enter
    - ⇒ *Preventive* accountability that avoids
  - Bank robber arrives with a hidden weapon
    - ⇒ Bank security officer lets the robber enter

  - It's easy to enter in a bank for an holdup! Why everybody doesn't rob a bank?
    - ⇒ Legal risks!
    - ⇒ *Retrospective* accountability that corrects and imposes consequences

Accountability for the Cloud

Why? [Sam01]

- General approaches are not sufficient
- They are too restrictive
- They are inadequate for a connected world

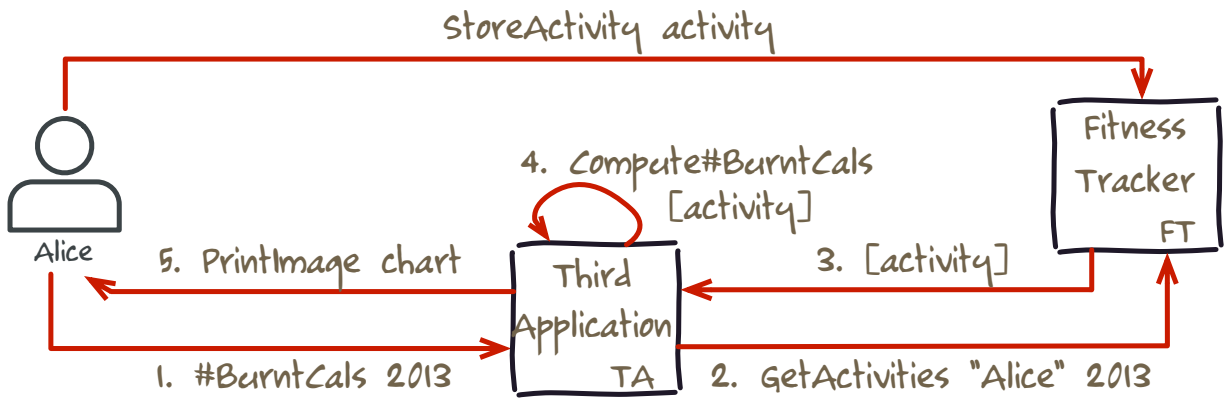
How? [WABL+08]

- Preventive accountability policy:
  - Prevent data from “escaping” when it’s applicable.
- Retrospective accountability policy:
  - Detective controls to identify risks.
  - Corrective controls to correct undesired (past) outcomes.

Fitness Tracker Example (Running Example)

Fitness Tracker:

Activity:
- id: "Alice",
- date: 2014-09-19,
- duration: 45,
- circuit: [GPS(...)],
- bcals: 310



Policy Examples:

- Alice only authorizes TA to get her activities, else …
- Alice authorizes TA to only read id, date and bcals, else …
- Alice requires FT to delete data after 2 years, else …

Fitness Tracker Example (Running Example)

Policy Examples:

- Alice only authorizes TA to get her activities, else …
- Alice authorizes TA to only read id, date and bcals, else …
- Alice requires FT to delete data after 2 years, else …

Accountability Representation Framework

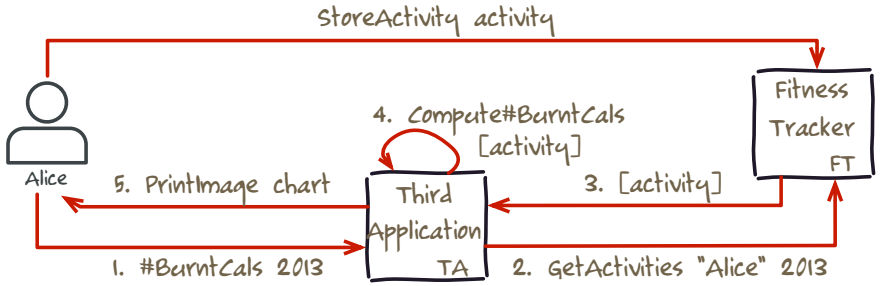
Accountability Representation Framework

- Express accountability policies:
  - Alice only authorizes TA to get her activities.
  - Alice authorizes TA to only read id, data and bcals.
  - Alice requires FT to delete data after 2 years.
- Readable language close to real obligations.
- Help lawyers and designers to introduce accountability.
- Current work [BGR+14a]:
  - Abstract Accountability Language
  - Model-checking and logical proof

Fitness Tracker Example: Representation Level

Fitness Tracker:

* Alice only authorizes TA to get her activities.
* Alice authorizes TA to only read id, date and bcals.
* Alice requires FT to delete data after 2 years.

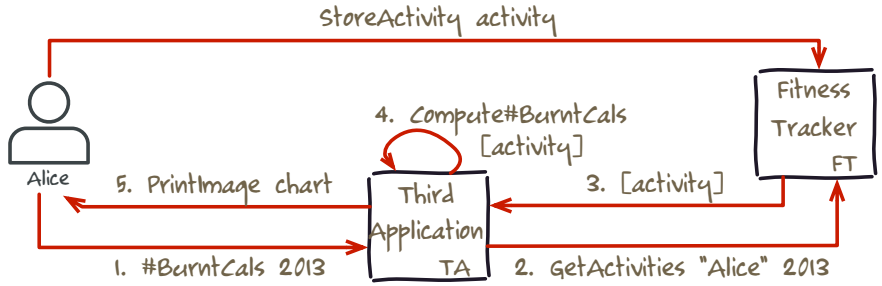



```
TYPE Activity ATTRIBUTES(id, date, duration, circuit, bcals)
AGENT Alice TYPE(Subject) REQUIRED(store #burnt) PROVIDED()
AGENT FT TYPE(DataController) REQUIRED() PROVIDED(store get)
AGENT TA TYPE(DataProcessor) REQUIRED(get) PROVIDED(#burnt)
DATA aData TYPE(Activity) Subject Alice

CLAUSE cAlice:
  FORALL x:Agent DENY x.get[FT](aData) AND DENY x.store[FT](aData)
  PERMIT alice.store[FT](aData) AND
  PERMIT TA.get[FT](aData.id, aData.date, aData,bcals) AND
  PERMIT alice.#burnt[TA](aData) AND
  MUST (FT.delete[aData]() AFTER 2 YEARS)
AUDITING auditor.audit[TA FT]() EVERYDAY
IF_VIOLATED_THEN auditor.sanction[FT](...)

CLAUSE cFT: ...
```

Fitness Tracker Example: Representation Level

Fitness Tracker:

* Alice only authorizes TA to get her activities.
* Alice authorizes TA to only read id, date and bcals.
* Alice requires FT to delete data after 2 years.




```
TYPE Activity ATTRIBUTES(id, date, duration, circuit, bcals)
AGENT Alice TYPE(Subject) REQUIRED(store #burnt) PROVIDED()
AGENT FT TYPE(DataController) REQUIRED() PROVIDED(store get)
AGENT TA TYPE(DataProcessor) REQUIRED(get) PROVIDED(#burnt)
DATA aData TYPE(Activity) Subject Alice

CLAUSE cAlice:
  FORALL x:Agent DENY x.get[FT](aData) AND DENY x.store[FT](aData)
  PERMIT alice.store[FT](aData) AND
  PERMIT TA.get[FT](aData.id, aData.date, aData,bcals) AND
  PERMIT alice.#burnt[TA](aData) AND
  MUST (FT.delete[aData]() AFTER 2 YEARS)
AUDITING auditor.audit[TA FT]() EVERYDAY
IF_VIOLATED_THEN auditor.sanction[FT](...)

CLAUSE cFT: ...
```

Accountability Representation Framework

```
CLAUSE cAlice:
  FORALL x:Agent DENY x.get[FT](aData) AND DENY x.store[FT](aData)
  PERMIT alice.store[FT](aData) AND
  PERMIT TA.get[FT](aData.id, aData.date, aData,bcals) AND
  PERMIT alice.#burnt[TA](aData) AND
  MUST (FT.delete[aData]() AFTER 2 YEARS)
AUDITING auditor.audit[TA FT]() EVERYDAY
IF_VIOLATED_THEN auditor.sanction[FT](...)
```

- AAL to express accountability policies.
- Pure temporal logic approach [BGR+14b] (*e.g.:* `MUST`).
- Verification with the mCRL2 checker [BGRS14]

  $\Rightarrow$ Check that policies are satisfiable (else there is a writing problem).
- Verification with TSPASS prover

  $\Rightarrow$ Check the consistency of accountability policies.

What's next? Enforce accountability policy!

Accountability Representation Framework

```
CLAUSE cAlice:
  FORALL x:Agent DENY x.get[FT](aData) AND DENY x.store[FT](aData)
  PERMIT alice.store[FT](aData) AND
  PERMIT TA.get[FT](aData.id, aData.date, aData,bcals) AND
  PERMIT alice.#burnt[TA](aData) AND
  MUST (FT.delete[aData]() AFTER 2 YEARS)
AUDITING auditor.audit[TA FT]() EVERYDAY
IF_VIOLATED_THEN auditor.sanction[FT](...)
```

- AAL to express accountability policies.
- Pure temporal logic approach [BGR+14b] (*e.g.:* `MUST`).
- Verification with the mCRL2 checker [BGRS14]

  $\Rightarrow$ Check that policies are satisfiable (else there is a writing problem).
- Verification with TSPASS prover

  $\Rightarrow$ Check the consistency of accountability policies.

What's next? Enforce accountability policy!

Accountability Policies Enforcement

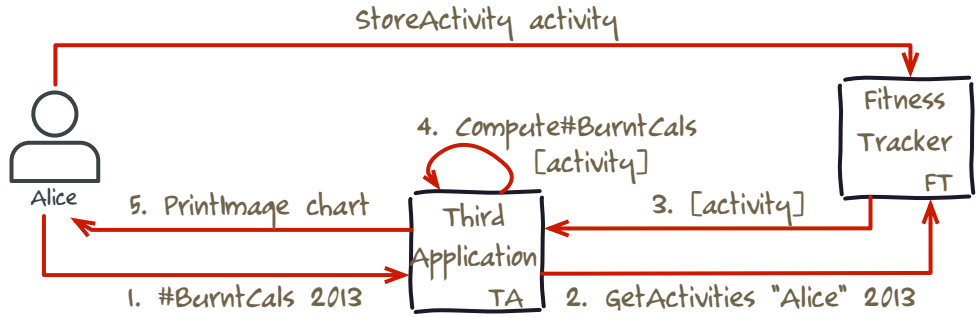
Accountability Policies Enforcement

- Enforce accountability policies:
  - Alice only authorizes TA to get her activities.
  - Alice authorizes TA to only read id, data and bcals.
  - Alice requires FT to delete data after 2 years.
- Preventive means to avoid undesired (past) outcomes.
- Retrospective means to correct and impose consequences:
  - Detective controls to identify risks.
  - Corrective controls to correct undesired (past) outcomes.
- Current work [CS14]:
  - Aspect4CXF – AOP for Cloud Apps [CCS13]
  - SAdapt – Service Adaptation tool [CSC13]

Fitness Tracker Example: Enforcement Level

Fitness Tracker:

* Alice only authorizes TA to get her activities.
* Alice authorizes TA to only read id, date and bcals.
* Alice requires FT to delete data after 2 years.



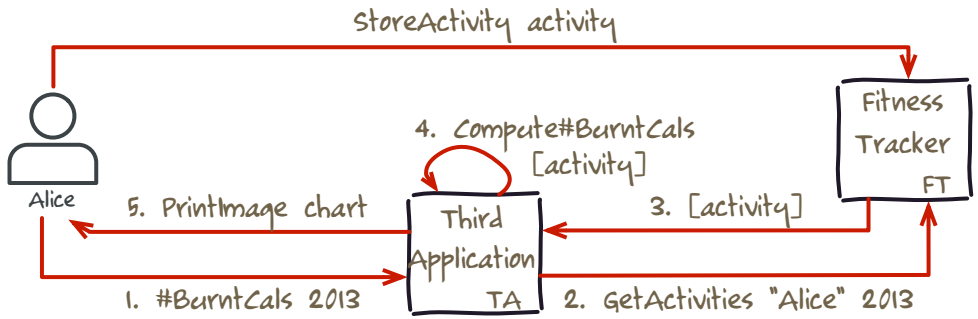

```
policy OnlyReadNonSecret {
  // Pointcut
  FT.getActivities(args,k)_s →
    _s,c,m* →
    [k(args′)_s
     & in(args′,"circuit")
     ]@NotifyAlice

  // Advice
  NotifyAlice {/* Java Code */}
}
```

```
policy DeletedAfter2Year {
  // Pointcut
  FT.dump_m →
    [read(args)_m
     & exist(args, "Alice")
     & (args.date < 2012-09-19)
     ]@DeleteData

  // Advice
  DeleteData {/* Java Code */}
}
```

Fitness Tracker Example: Enforcement Level

Fitness Tracker:

* Alice only authorizes TA to get her activities.
* Alice authorizes TA to only read id, date and bcals.
* Alice requires FT to delete data after 2 years.




```
policy OnlyReadNonSecret {
  // Pointcut
  FT.getActivities(args,k)_s →
    _s,c,m* →
    [k(args′)_s
     & in(args′,"circuit")
     ]@NotifyAlice

  // Advice
  NotifyAlice {/* Java Code */}
}
```

```
policy DeletedAfter2Year {
  // Pointcut
  FT.dump_m →
    [read(args)_m
     & exist(args, "Alice")
     & (args.date < 2012-09-19)
     ]@DeleteData

  // Advice
  DeleteData {/* Java Code */}
}
```

Enforce Expressive Accountability Policies with Stateful Aspect Oriented programming

- Pointcut language:
  - Query that tests a policy violation
  - Query is a sequence of events on the service applications execution trace
- Advice language:
  - Dynamically adapts the service application
  - Adaptation application should correct or prevent a policy violation
- Implementation over Apache CXF [CSC13]

State of Current ASCOLA Work

- Accountability representation framework
  - Abstract Accountability Language
  - Model-checking and logical proof

  ⇒ Express accountability policies and check there satisfiability.
- Accountability policies enforcement
  - Query that tests the violation of an obligation.
  - Adaptation that dynamically adapts the service application.

  ⇒ Enforce preventive and retrospective accountability policies.

Problem?

- Our aspect executor service is in charge of enforcing policies.

⇒ System involves trusted service to trust the global system: Vicious circle!

⇒ Can we do policy enforcement without trusted parties?

State of Current ASCOLA Work

- Accountability representation framework
  - Abstract Accountability Language
  - Model-checking and logical proof

  ⇒ Express accountability policies and check there satisfiability.
- Accountability policies enforcement
  - Query that tests the violation of an obligation.
  - Adaptation that dynamically adapts the service application.

  ⇒ Enforce preventive and retrospective accountability policies.

Problem?

- Our aspect executor service is in charge of enforcing policies.

⇒ System involves trusted service to trust the global system: Vicious circle!

⇒ Can we do policy enforcement without trusted parties?

Owner Centric Control

or How to Continue Using your Cloud Services Without Leaking Personal Information (Open Discussion)

Make personal information indistinguishable

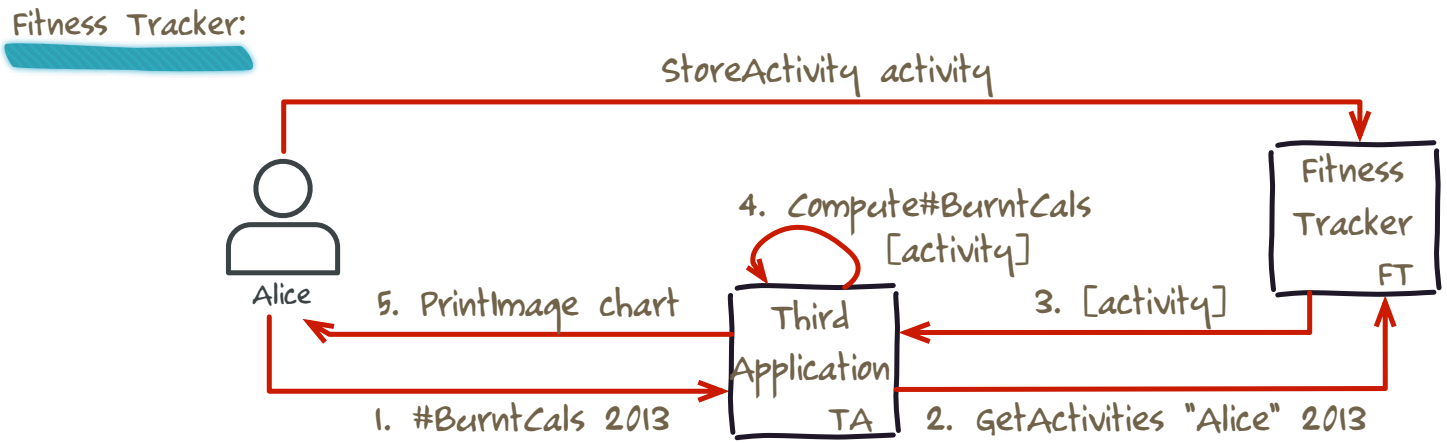
- Sanitization [ABE+99, VBF+04]
  - Removes sensitive information from the data.
- Differential Privacy [RP10, GHH+13]
  - Indistinguishable information even with auxiliary knowledge.
- Symmetric/Asymmetric Encryption
  - Various well-known techniques (e.g.: AES-*, Twofish, RSA, ...).
- Homomorphic Encryption [TLMM13]
  - $pk \in PublicKey;\ sk \in SecretKey;\ d_1, d_2 \in EncryptedData$
    $decrypt(add_{hphic}(d_1, d_2, pk), sk) \equiv decrypt(d_1, sk) + decrypt(d_2, sk)$
- Pulled Functionality [FKDL13]
  - Performs computation at client side.

General Idea

- Various techniques with strong properties **but none is a panacea**:
  - Symmetric encryption is good for image encryption but leads to an unprocessable image.
  - Pulled functionality is good for taking your privacy back but severely reduces the attractiveness of the Cloud model.

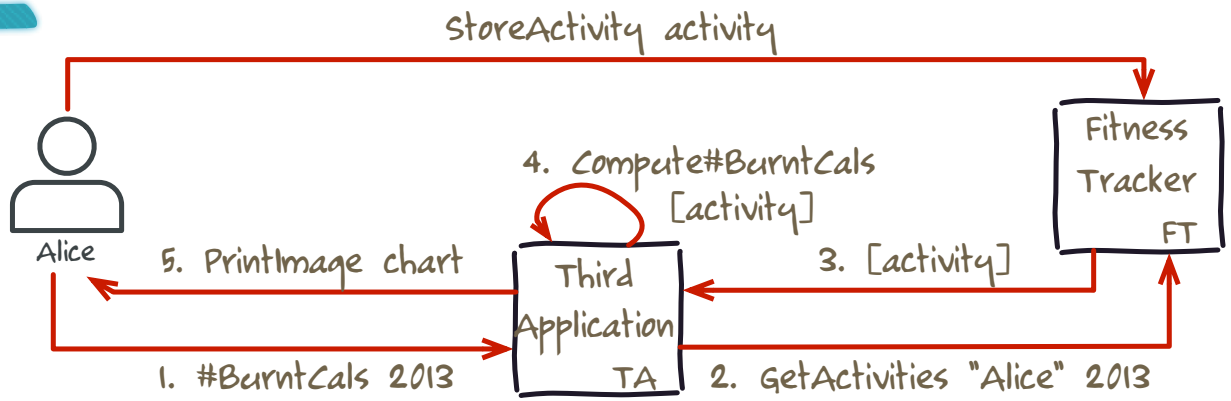
- Idea: combine techniques to **take back your privacy** and **keep cloud competitive**.

Fitness Tracker Example: Owner Centric View

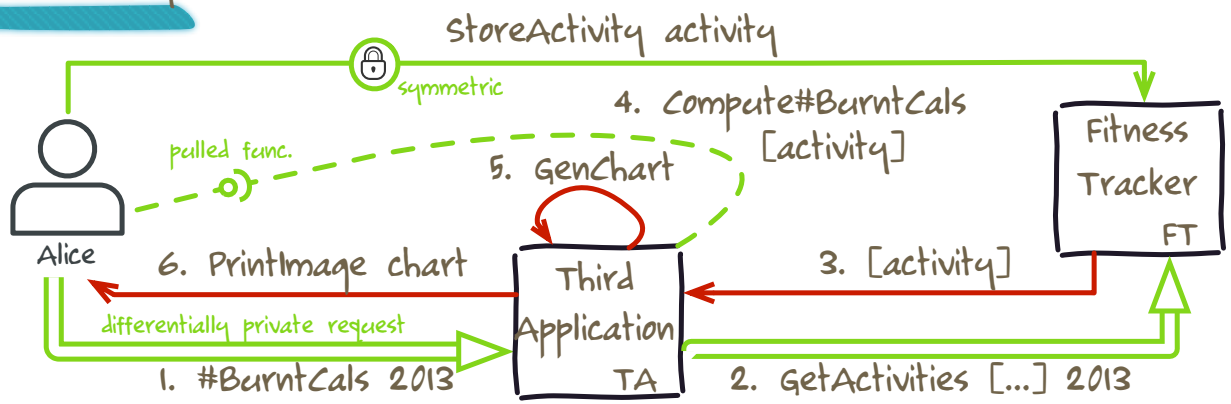


Fitness Tracker Example: Owner Centric View

Fitness Tracker:



Takes Back Alice Privacy:



Contributions

Expected contributions:

- View that limits the visibility of personal info
- Program transformation that adopts the most suitable views
  - ⇒ User takes back its privacy.
  - ⇒ We keep the Cloud model competitive.
  - ⇒ No more trusted parties involved.

Current work:

- Types library that ensures that the transformation is semantically correct.

Owner Centric Control

Your Opinion Matters?!

References (1)

Mike Atallah, Elisa Bertino, Ahmed Elmagarmid, Mohamed Ibrahim, and Vassilios Verykios.
Disclosure limitation of sensitive rules.
In *Knowledge and Data Engineering Exchange, 1999.(KDEX'99) Proceedings. 1999 Workshop on*, pages 45–52. IEEE, 1999.

Walid Benghabrit, Hervé Grall, Jean-Claude Royer, Mohamed Sellami, Monir Azraoui, Kaoutar Elkhiyaoui, Melek Önen, Anderson Santana De Oliveira, and Karin Bernsmed.
A Cloud Accountability Policy Representation Framework.
In *CLOSER - 4th International Conference on Cloud Computing and Services Science*, Barcelone, Espagne, 2014.

References (2)

Walid Benghabrit, Hervé Grall, Jean-Claude Royer, Mohamed Sellami, Karin Bernsmed, and Anderson Santana De Oliveira.
Abstract Accountability Language.
In *IFIPTM - 8th IFIP WG 11.11 International Conference on Trust Management*, Singapore, Singapour, July 2014.

Walid Benghabrit, Hervé Grall, Jean-Claude Royer, and Mohamed Sellami.
Accountability for Abstract Component Design.
In *EUROMICRO DSD/SEAA 2014*, Verona, Italie, August 2014.

References (3)

Ronan-Alexandre Cherrueau, Omar Chebaro, and Mario Südholt.
Flexible and expressive aspect-based control over service compositions in the Cloud.
In *4th International Workshop on Variability & Composition*, Digital Library, Fukuoka, Japon, March 2013. ACM.

Ronan-Alexandre Cherrueau and Mario Südholt.
Enforcing Expressive Accountability Policies.
In *WETICE - IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Parma, Italie, June 2014.

References (4)

Ronan-Alexandre Cherrueau, Mario Südholt, and Omar Chebaro.
Adapting workflows using generic schemas: application to the security of business processes.
In *CloudCom - 5th IEEE International Conference on Cloud Computing Technology and Science - 2013*, pages 519–524, Bristol, Royaume-Uni, December 2013.

Cédric Fournet, Markulf Kohlweiss, George Danezis, and Zhengqin Luo.
Zql: A compiler for privacy-preserving data processing.
In *USENIX Security*, pages 163–178, 2013.

Marco Gaboardi, Andreas Haeberlen, Justin Hsu, Arjun Narayan, and Benjamin C. Pierce.
Linear dependent types for differential privacy.
In *POPL*, pages 357–370, 2013.

References (5)

Jason Reed and Benjamin C. Pierce.
Distance makes the types grow stronger: a calculus for differential privacy.
In *ICFP*, pages 157–168, 2010.

Pierangela Samarati.
Protecting respondents' identities in microdata release.
IEEE Trans. Knowl. Data Eng., 13(6):1010–1027, 2001.

Sai Deep Tetali, Mohsen Lesani, Rupak Majumdar, and Todd Millstein.
Mrcrypt: static analysis for secure cloud computations.
In *Proceedings of the 2013 ACM SIGPLAN international conference on Object oriented programming systems languages & applications OOPSLA*, pages 271–286. ACM, 2013.

References (6)

Vassilios S. Verykios, Elisa Bertino, Igor Nai Fovino, Loredana Parasiliti Provenza, Yücel Saygin, and Yannis Theodoridis. State-of-the-art in privacy preserving data mining. *SIGMOD Record*, 33(1):50–57, 2004.

Daniel J. Weitzner, Harold Abelson, Tim Berners-Lee, Joan Feigenbaum, James A. Hendler, and Gerald J. Sussman. Information accountability. *Commun. ACM*, 51(6):82–87, 2008.