Secure Big Data Processing

Patrick Eugster
Purdue University and TU Darmstadt

Outline

- Context
- Availability and integrity in big data analytics
- Privacy in big data analytics
- Conclusions and outlook

Outline

- **Context**
- Availability and integrity in big data analytics
- Privacy in big data analytics
- Conclusions and outlook

Secure Big Data?

- Big data analytics is commonly performed in 3rd-party *cloud* datacenters (DCs)
- Geo-distribution exacerbates security issues
- Multi-tenancy issue even in single cloud DCs
  - Successful tampering with application *X* may yield access to app *Y*

Cloud Computing

Cloud Computing

Cloud Computing

Cloud Computing

Cloud Computing

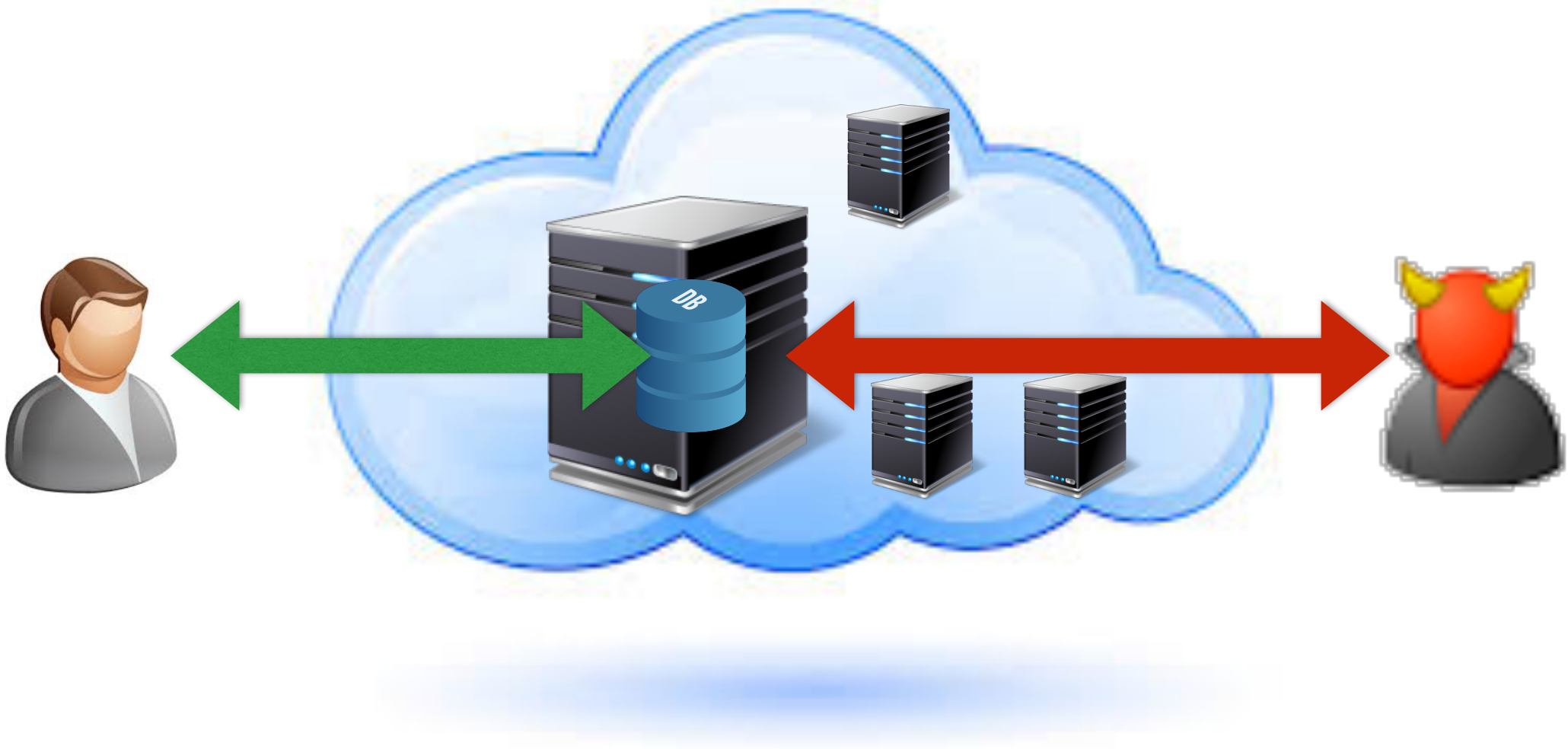
Cloud Computing

Cloud Computing

Cloud Computing

Cloud Computing

Cloud Computing



Assurance

- Multi-faceted problem including
  - Availability
  - Integrity
    - Code: mission-critical, e.g., government, business processes, CPS, disaster response
    - Data: similar
  - Privacy
    - Data: confidential, e.g., government, personal
    - Code: competitive, e.g., algorithmic trading and business processes
  - …

Existing Work

- **Communication-centric**: focus on exchanged messages
  - Firewalls, etc. - perimeter security
- **Data-centric**: mostly data at rest
  - Access control/"differential privacy" (e.g., Airavat [Roy et al.;NSDI'10]), encrypted storage (e.g., DepSky [Bessani et al.;Eurosys'11]), etc.
  - Homomorphic encryption (e.g., [Gentry;STOC'09])
    - Partial (e.g., CryptDB [Popa et al.;CACM'12], MrCrypt [Lesani et al.;OOPSLA'13])
- **Computation-centric**: generating "correct" results
  - Functional encryption, proof-based computation (e.g. Ginger [Setty et al.;S&P'12])

Outline

- Context
- **Availability and integrity in big data analytics**
- Privacy in big data analytics
- Conclusions and outlook

Problem Statement

- How to prevent an attacker from tampering with computations?
- Cryptographic primitives such as zero-knowledge proofs costly (e.g. Ginger [Setty et al.;S&P'12])

Problem

Problem

Problem

Problem

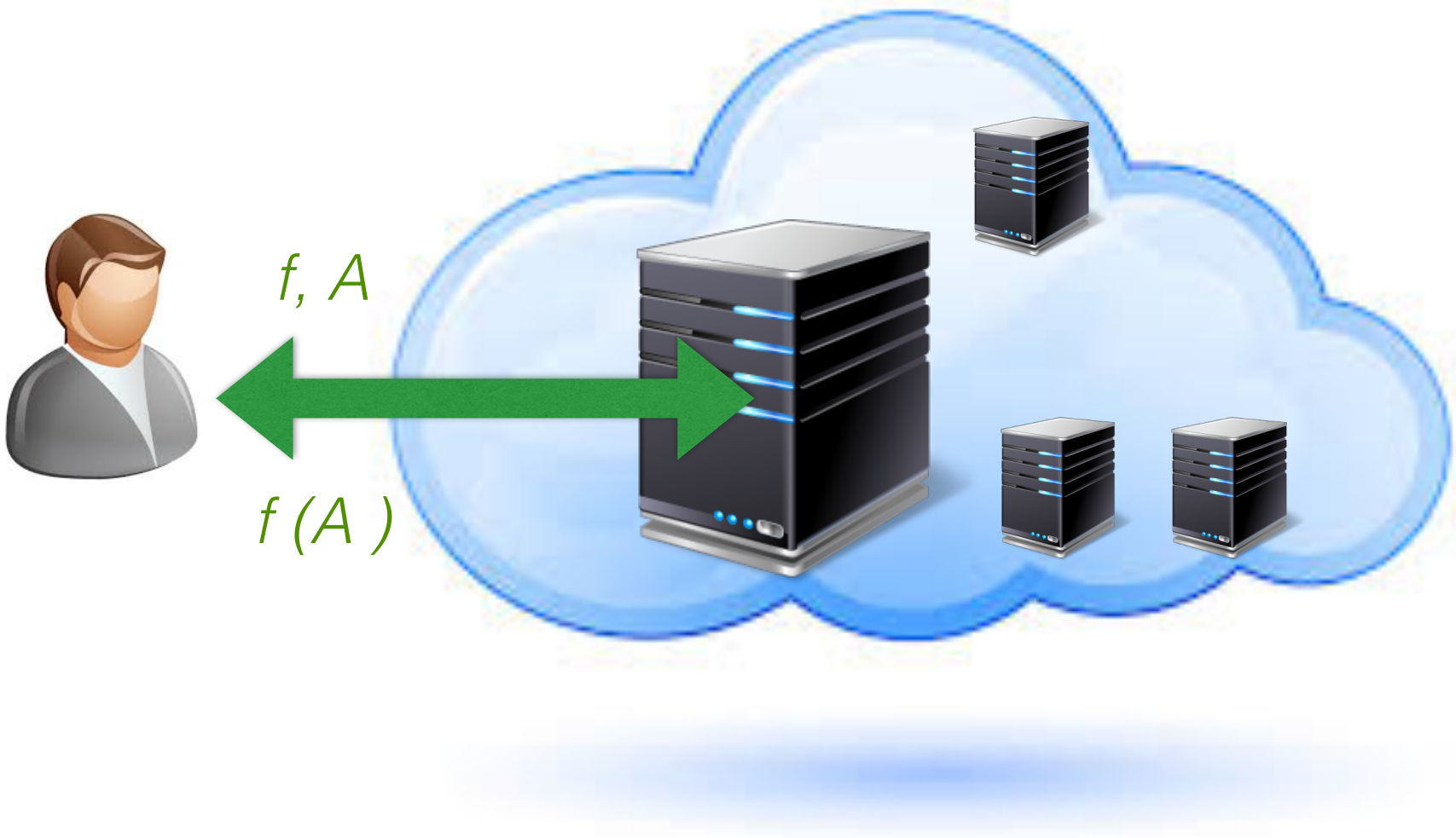
Problem

Problem

Problem

Problem

Problem



Problem

Problem

Problem

Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proofs

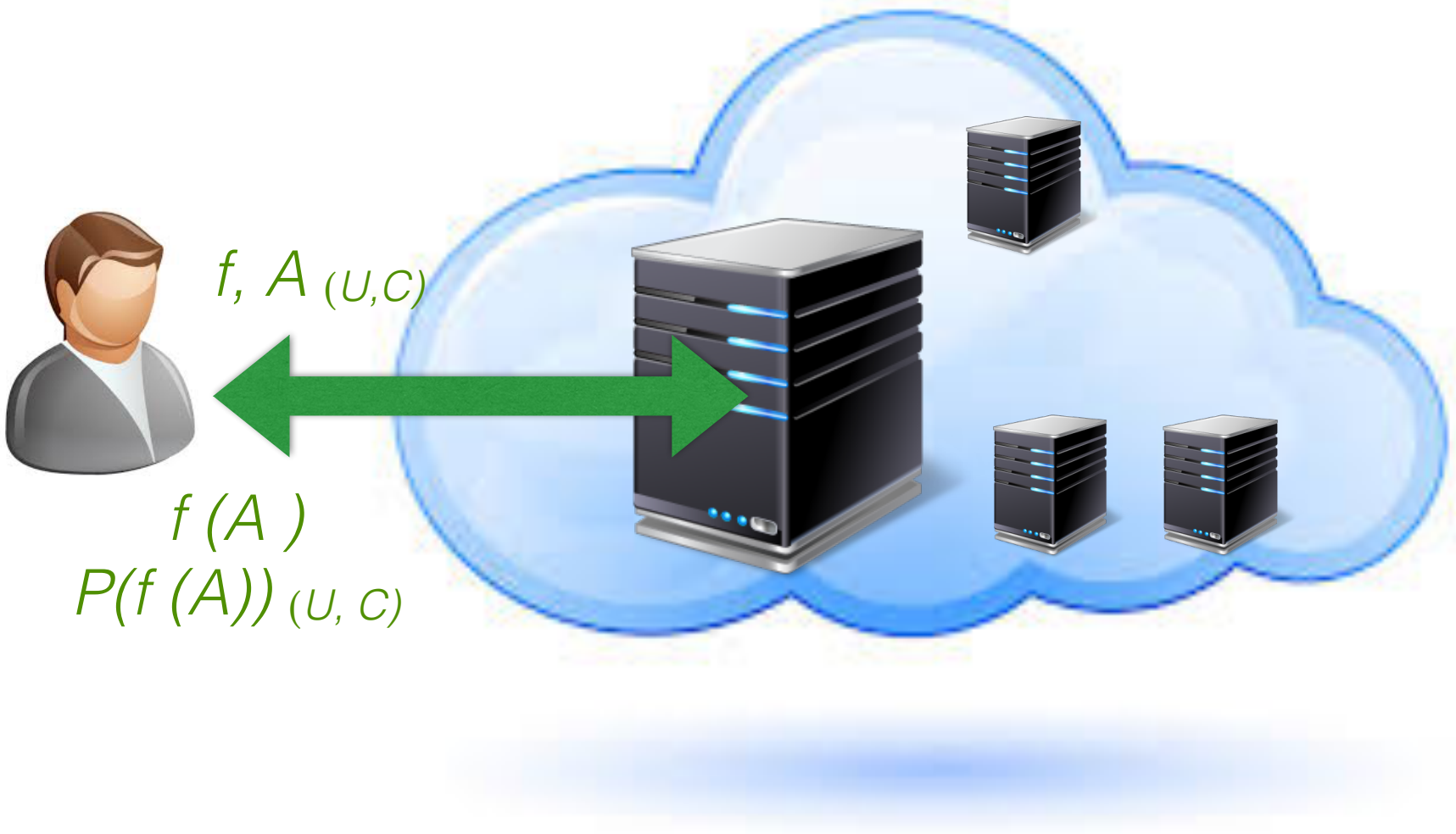
Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proofs



Zero Knowledge Proofs

Zero Knowledge Proofs

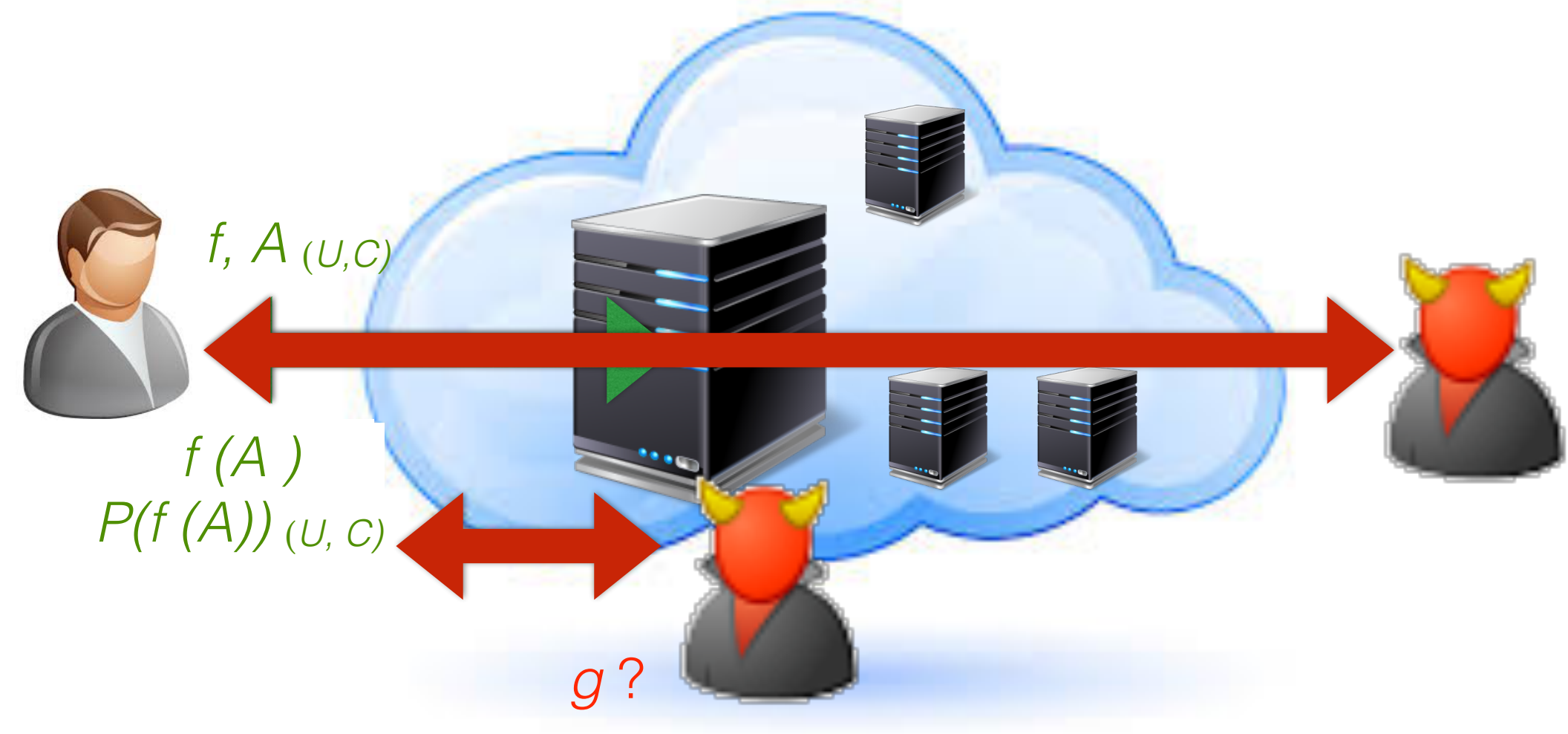
Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proofs



Byzantine Fault Tolerance

- Protect “computation” with BFT [Lamport, Shostak, Pease;TOPLAS’82] replication
  - Processes (state machines) with benign and malicious failures
  - $3f+1$ replicas for f failures in asynchronous distributed systems
    - Safety with $f+1$
    - Liveness with $2f+1$ in synchronous system
  - Comparison of outputs

BFT Replication

BFT Replication

BFT Replication

BFT Replication

BFT Replication

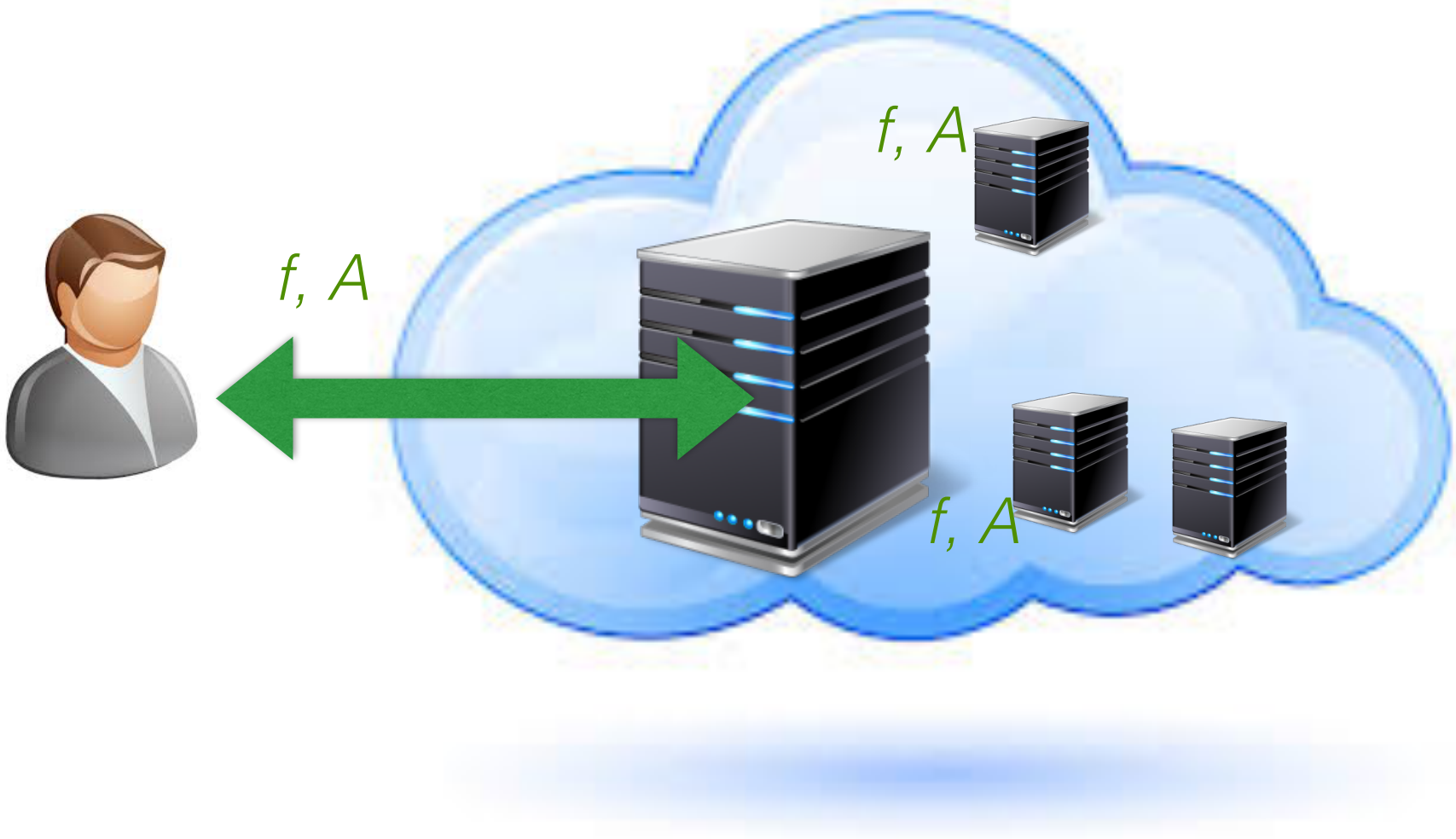
BFT Replication

BFT Replication

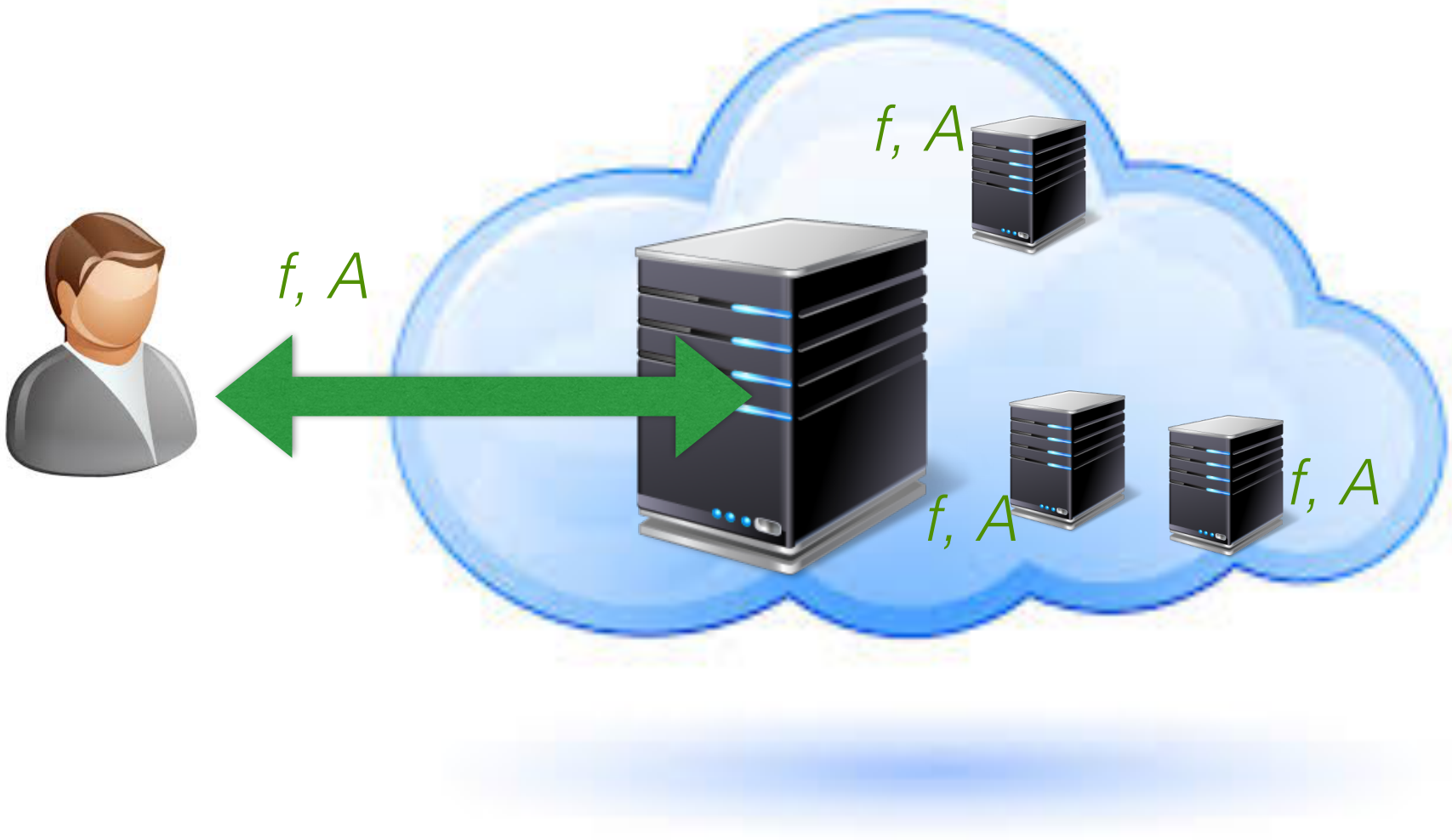
BFT Replication

BFT Replication

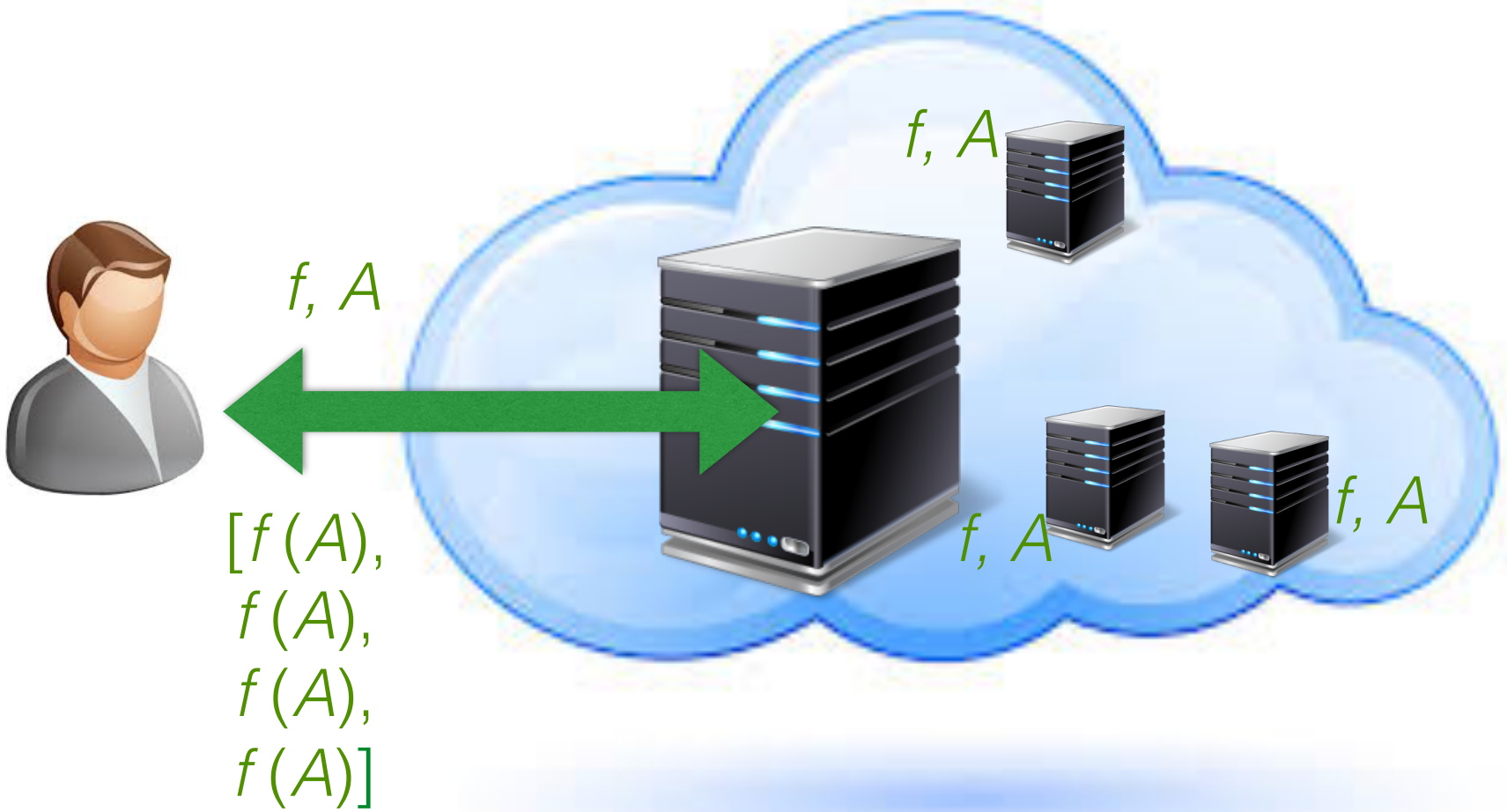
BFT Replication



BFT Replication

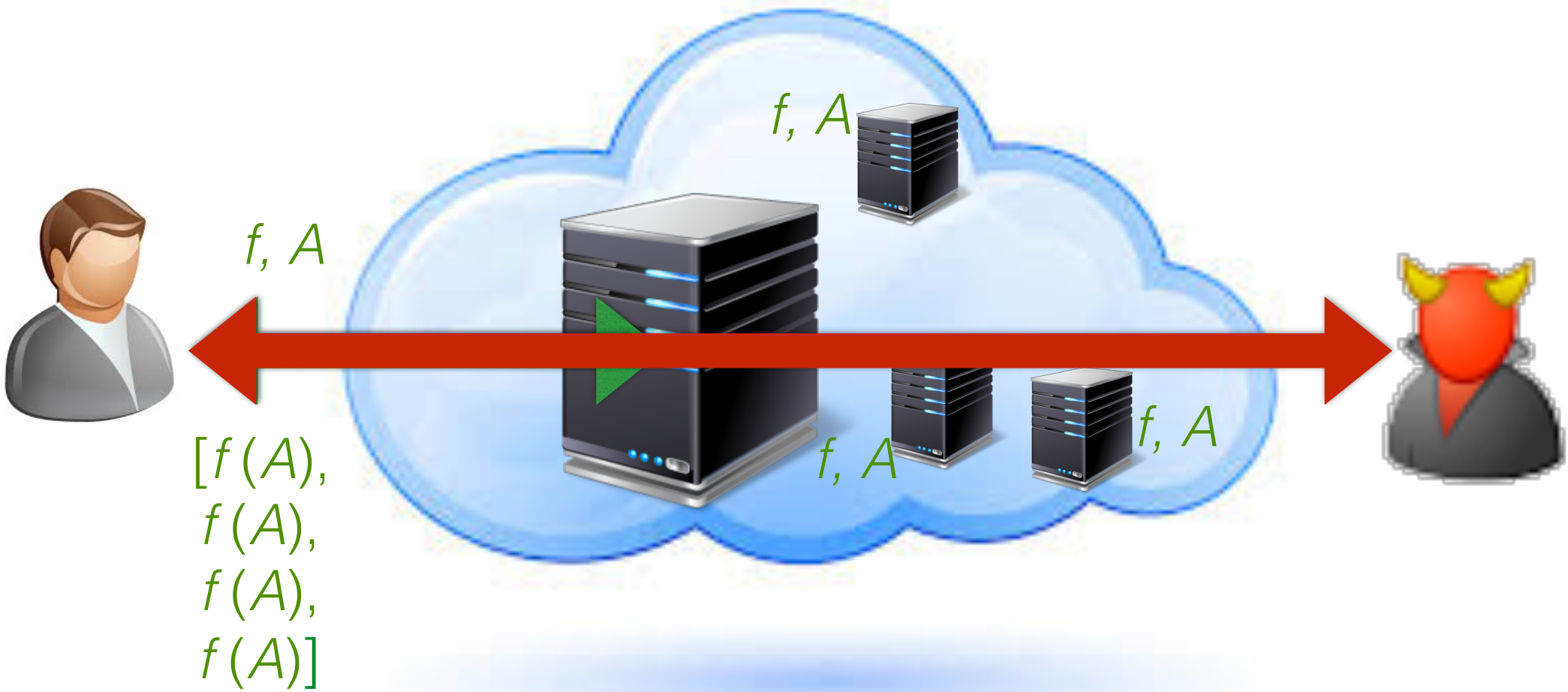


BFT Replication

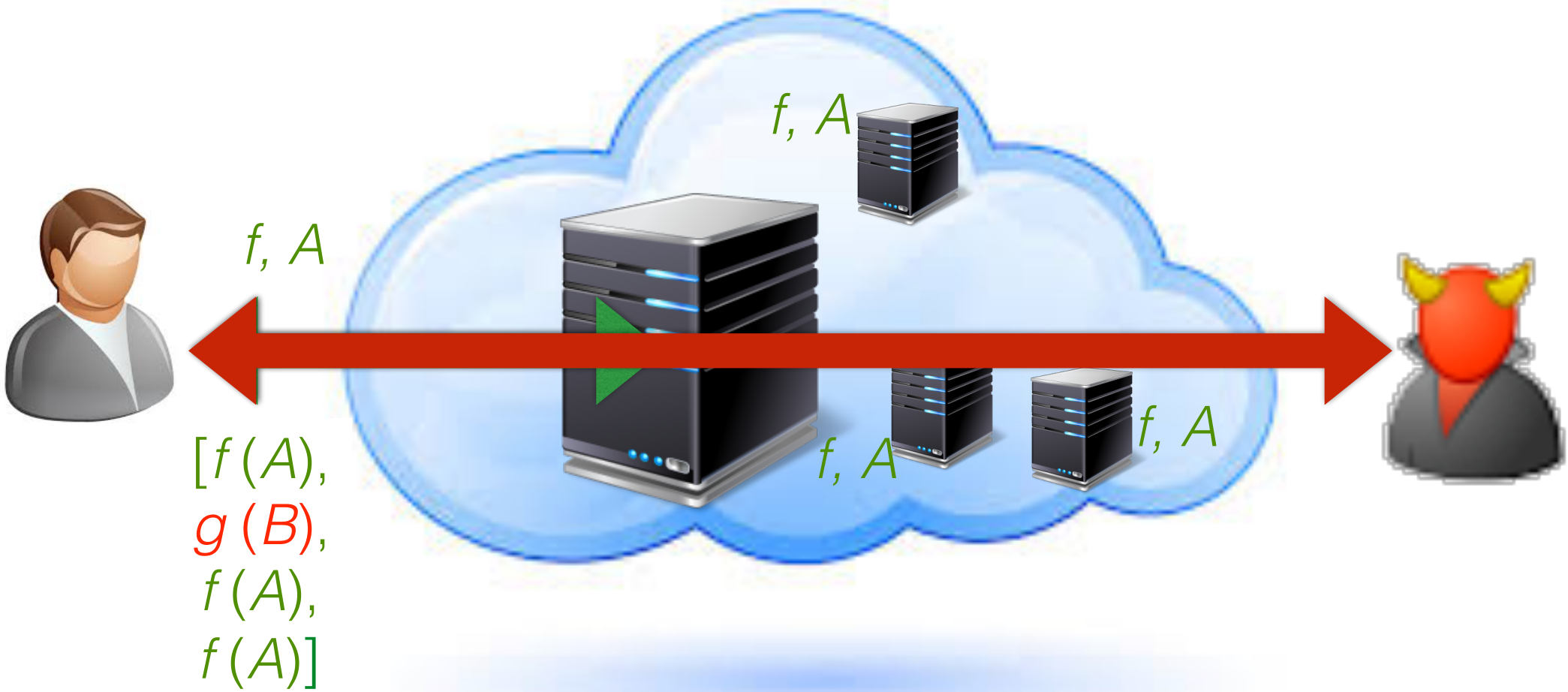


BFT Replication

BFT Replication



BFT Replication



BFT — Hammer or Nail?

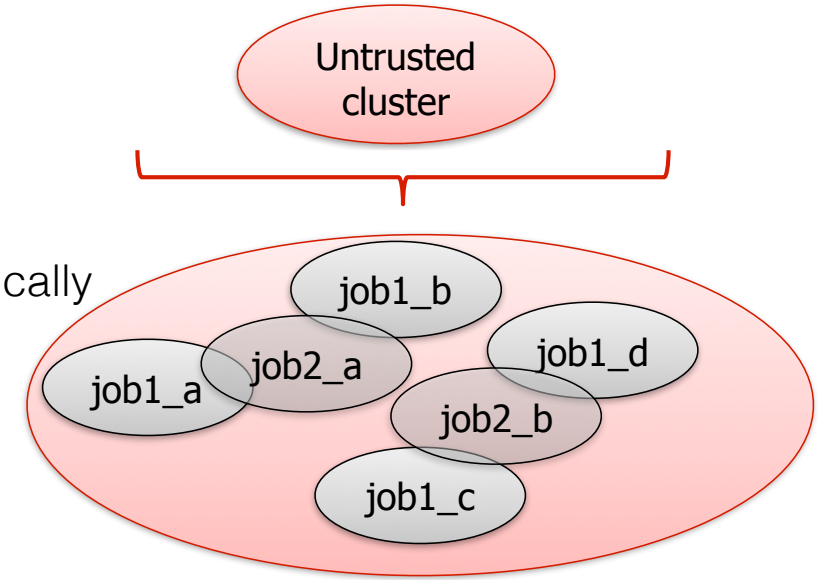
- Why BFT?
  - Masks faulty components (availability, integrity)
  - Identifies faulty components (attribution, isolation)
  - Solution higher up in protocol stack (interoperability, portability)
- Generic challenges of BFT
  - Homogeneity: cloud providers have different OSs, OS versions/images, ASLR standard
  - Non-determinism: largely avoided by deterministic parallelization
  - Agreement: only one “client”

Specific Challenges

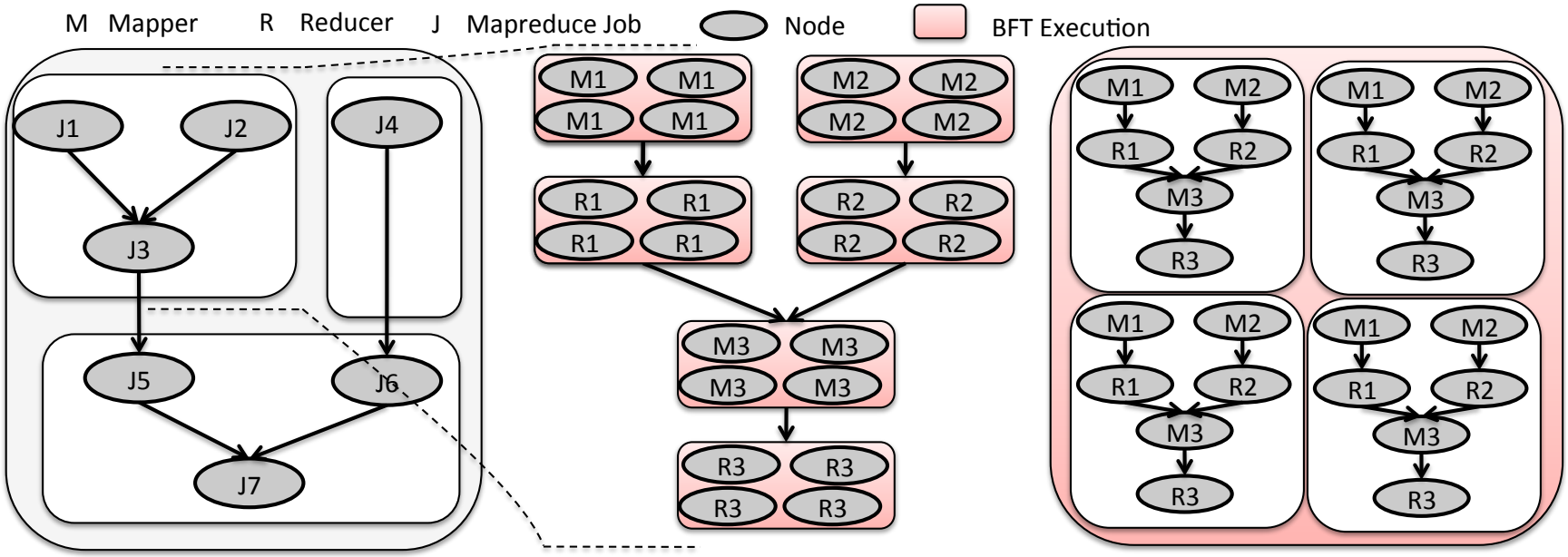
- No monolithic server — each job runs on multiple nodes
  - Cf. Zyzzyva [Kotla et al.;SOSP'07], Upright [Clement et al.;SOSP'09]
- Tasks may spawn multiple sub-tasks
- Size of data too high for naive agreement

ClusterBFT [Stephen&Eugster;Middleware'13]

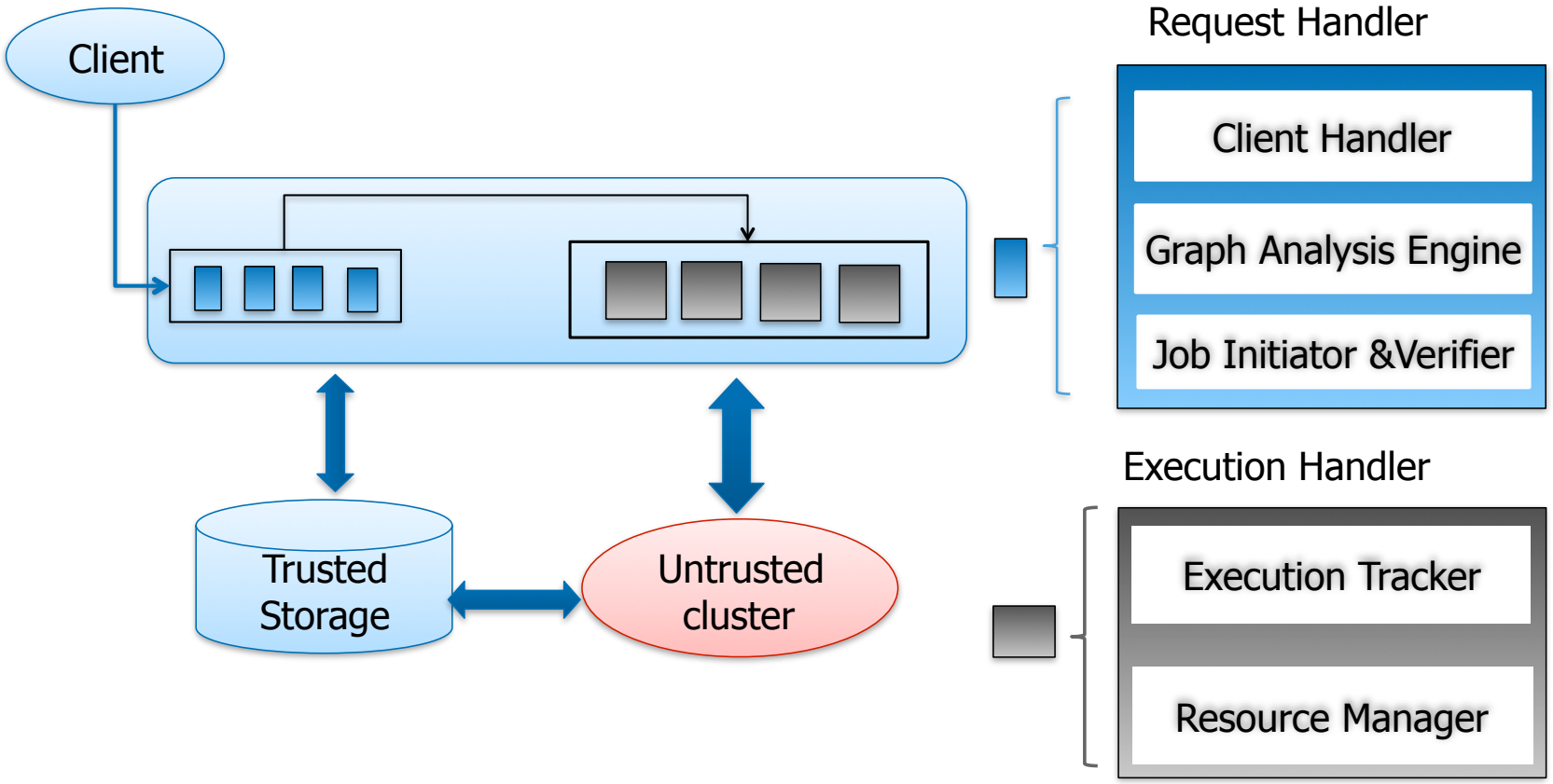
- Variable granularity replication
  - Replication level configurable
  - Trade overhead vs. guarantees
- Streaming and lazy validation
  - No need to wait for (2)*f*+1
  - Compare in background, continue optimistically
- Separation of duty with slim trust base
    - Untrusted nodes compute
    - Trusted nodes control
- Smart replica set overlapping
  - Restore accuracy



Intuition


M Mapper
R Reducer
J Mapreduce Job
Node
BFT Execution
J1
J2
J3
J4
J5
J6
J7
M1
M2
R1
R2
M3
R3

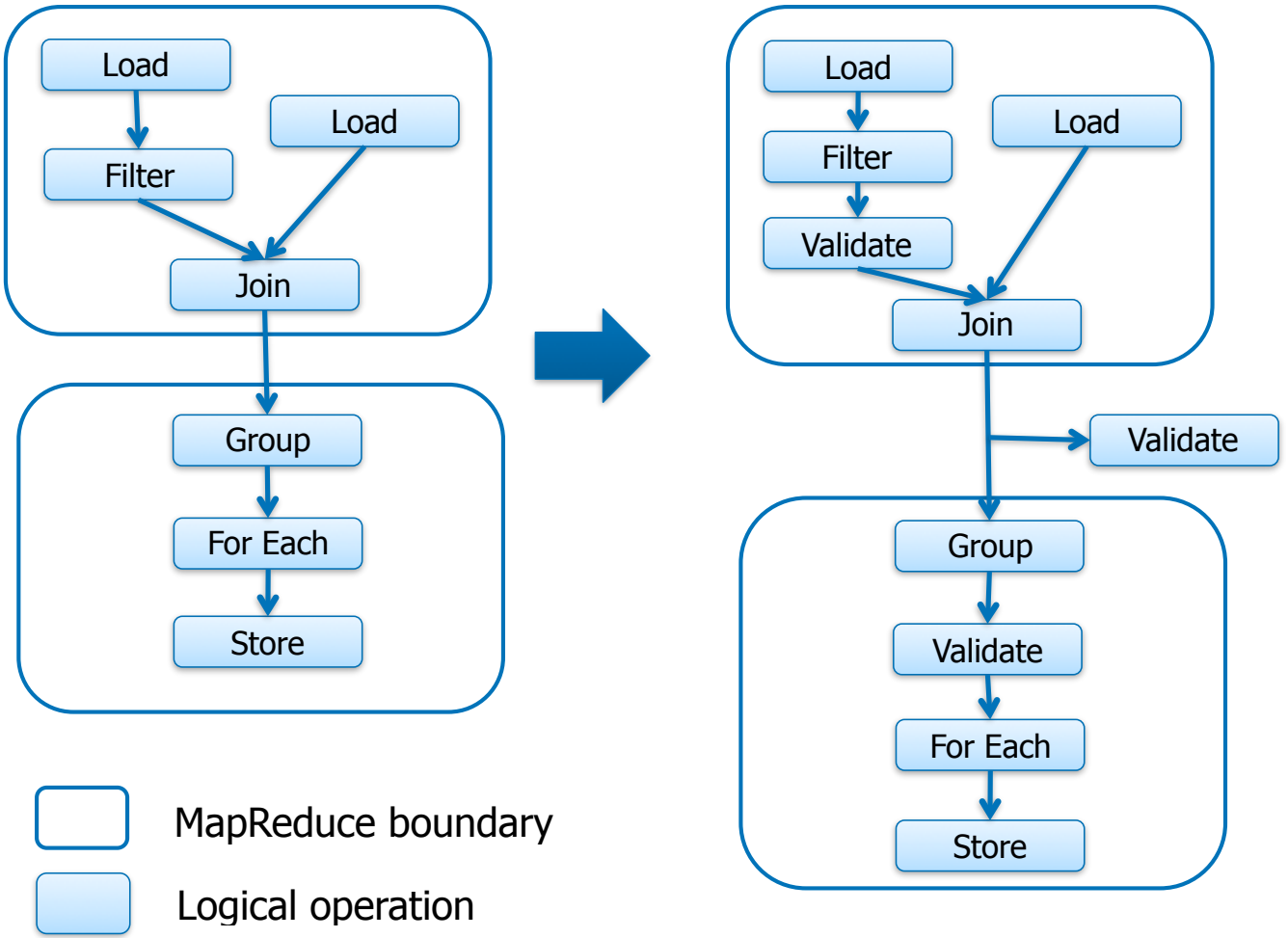
Architecture



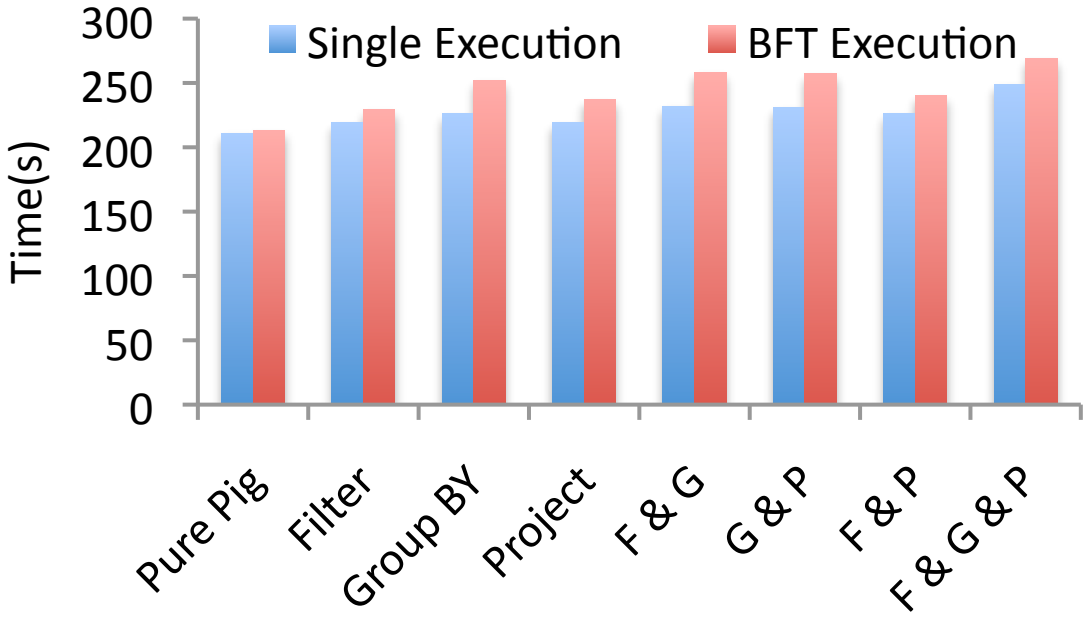
Components

- *Request Handler*
  - Creates logical graph
  - Automatically instruments and embeds validation
  - Sets up inter and/or intra MR validation points
  - Submits job to MR engine (modified Hadoop)
- *Execution Handler*
  - Keeps track of execution progress
  - Ensures cluster overlap
  - Ensures task replicas do not overlap

Illustration



Evaluation



- 11 node cluster (10 data nodes + 1 name node / job tracker)
- Twitter dataset
- Pig script counting number of followers for each user

Outline

- Context
- Availability and integrity in big data analytics
- **Privacy in big data analytics**
- Conclusions and outlook

Problem Statement

- How to ensure that data does not leak in the face of tampering?
  - Intruders, internal threats
- Holy grail — fully homomorphic encryption (FHE)
  - Prohibitive costs in general case
  - Fine-print in expressiveness

Fully Homomorphic Encryption

Fully Homomorphic Encryption

A

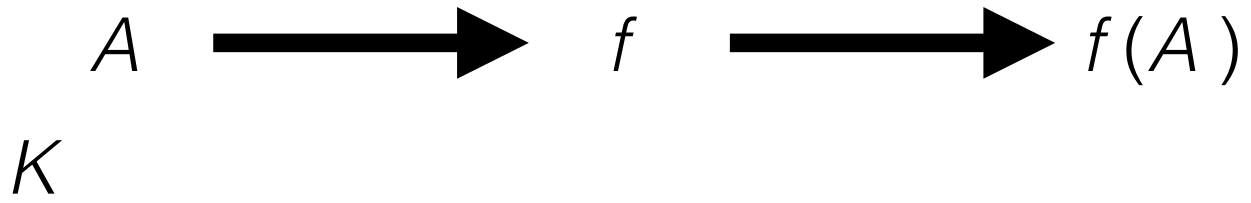
Fully Homomorphic Encryption

Fully Homomorphic Encryption

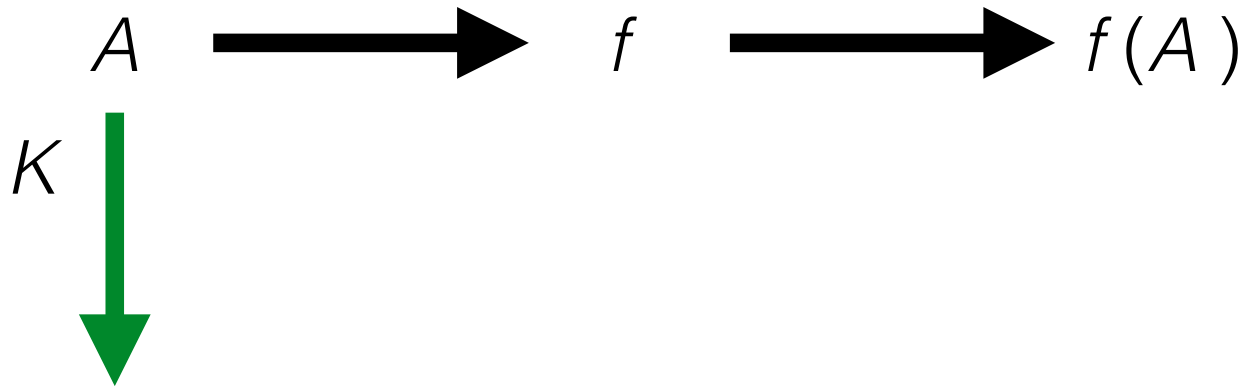
Fully Homomorphic Encryption

Fully Homomorphic Encryption

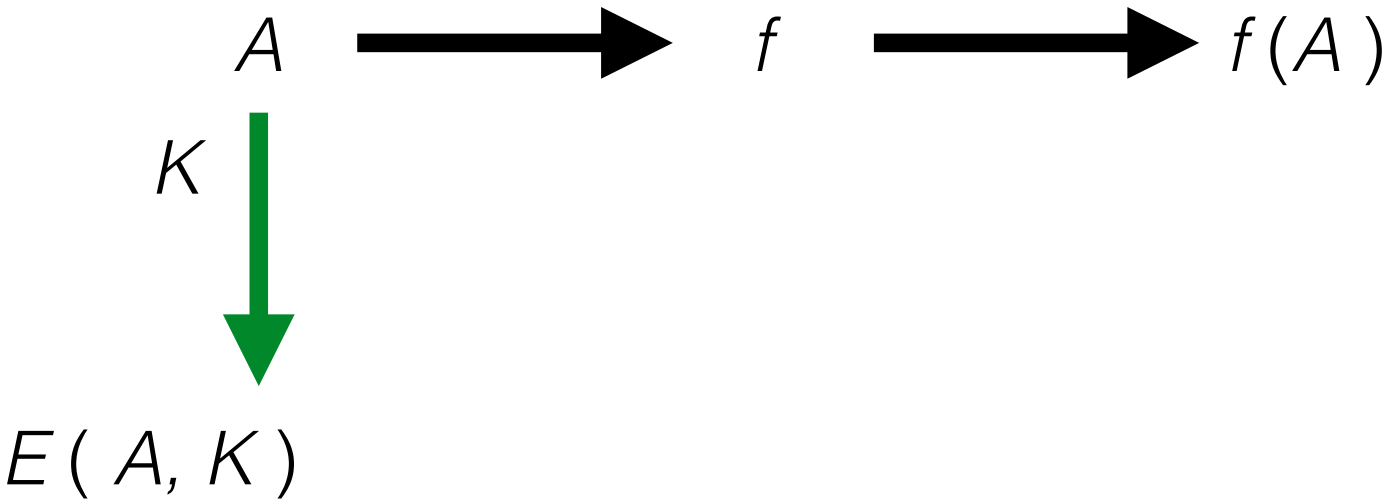
Fully Homomorphic Encryption



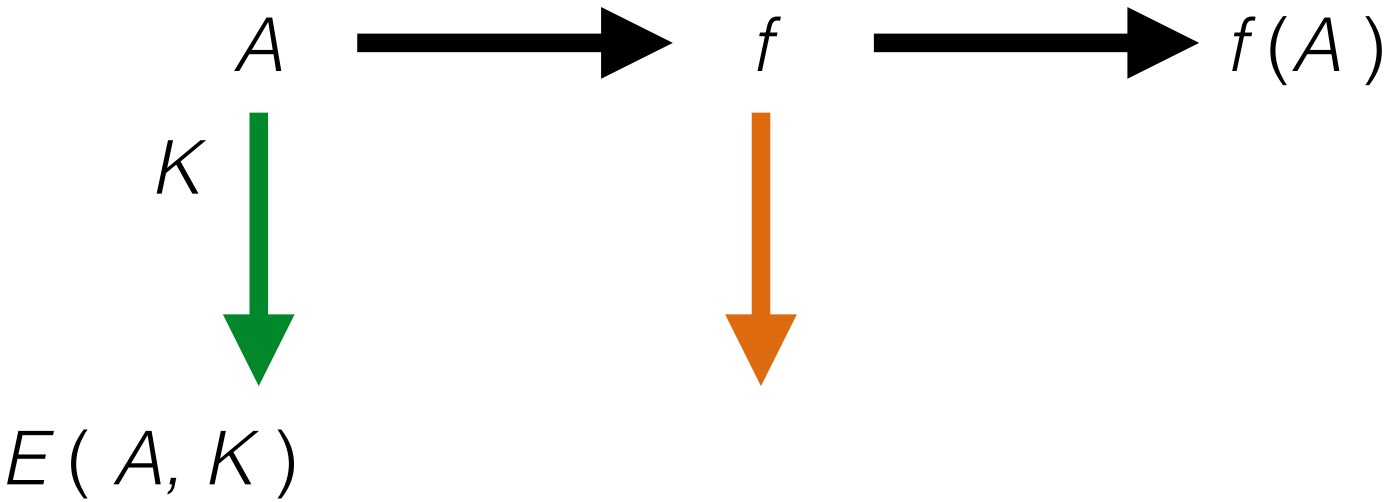
Fully Homomorphic Encryption



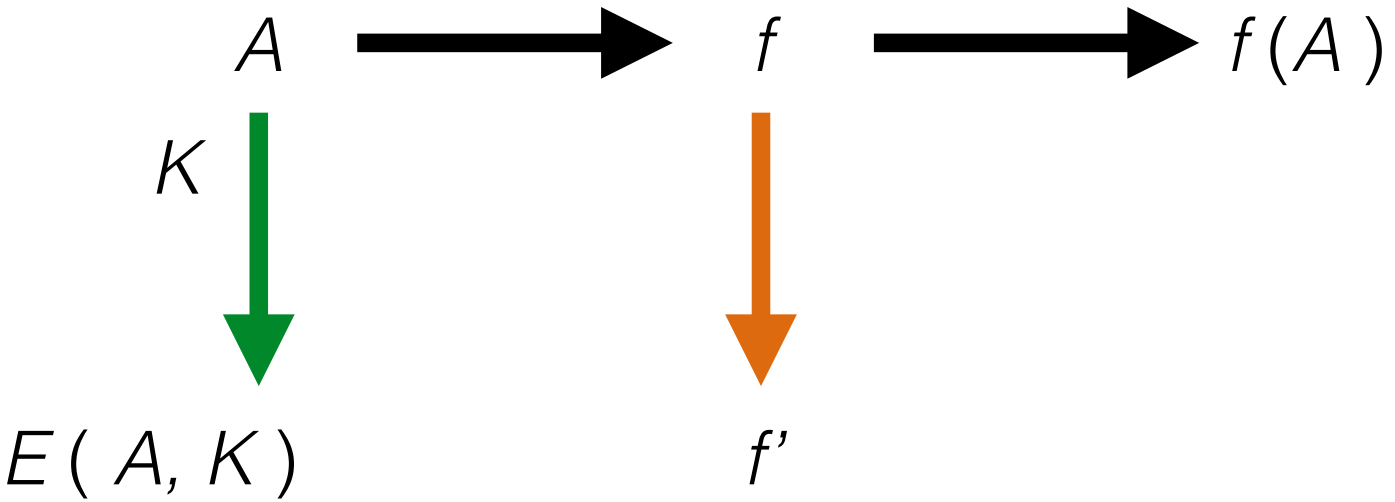
Fully Homomorphic Encryption



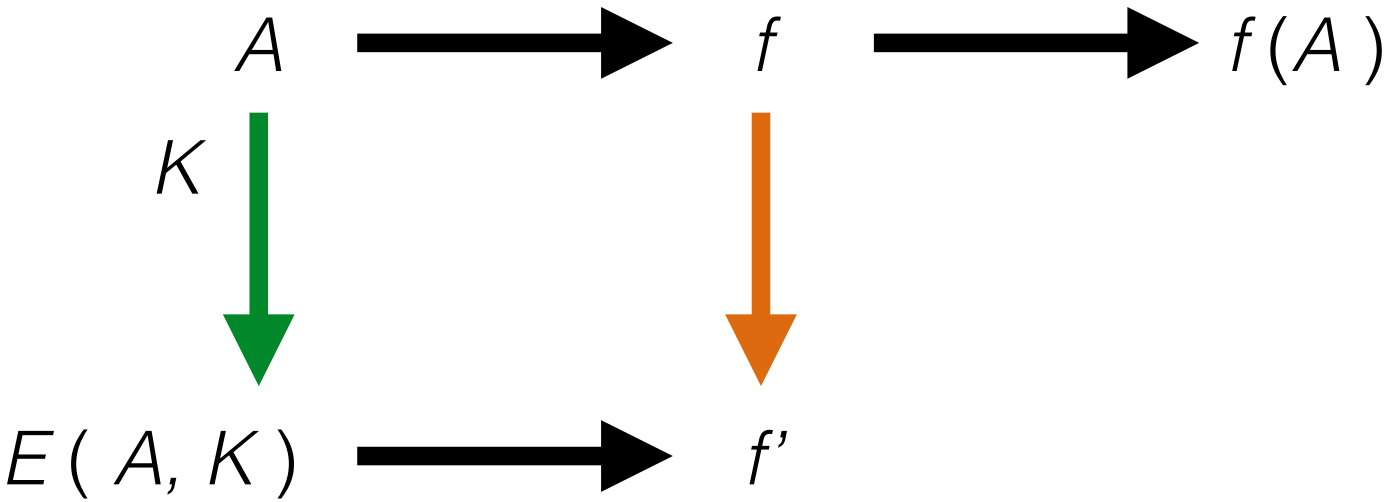
Fully Homomorphic Encryption



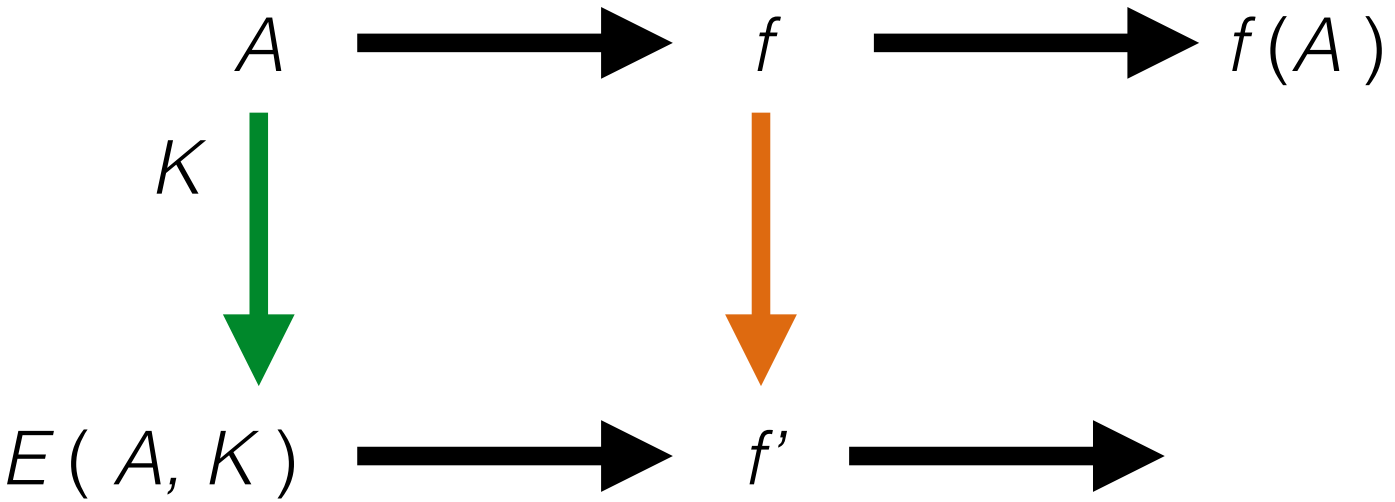
Fully Homomorphic Encryption



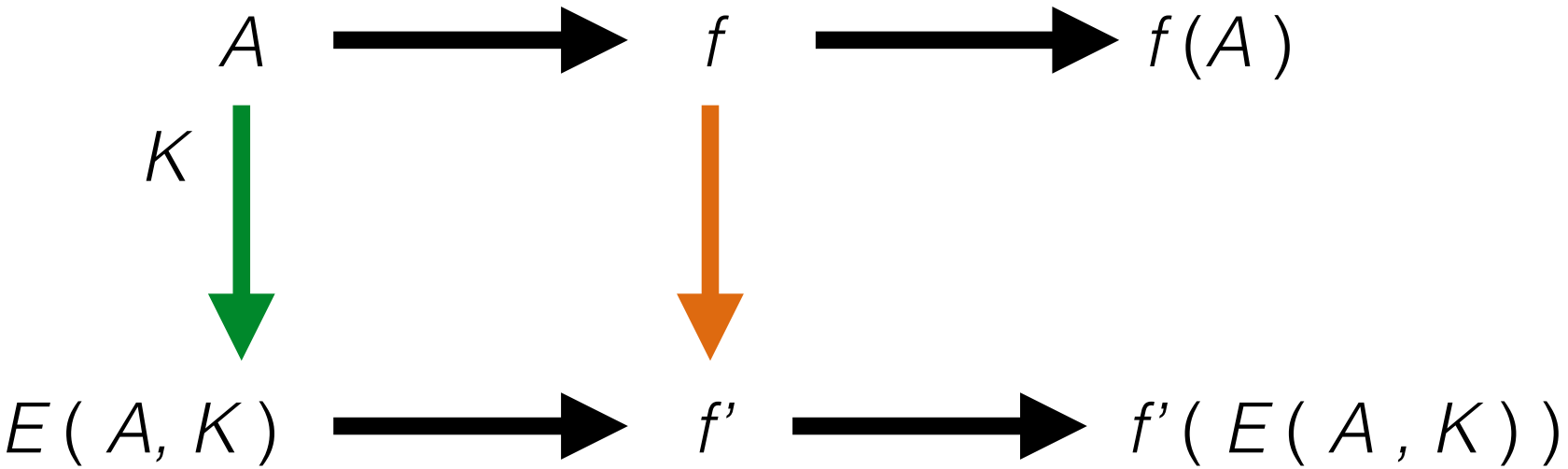
Fully Homomorphic Encryption



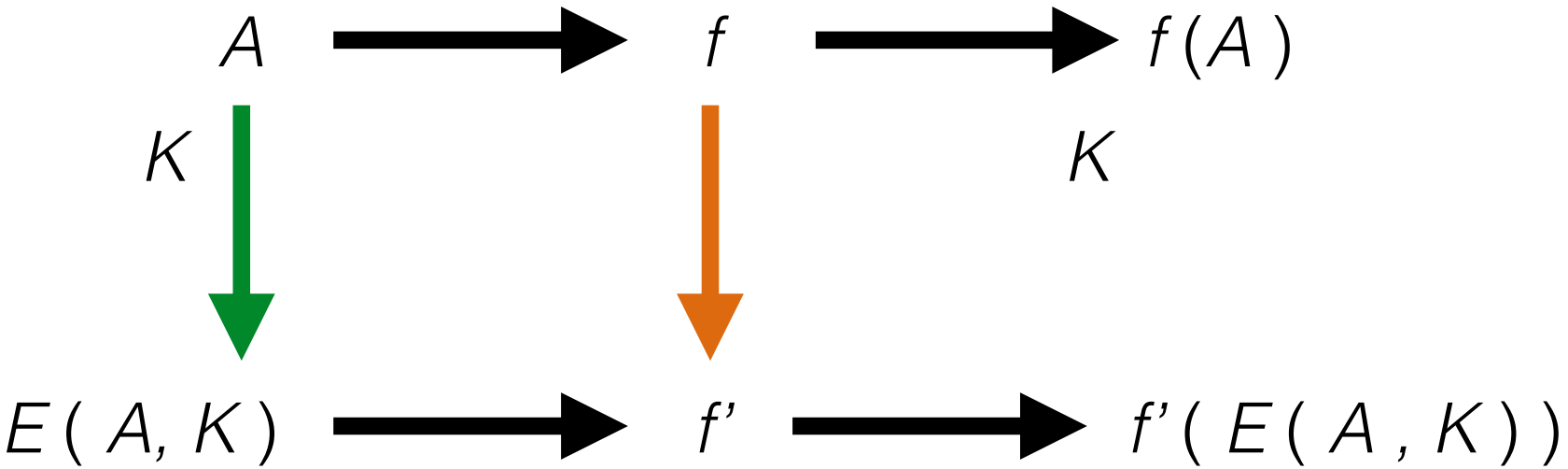
Fully Homomorphic Encryption



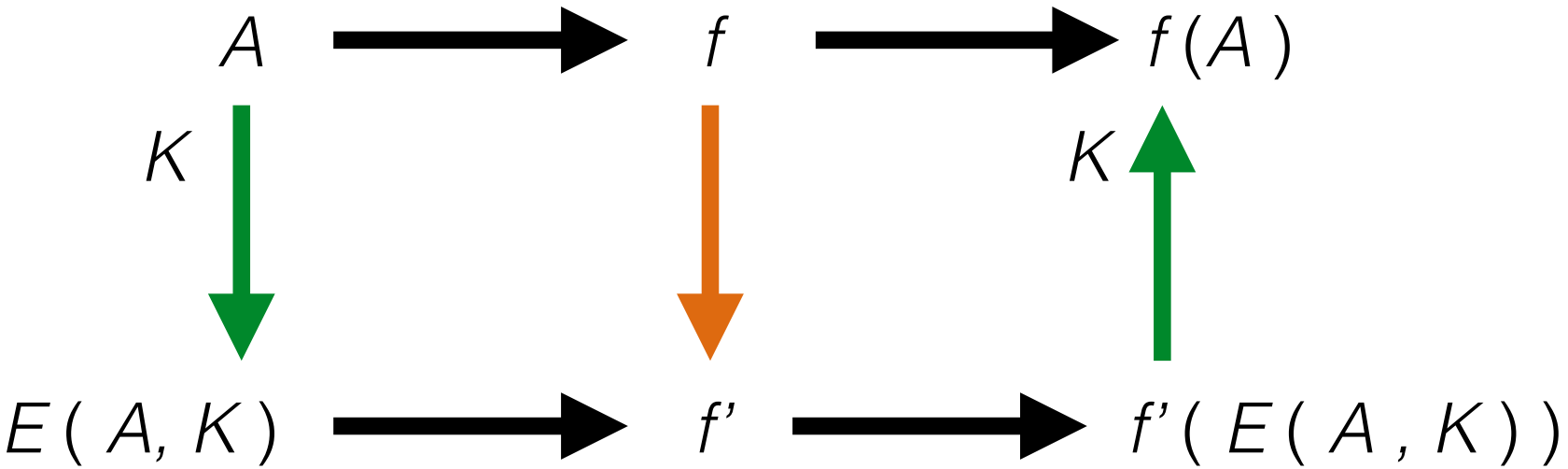
Fully Homomorphic Encryption



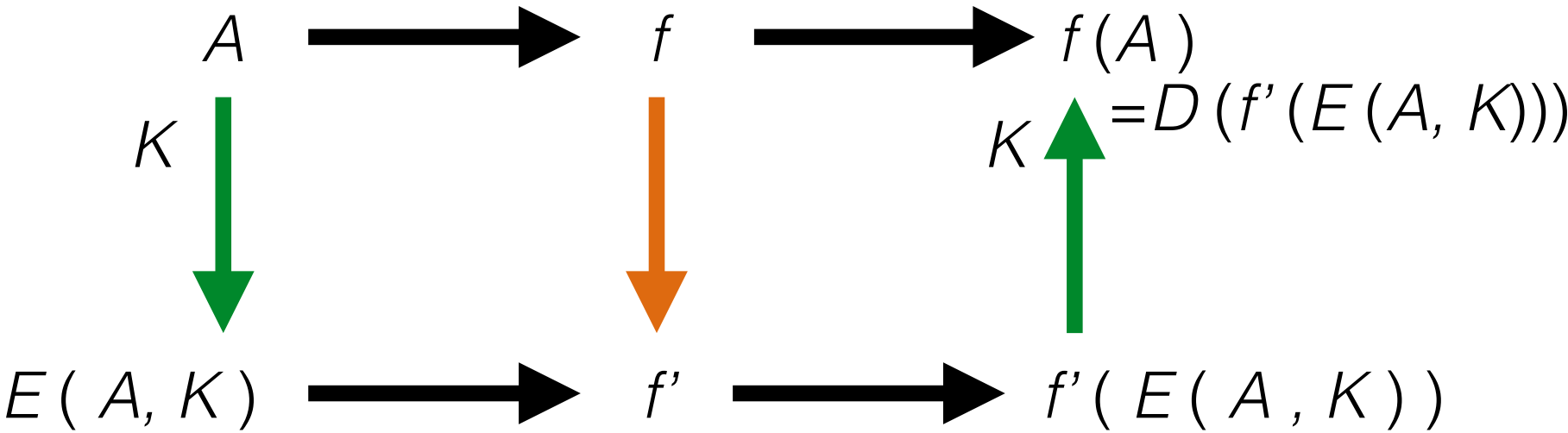
Fully Homomorphic Encryption



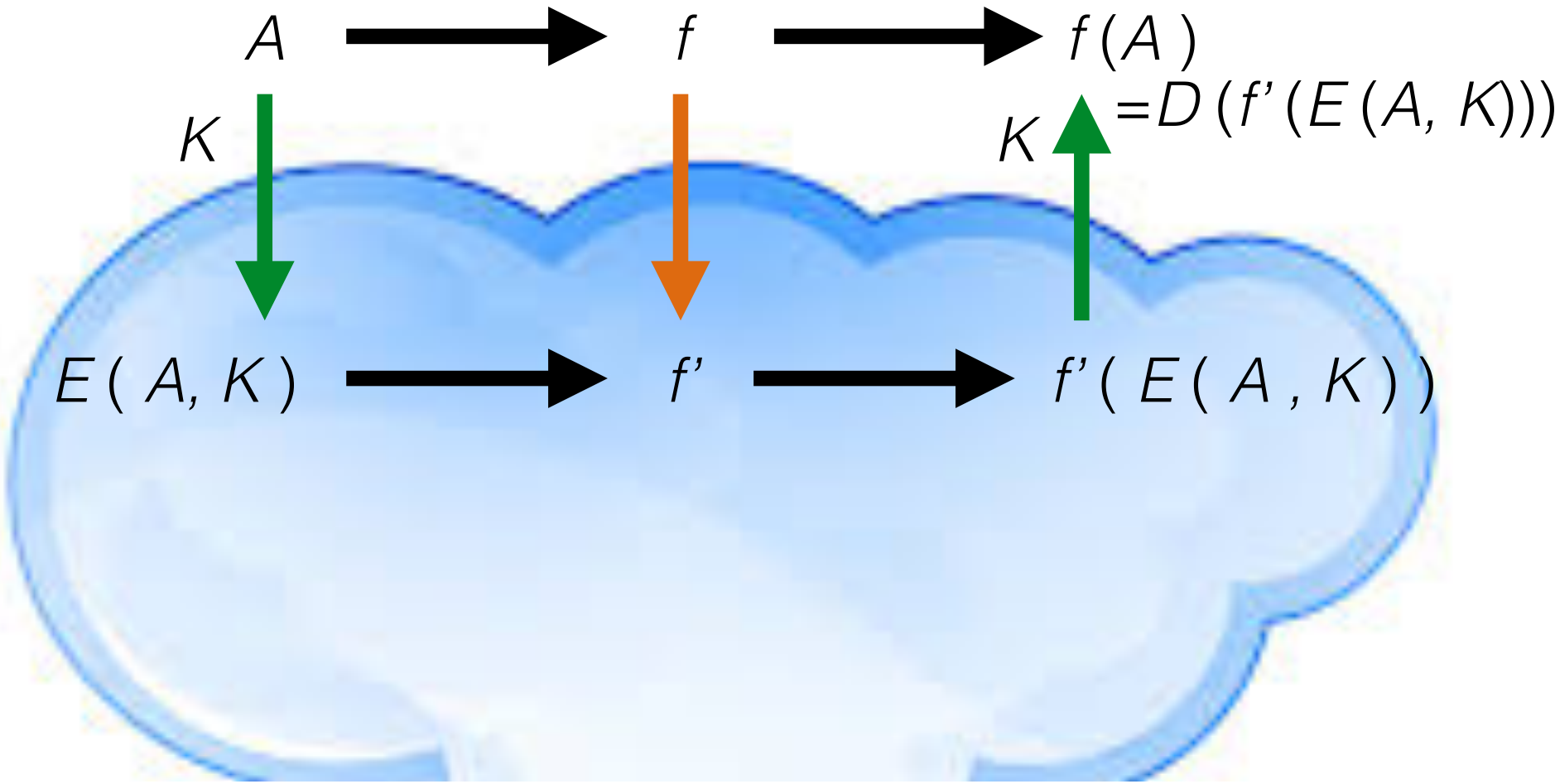
Fully Homomorphic Encryption



Fully Homomorphic Encryption



Fully Homomorphic Encryption



Partially Homomorphic Encryption (PHE)

- Some crypto systems can perform certain operations "under encryption",e.g.,
  - Paillier [Paillier;EuroCrypt'99] ▶ AHE: $\exists \oplus$ s.t. $D(E(x_1) \oplus E(x_2)) = x_1 + x_2$
  - Unpadded RSA [Rivest et al.;CACM'78], ElGamal [ElGamal;ToIT'86]

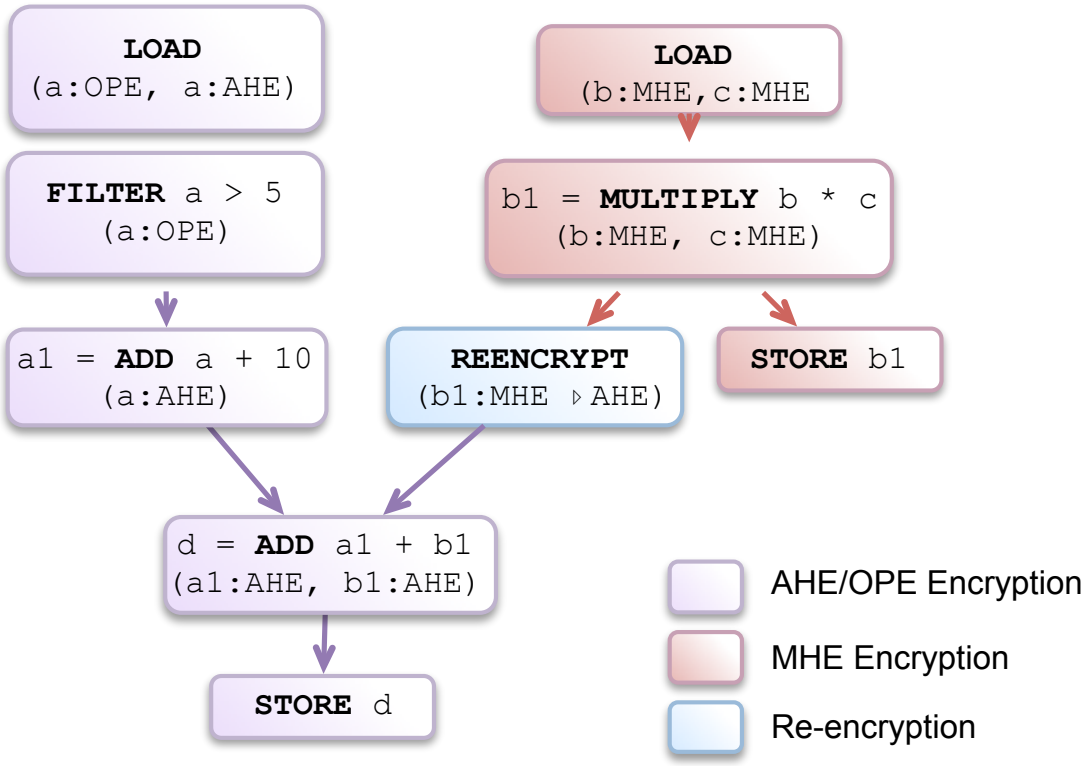
    ▶ MHE: $\exists \circledast$ s.t. $D(E(x_1) \circledast E(x_2)) = x_1 * x_2$
  - DET (=), OPE (<), SRCH

Privacy in Big Data Analytics

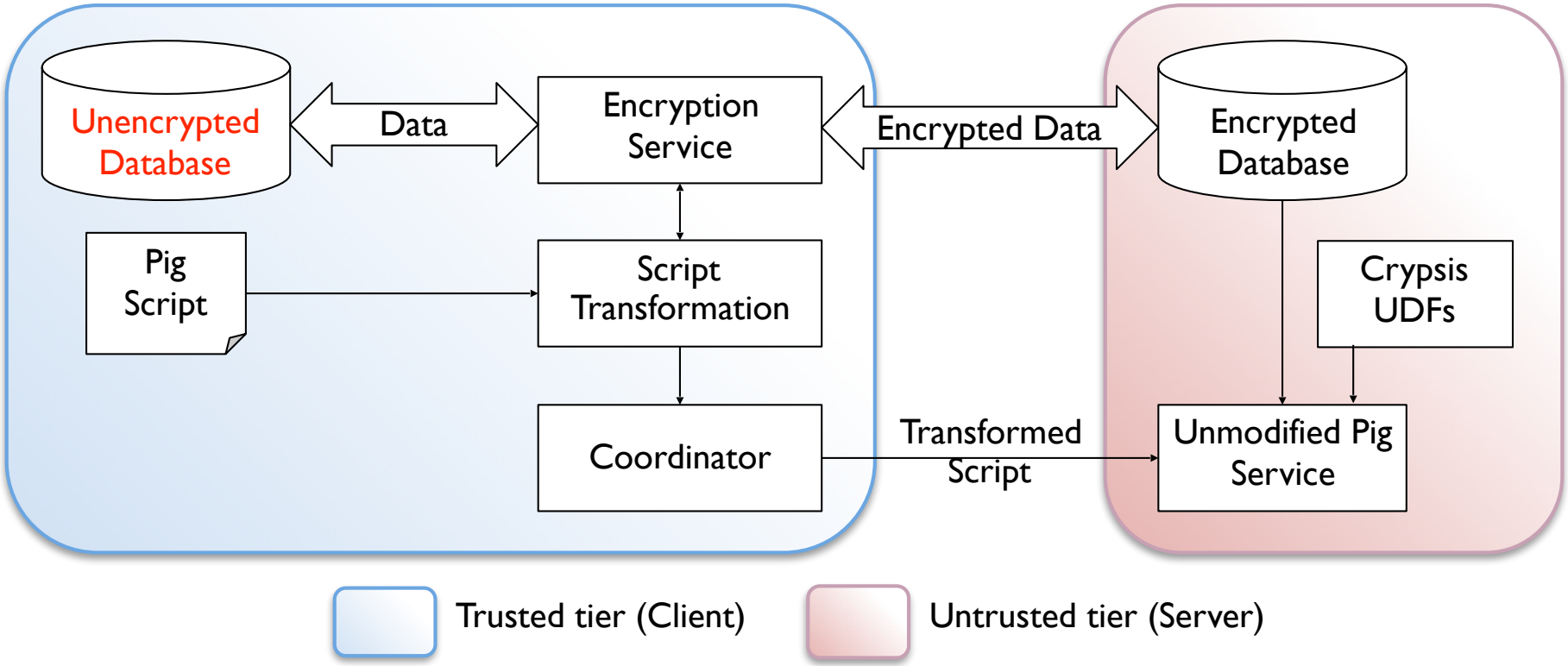
[Stephen et al.;HotCloud'14],[Stephen et al.;ASE'14]

- Intuition
  - Can use multiple cryptosystems side-by-side
  - Leverage parallelization (vs CryptDB [Popa et al.;CACM'12], Monomi [Pu et al.;PVLDB'13], Talos [Shafagh et al.;SenSys'15])
  - Client-side completion or re-encryption (vs MrCrypt [Lesani et al.;OOPSLA'13])

Crypsis Intuition

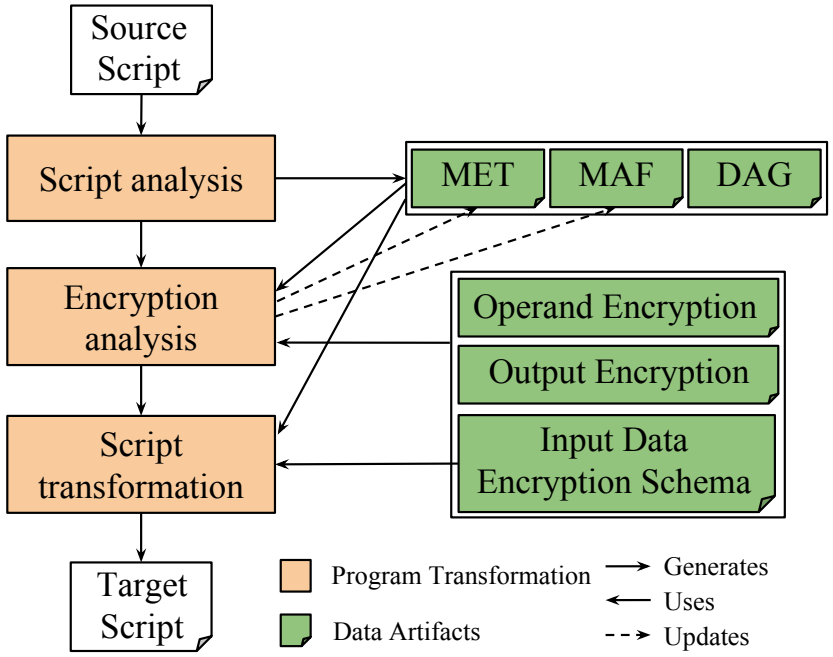


Architecture Overview



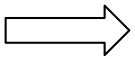
Script Transformation

- Generate d*ata-flow graph* (DFG)
  - Nodes are relations (**`LOAD`**, **`FOREACH`**, …)
  - Edges are data-flow between operators
- Generate *map of expression trees* (MET)
  - Contains all expressions
  - Keys are used to assign expressions to DFG
- Generate *set of annotated fields* (SAF)
  - One entry for each ⟨*relation*, *field* ⟩ of script
  - ⟨*relation*, *field* ⟩, parent, available encryptions, required encryptions
  - Get available encryptions from lineage of field, required encryptions using MET



Example Transformation

```
A = LOAD 'input1' AS
      (a0, a1);
B = LOAD 'input2' AS (x0);
C = FILTER A BY a0 > 10;
D = GROUP C BY a1;
E = FOREACH D GENERATE group AS
       b0, SUM(C.a0) AS b1;
F = JOIN E BY b0, B BY x0;
STORE F INTO 'out';
```

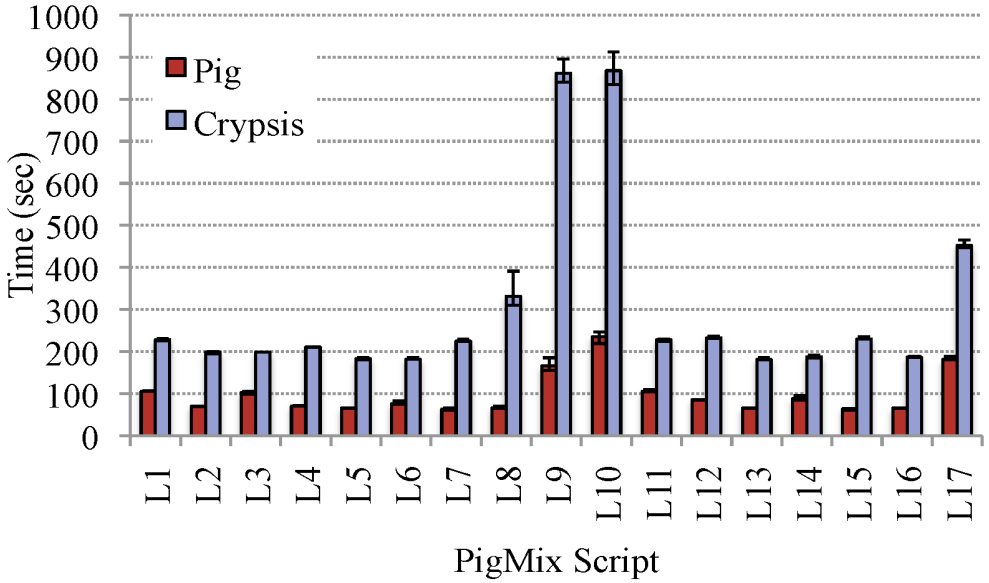


```
A = LOAD 'enc_input1' AS
       (a0_ope, a0_ah, a1_det);
B = LOAD 'enc_input2' AS (x0_det);
C = FILTER A BY a0_ope > OPE(10);
D = GROUP C BY a1_det;
E = FOREACH D GENERATE group AS
       b0, ENCSUM(C.a0_ah) AS b1;
F = JOIN E BY b0, B BY x0_det;
STORE F INTO 'out';
```

Simple Re-Encryption

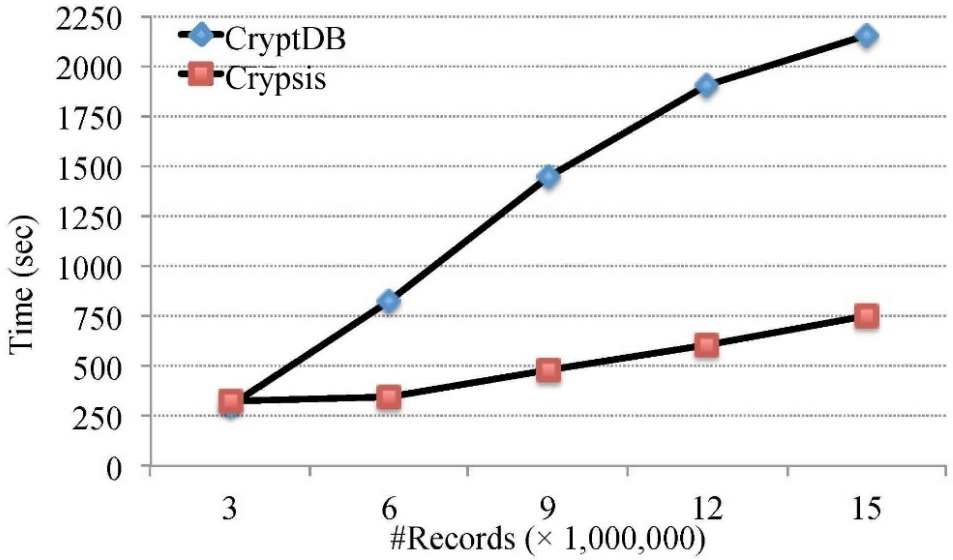
- Re-encryption required when
  - Required encryption unavailable
  - Incompatible operations, e.g., addition followed by multiplication
- Re-encryption conceptual
  - Can continue on client side
- 17 PigMix II benchmarks
  - Only script 8 requires re-encyption (averaging)
  - 1 script gets by with same attribute in several cryptosystems

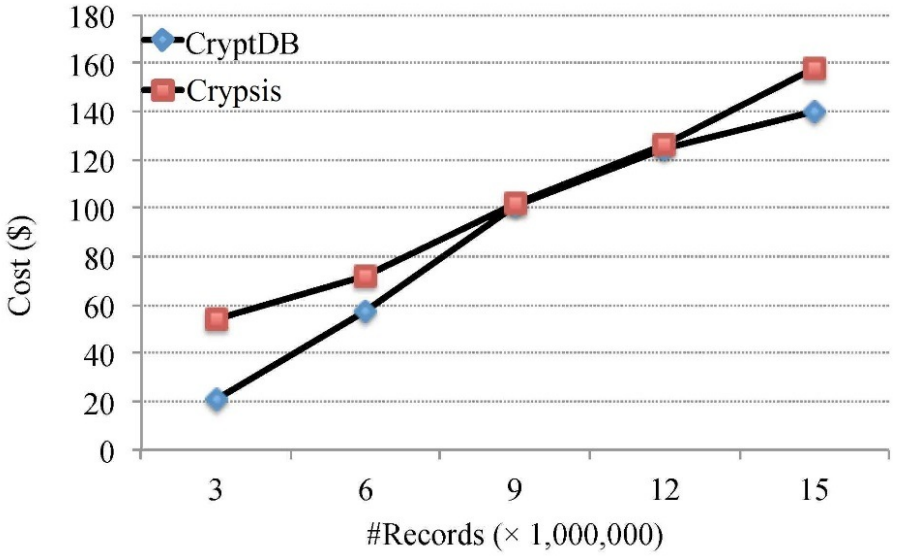
Evaluation (PigMix II)



- 11 EC2 large instances (2 CPUs, 3.75BB RAM)
- 5 GB data
- Average of 3x overhead (FHE can lead to 1000…s)

Crypsis Comparison





- 3 EC2 medium instances
- ~3x faster for 15Mio records
- Similar overall cost

as fast as cryptdb when only on 1 node, because CPU dominated

Limitations

- Drawback of PHE based solutions
  - Multiple users must share same key
  - Dealing with multiple domains adds complexity, e.g., garbled circuits, re-encryption
- Transparent solutions rarely yield optimal performance
  - Encrypting everything is costly
- Data specification
  - Only owners of data can know which parts of their datasets can be shared, and to what extent
- Queries
  - Can be expressed only if structure of data and accessibility constraints are known

PECCARY (**P**rivacy-preserving **E**fficient **C**loud-based **C**omputation **a**pplying **R**e-Encr**y**ption)

- Specification language
  - Structure of data
  - Constraints on visibility of attributes
  - Relationships between attributes
  - Precise model of data types (e.g., 100 > `value` > 0)
  - E.g., data owners specify which (aggregation) operations are supported on which attributes

- Query language/compilation
  - Allow access requirements to be derived
  - Allow feasibility to be matched with specification
  - Optimization, especially reduction of re-encryption

Secure Data Types (SDTs)

- Sensitivity levels
  - **`HIGH, LOW, NONE`**
  - Accounts for different security guarantees offered by crypto systems, avoids unnecessary overhead
- Ranges and precision for data
  - Positive/negative numbers
  - Fixed ranges e.g. 0-100
  - Decimal points for floats to preserve
- Enumerations
  - Fixed set of values, e.g. **`enum{`**`EUROPE, ASIA, AMERICA, AFRICA`**`}`**
- Composite data types
  - Values containing multiple parts, e.g. country code&local in phone #, year&month&day in date

```
composite[(4:int[+])-(2:int[range(1-12)])]
```

SDT Example

```
DEFINE Lineitem AS {
  orderkey:          long[+],
  linenumber:        long[+, unique],
  tax:               double[2]<NONE>,
  price:             double[2]<HIGH>,
  shipdate:          composite[(4:int[+])-(2:int[range(1-12)])-
                     (2:int[range(1-31)])],
  shipinstruct:      enum{IN_PERSON, COLLECT_COD, RETURN, NONE},
  comment:           chararray
  …
}
```

Compilation Techniqes

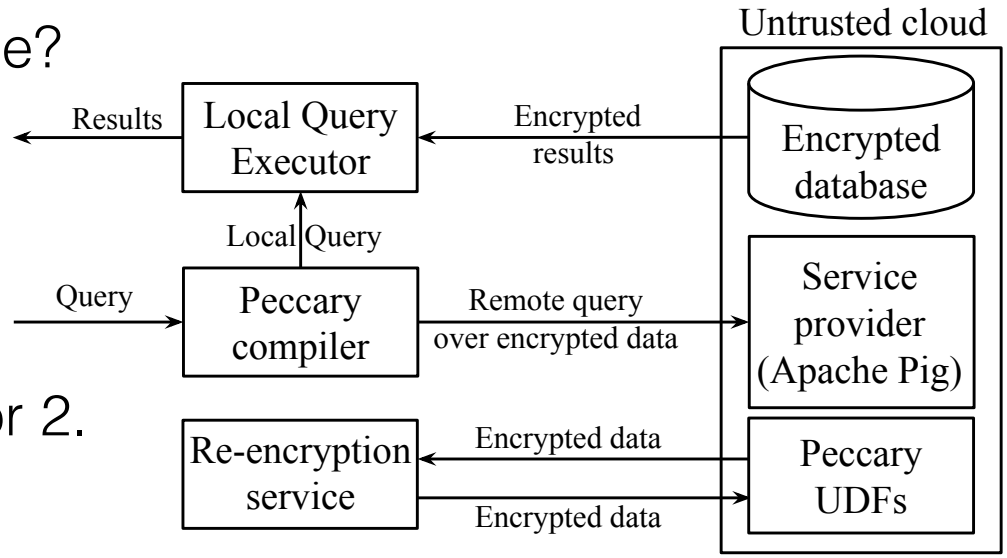
- Expression rewriting, e.g.,
  - `SUBSTRING(shipdate, 0, 4) == `1994' -> shipdate.year == `1994'`
  - `x, y >= 0; x + y > 0 -> x > 0 || y > 0`
  - `((a * b) + c) * d -> (a * b * d) + (c * d)`
- Selective encryption, e.g.,
  - **NONE** attributes
  - `(a+b)*c` with non-sensitive c -> Paillier secondary homomorphic property

Compilation Techniques

- Subexpression elimination, e.g. (TPC-H Q01)
  - `price*(1-discount)` vs `price*(1-discount)*(1+tax)`
  - `SUM, AVG, COUNT` vs `SUM, COUNT; AVG = SUM/COUNT`
- Efficient encryption, e.g.,
  - Enum types with only DET and ORD -> (random) int values
  - OPE -> DET
  - unpadded RSA for AHE and '=='
  - Boneh-Dan-Goh [Boneh et al.;TCC'05] for (multiple) AHE followed by (one) MHE
  - Packing multiple values

Re-Encryption

- Why throw in the towel when you can…
  - … complete computation on the client side?
  - … re-encrypt on the client side?
- Or better yet generalize?
  - Whenever hitting a PHE limit
  - Apply heuristic to choose 1. or 2.
- Plus
  - Caching of DET values
  - Speculative re-encryption for DET

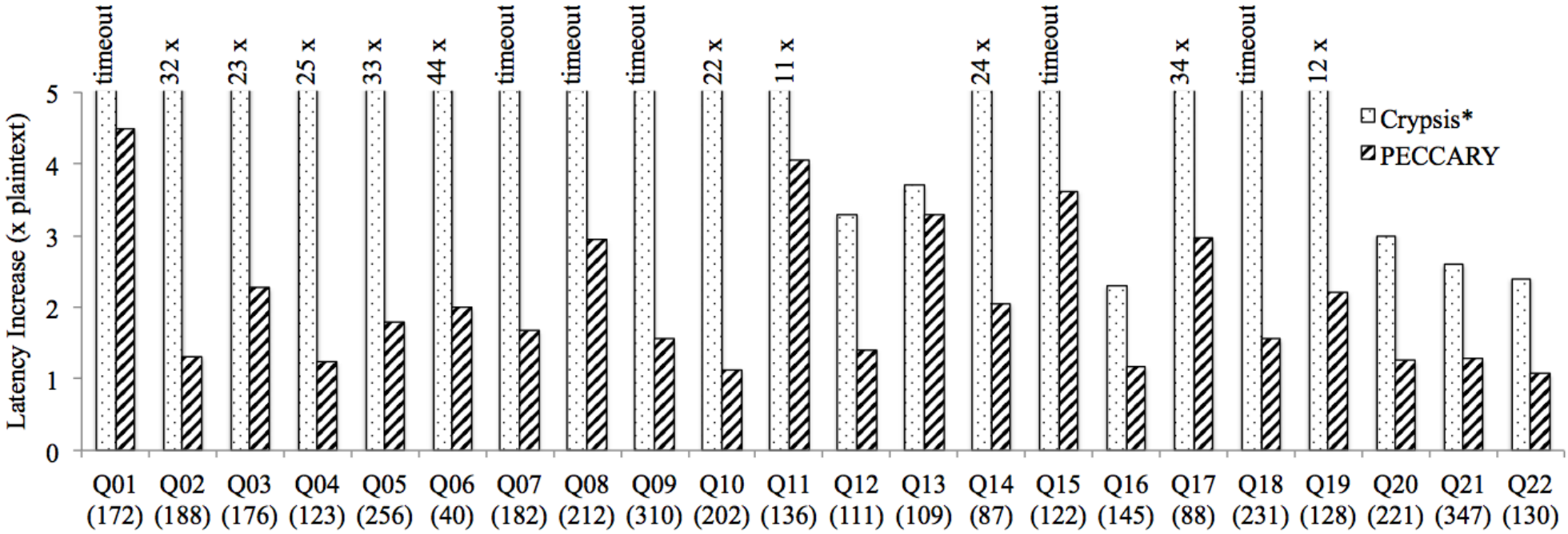


Benefits

	Expression rewriting	Selective encryption	Subexpr. elimination	Efficient encryption	Caching and speculative encryption
Secure data types		✓		✓	
Data range and precision	✓				✓
Enumerated type				✓	
Composite type	✓				

Performance

- TPC-H
  - 10GB plaintext
  - Amazon EC2 (10 xlarge nodes)
  - Only system to execute Q1 and Q15 entirely in the cloud



* Augmented to support (sub)string queries, floating points

Effectiveness

NORTHROP GRUMMAN

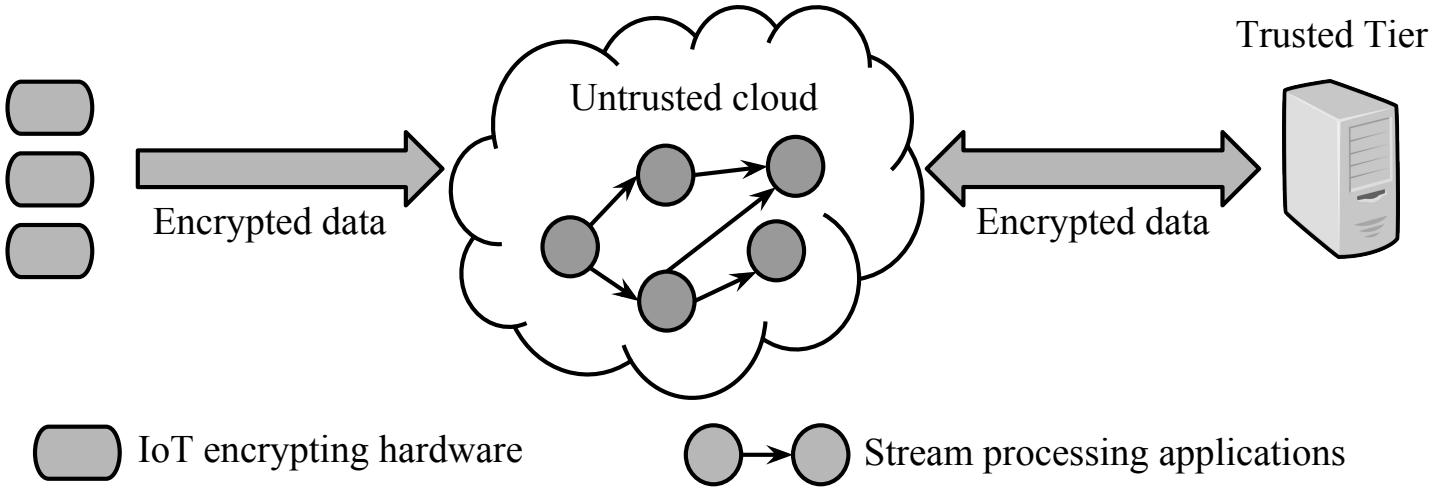
	# of scripts	
	PigMix2	TPC-H
Expression Rewriting	-	1
Selective Encryption	7	14
Efficient Encryption Strategy	2	20
Caching and Speculative Re-encryption	-	6
Subexpression Elimination	-	1

	Crypsis*	PECCARY
# of queries with re-encryption	18	6
Total # of re-encryptions	51	7

* Augmented to support (sub)string queries, floating points

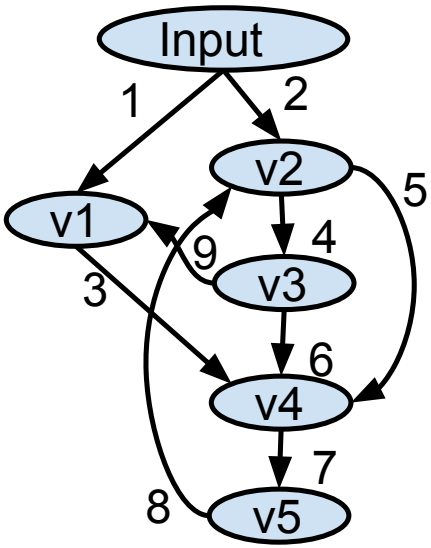
Online Processing

- Same principles (transformation) can be applied online
  - Implemented in Apache Storm (STYX — **s**tream processing with **t**rustworth**y** cloud-based e**x**ecution)
- Need to merge with corresponding optimizations

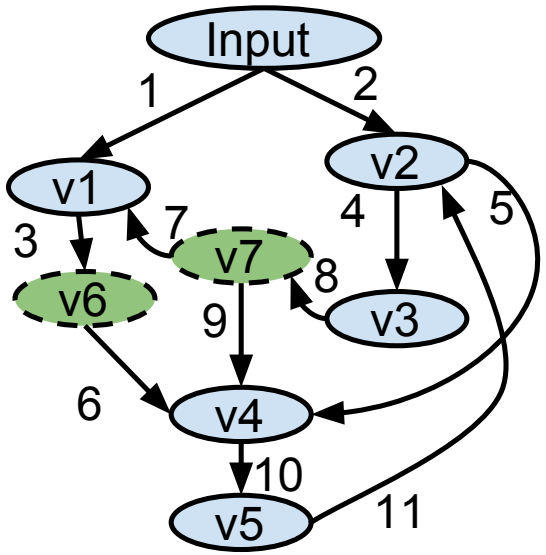


Linear Road Benchmark

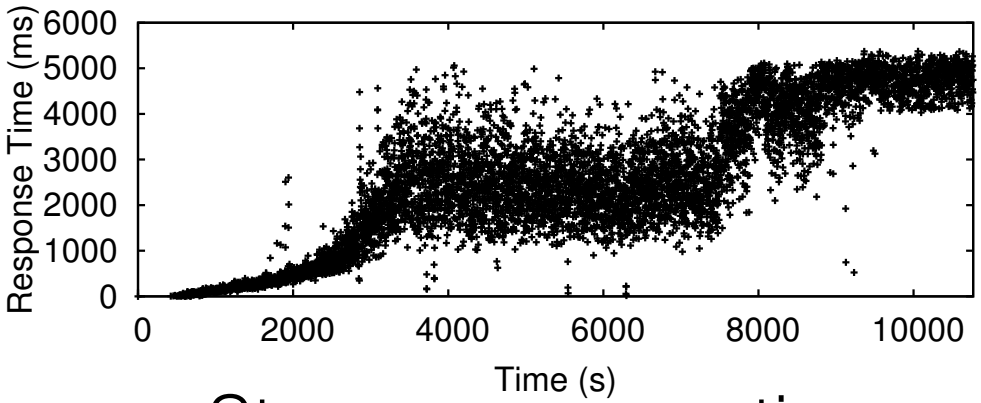
Storm



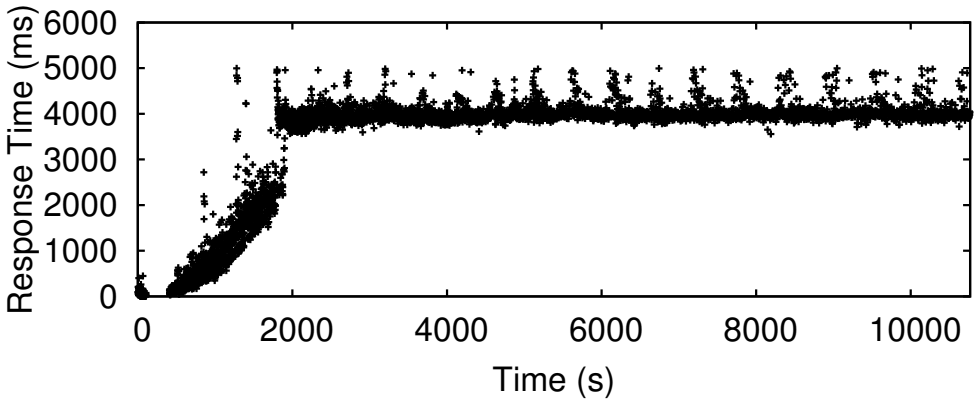
STYX



v6, v7 for re-encr.



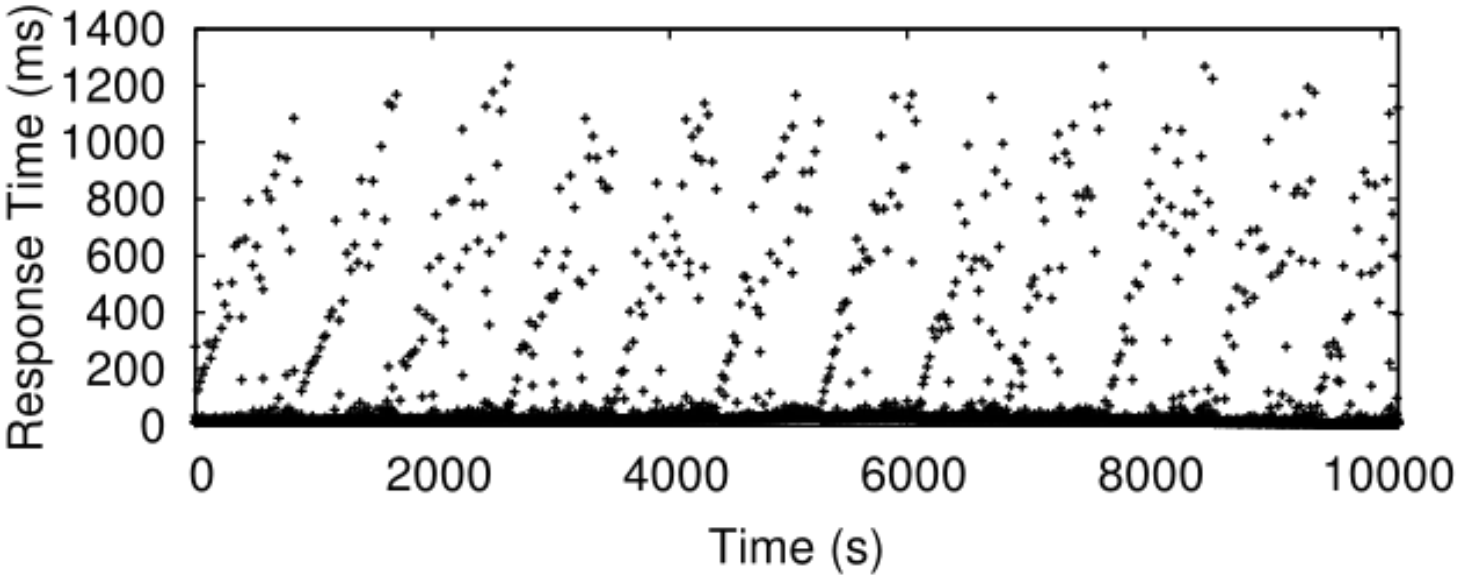
Storm response times



STYX response times

Re-Keying

- Key change
  - New York taxi route data (10G)
  - Application finds the top 10 most frequent routes during the last 30 minutes of taxi servicing
  - Amazon EC2 (9 large nodes)



Outline

- Context
- Availability and integrity in big data analytics
- Privacy in big data analytics
- **Conclusions and outlook**

Conclusions and Outlook

- Cloud security is gigantic topic
  - Big data security subset still huge
  - Many building blocks, e.g., also functional encryption, oblivious RAM, garbled circuits
- Much work left for described approaches, e.g.,
  - BFT
    - Peer-based trust management and attribution
  - PHE
    - Heuristics for re-encryption vs. client side completion

Conclusions and Outlook

- Much potential in hybrid techniques
  - E.g. garbled circuits + PHE, HW extensions (Intel SGX) + …?
  - Also PL + SE + DS + Crypto + NW + OS + HW + …

Conclusions and Outlook

- Much potential in [illegible]ques
  - E.g. garbled circuits + PHE, HW extensions (Intel SGX) + …?
  - Also PL + SE + DS + Crypto + NW + OS + HW + …