Secure Data Analytics in Apache Spark

with Fine-grained Policy Enforcement and Isolated Execution

Byeongwook Kim*, **Jaewon Hur***,
Adil Ahmad, and Byoungyoung Lee

서 울 대 학 교
SEOUL NATIONAL UNIVERSITY

Arizona State University

Cloud-based Spark: Collaborative Big Data Analytics

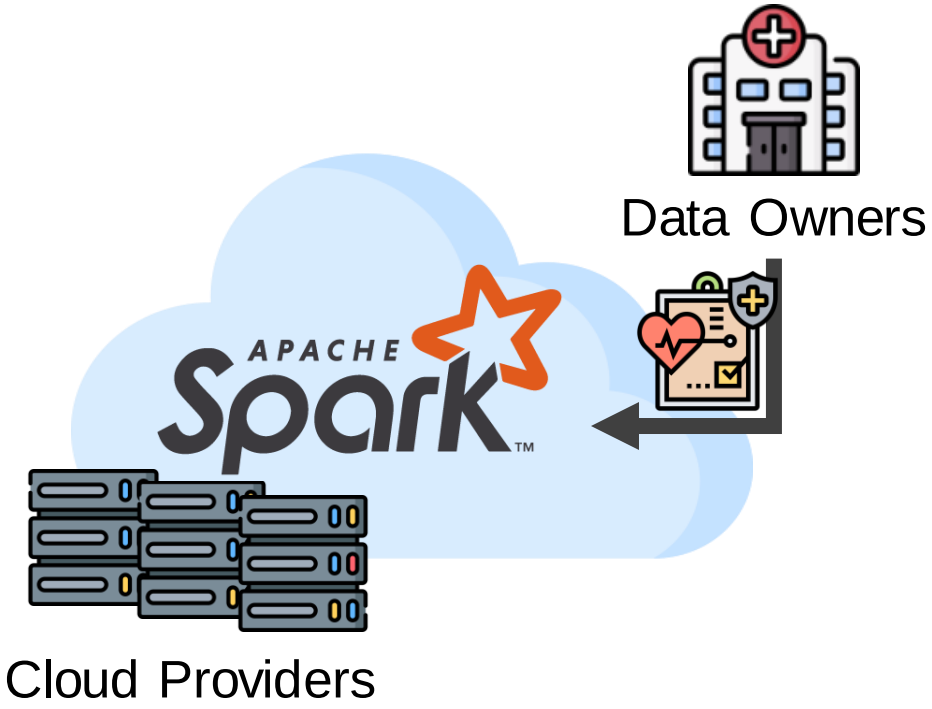
Cloud-based Spark: Collaborative Big Data Analytics

Cloud Providers

- Cloud based **Spark** platform

  Tempting approach for
  Collaborative big data analytics

Cloud-based Spark: Collaborative Big Data Analytics

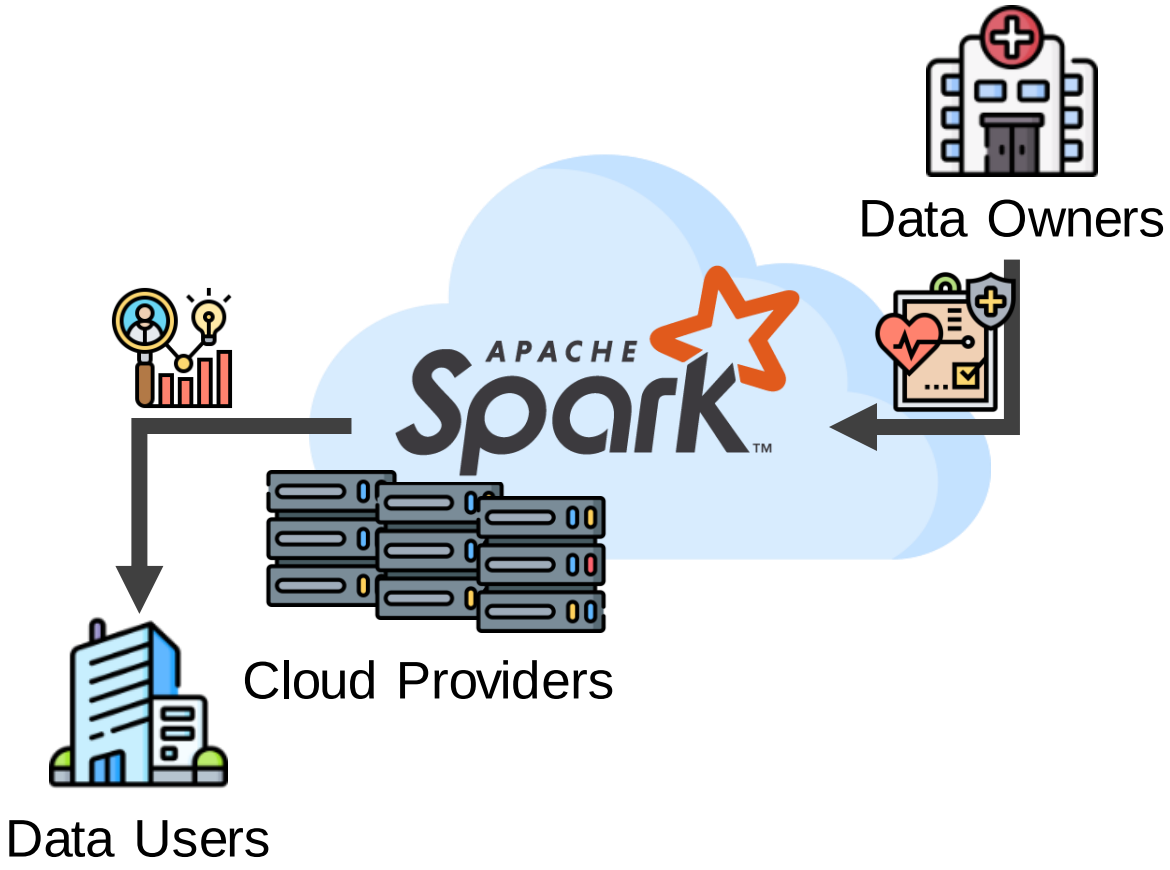


- Cloud based **Spark** platform

  Tempting approach for
  Collaborative big data analytics

- **Data owners**
  Easy deployment and management

Cloud-based Spark: Collaborative Big Data Analytics

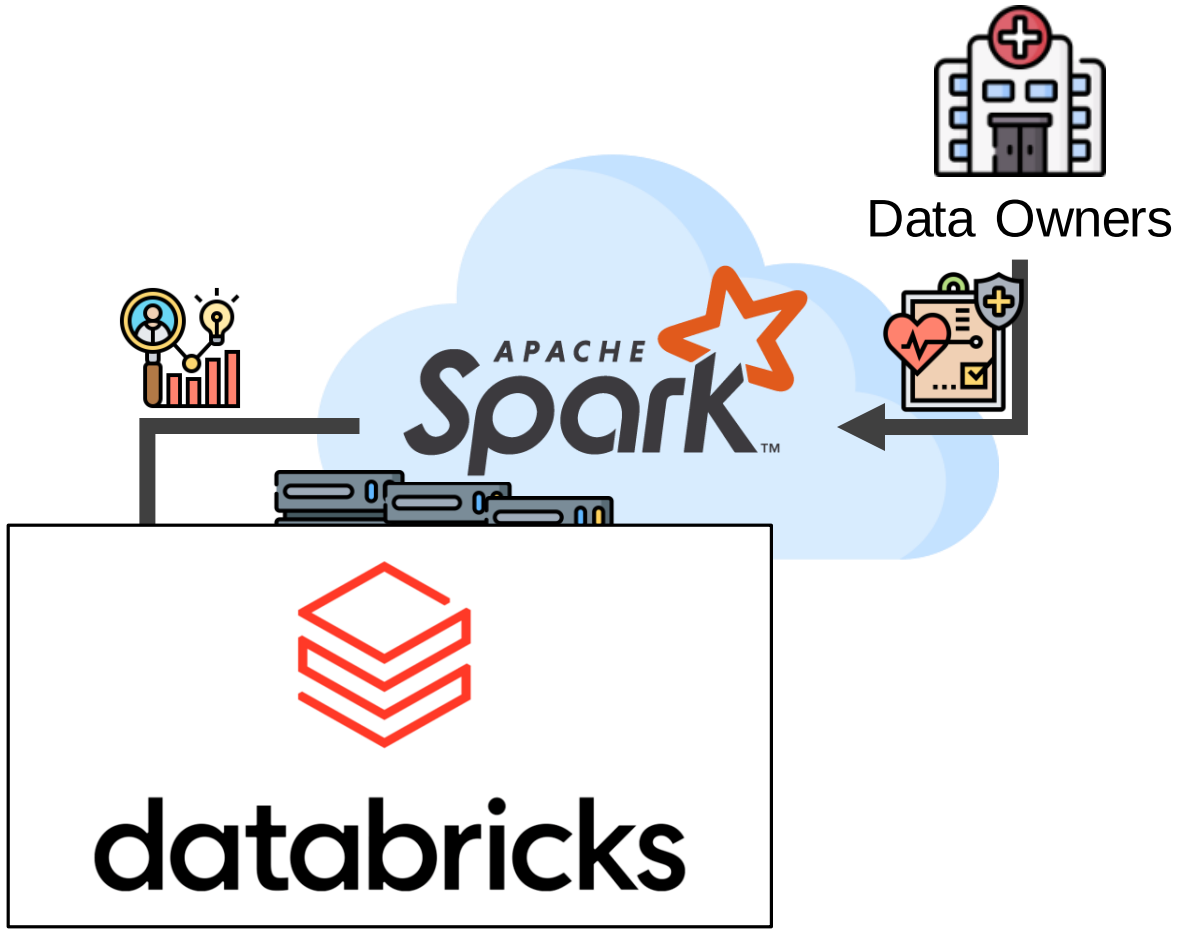


- Cloud based **Spark** platform

  Tempting approach for
  Collaborative big data analytics

- **Data owners**
  Easy deployment and management
- **Data users**
  Easy access and data analysis

Cloud-based Spark: Collaborative Big Data Analytics

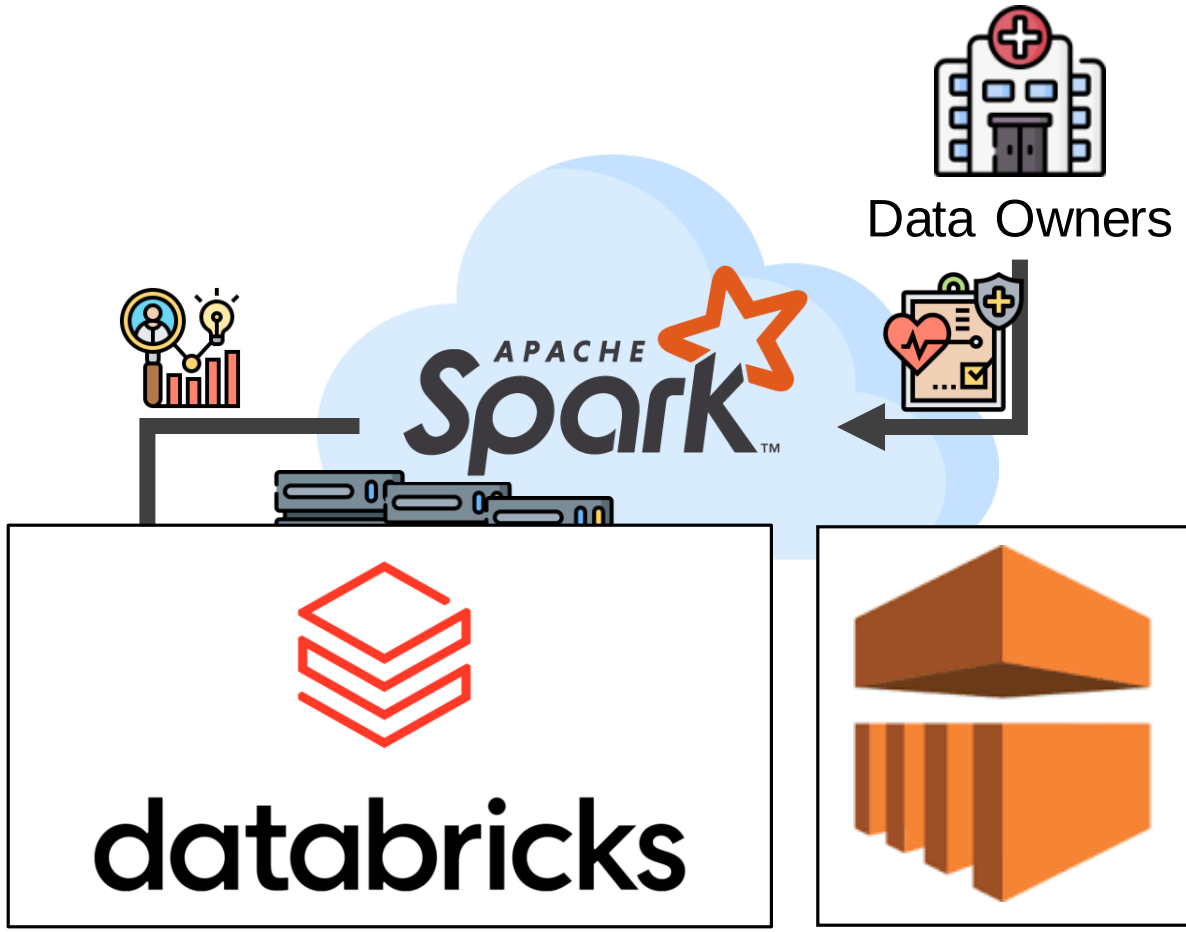


- Cloud based **Spark** platform

  Tempting approach for
  Collaborative big data analytics

- **Data owners**
  Easy deployment and management
- **Data users**
  Easy access and data analysis

Cloud-based Spark: Collaborative Big Data Analytics



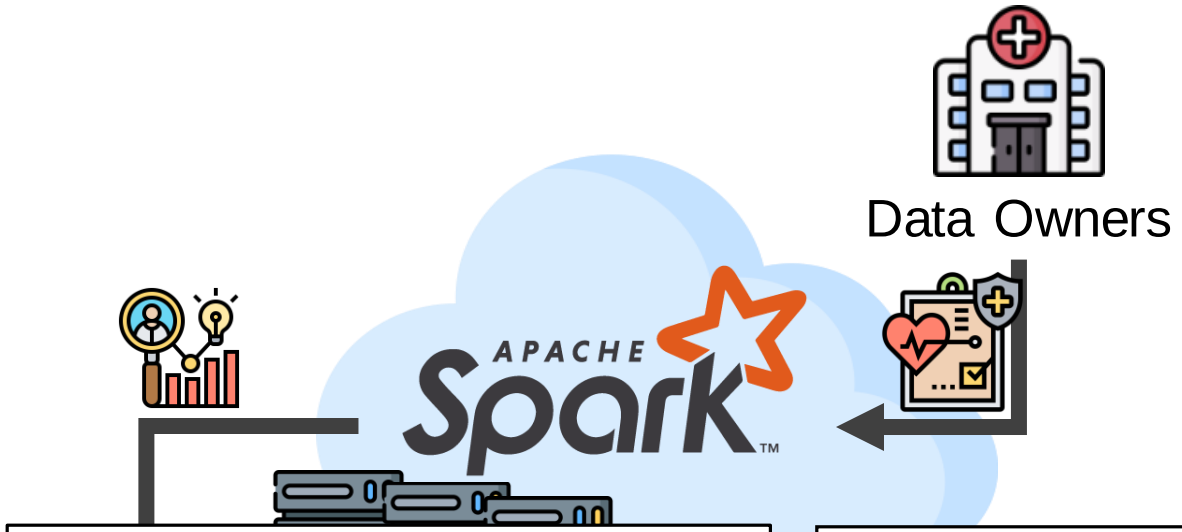
- Cloud based **Spark** platform

  Tempting approach for
  Collaborative big data analytics

- **Data owners**
  Easy deployment and management

s and data analysis

Cloud-based Spark: Collaborative Big Data Analytics



- Cloud based **Spark** platform

  Tempting approach for
  Collaborative big data analytics

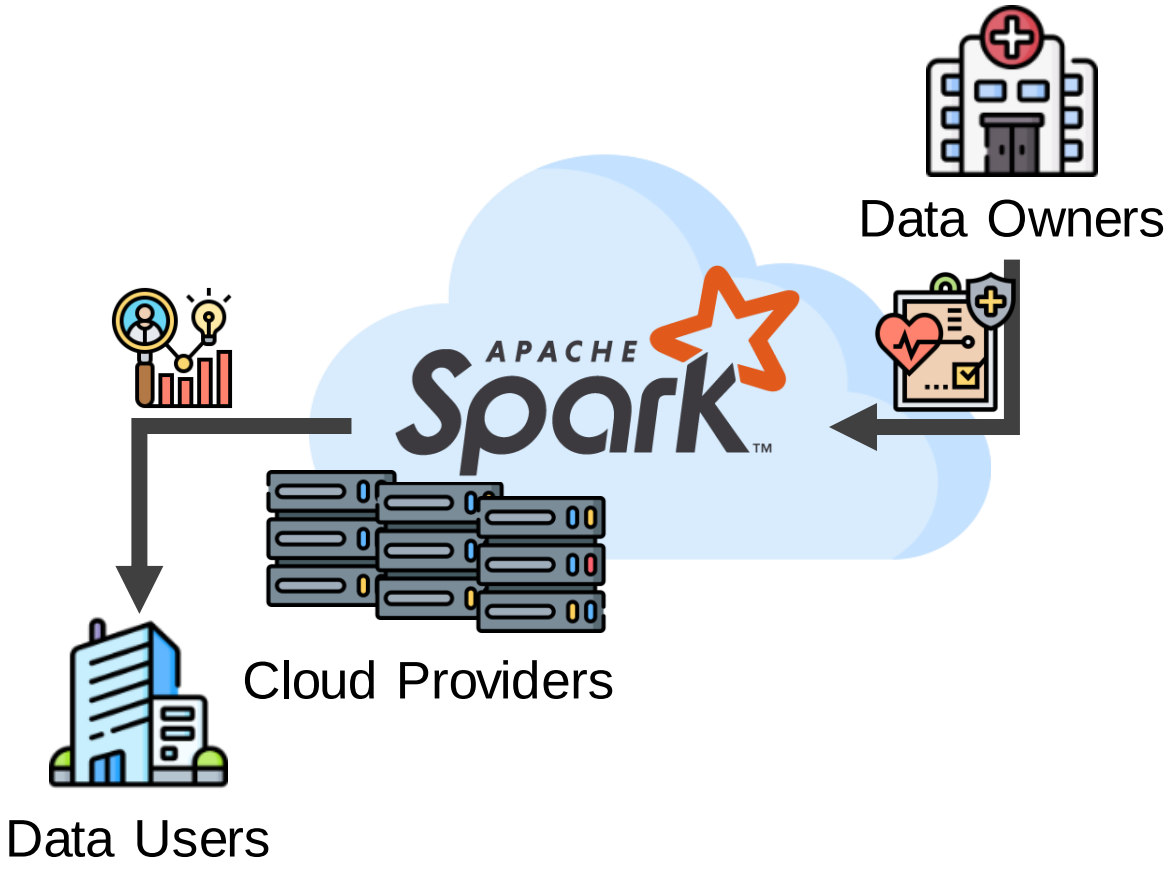
- **Data owners**
  Easy deployment and management

databricks

amazon EMR

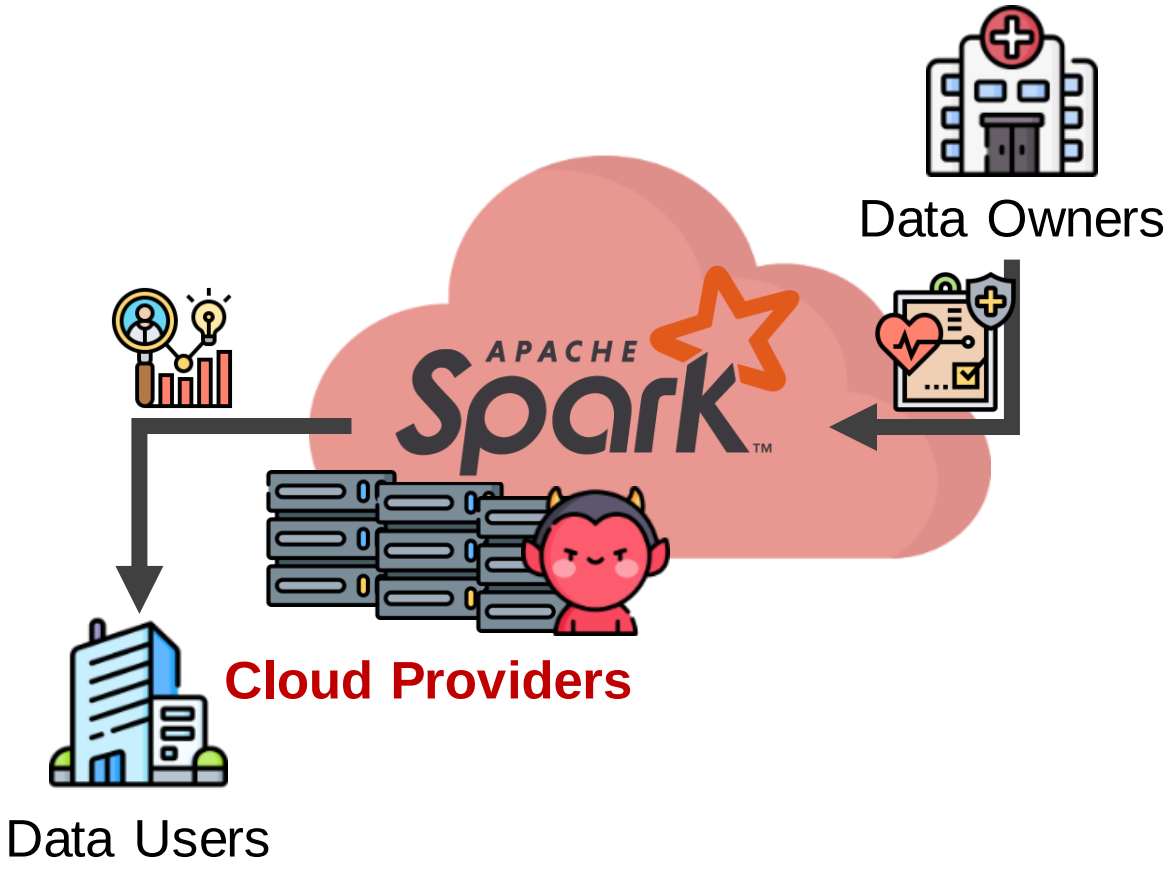
Azure Synapse Analytics

Problem: Possible Data Breach while Analysis



- **Risk of violating data owner's expectation**

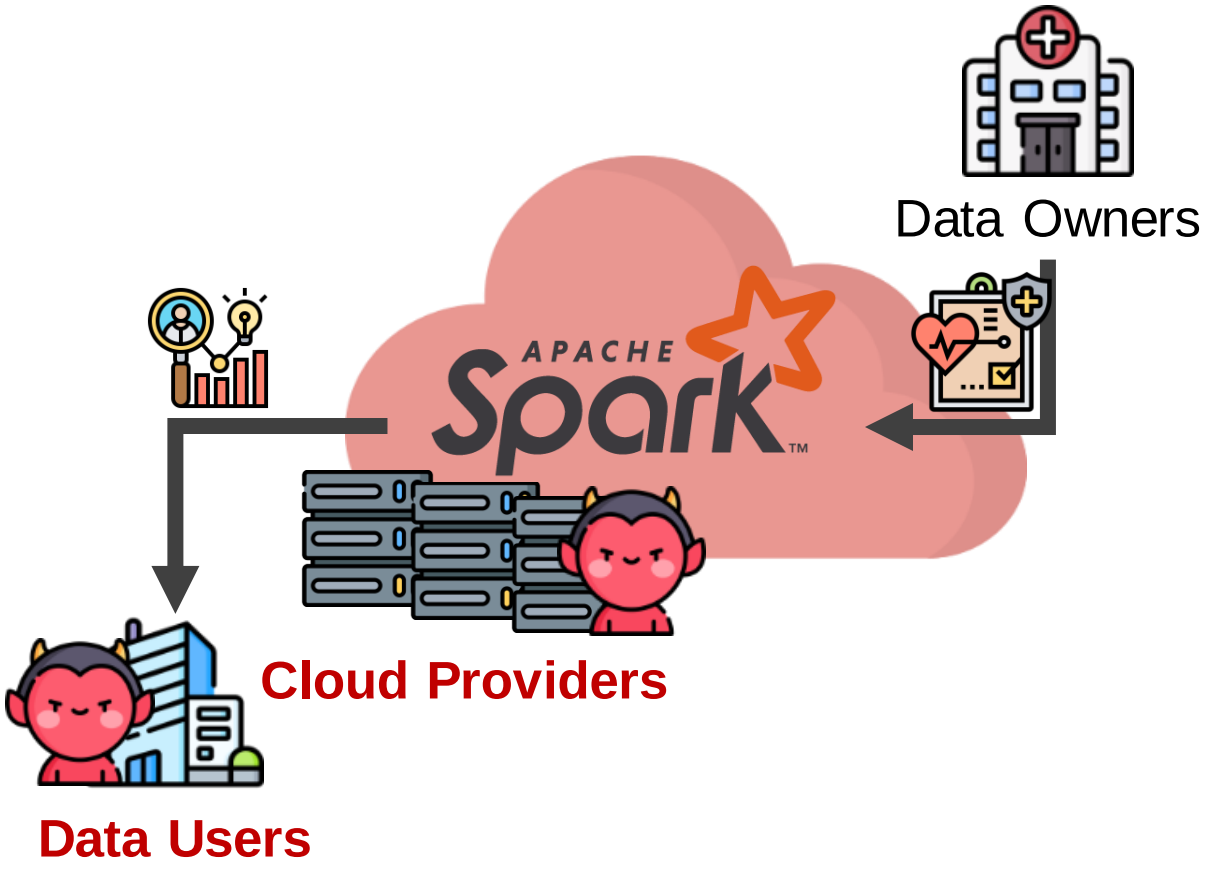
Problem: Possible Data Breach while Analysis



- **Risk of violating data owner's expectation**

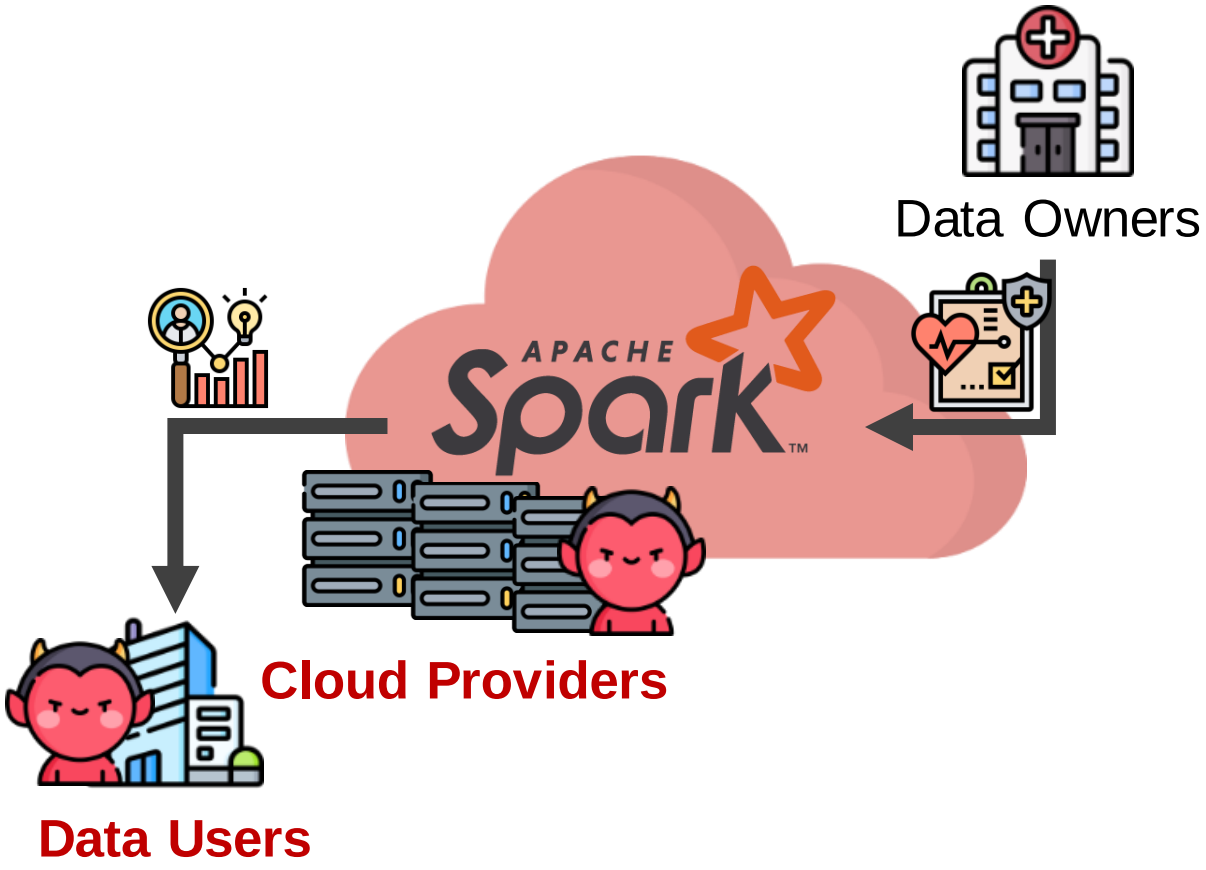
  Untrusted cloud providers

Problem: Possible Data Breach while Analysis



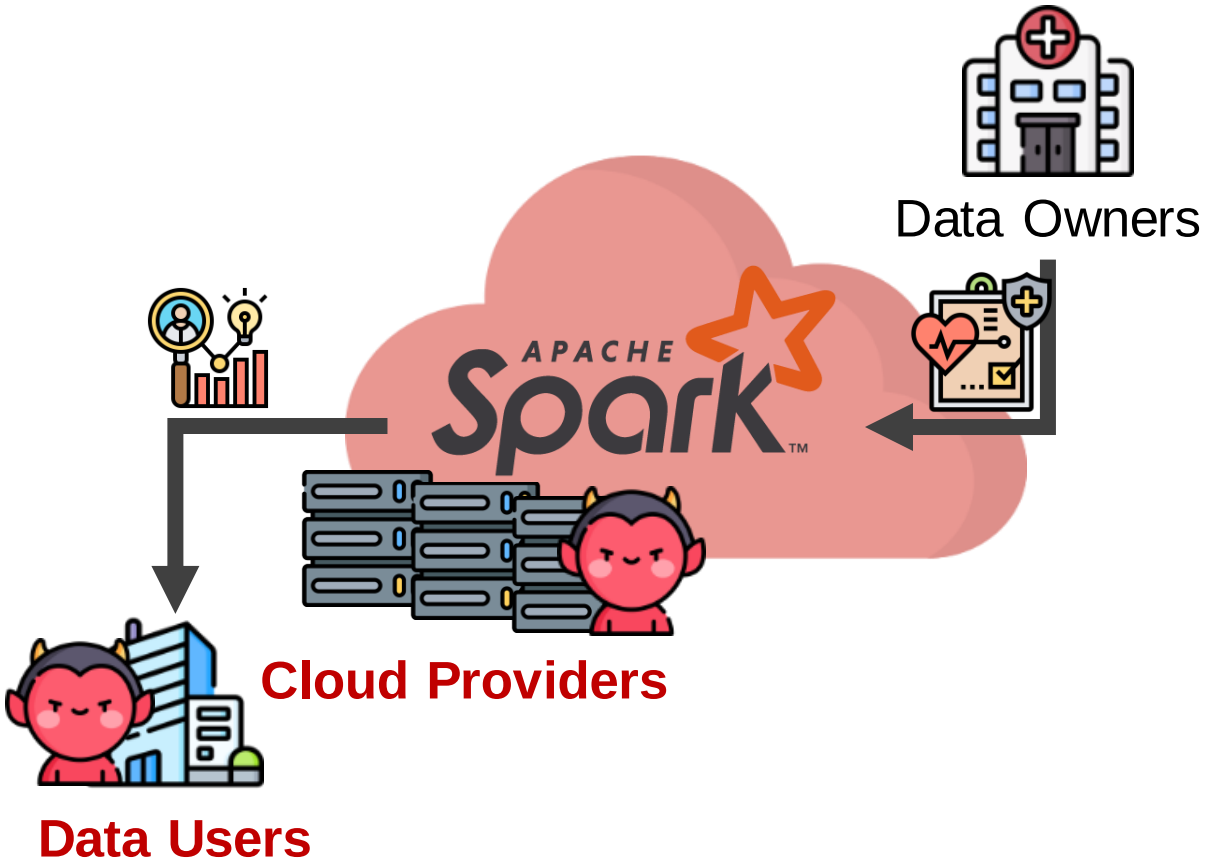
- **Risk of violating data owner's expectation**
  - *Untrusted cloud providers*
  - *Untrusted data users*

Problem: Possible Data Breach while Analysis



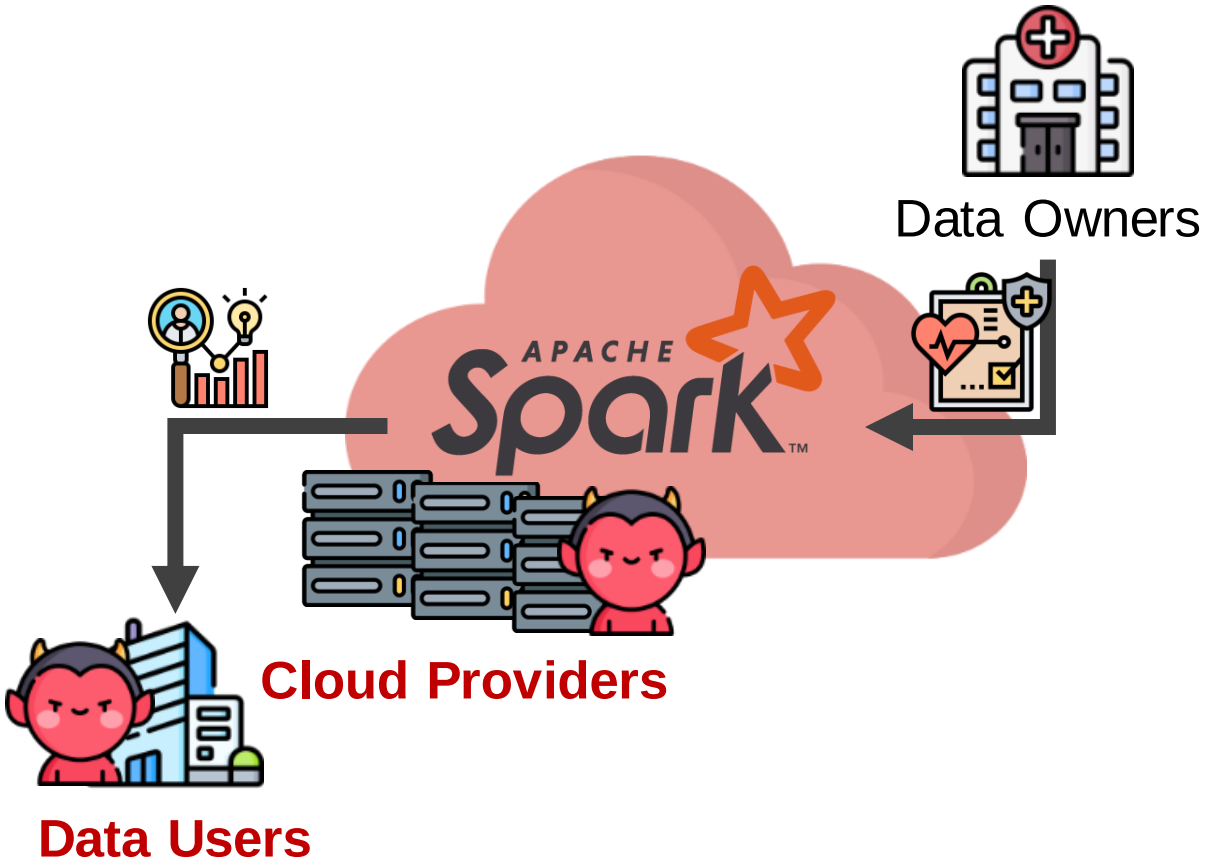
- **Risk of violating data owner's expectation**
  - *Untrusted cloud providers*
  - *Untrusted data users*
- Data under **privacy regulations**

Problem: Possible Data Breach while Analysis



- **Risk of violating data owner's expectation**
  - *Untrusted cloud providers*
  - *Untrusted data users*
- Data under **privacy regulations**
  - Clinical data
    - genome, medication history, …
  - Financial data
    - Credit history, transactions, …

Problem: Possible Data Breach while Analysis



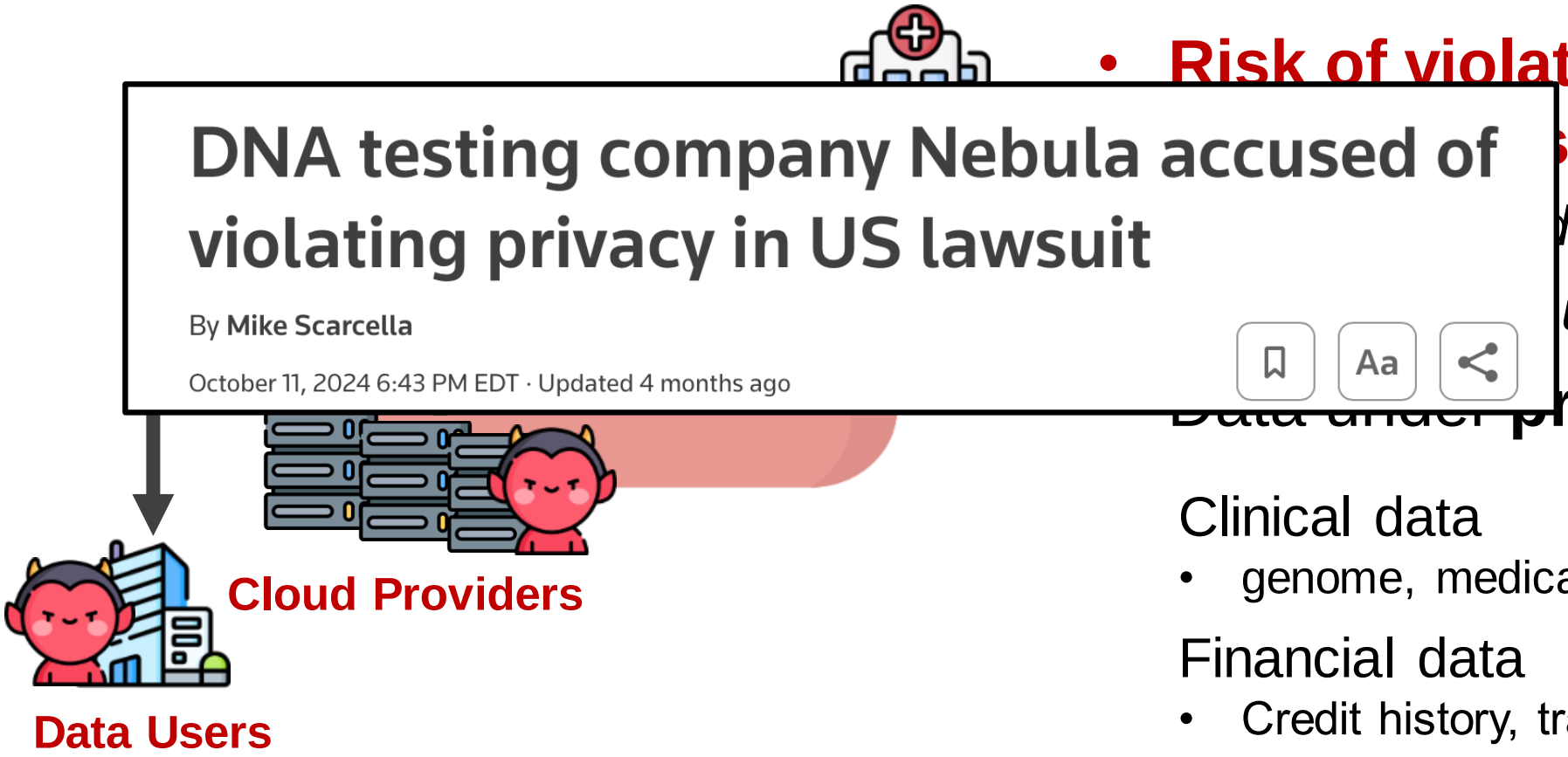
- **Risk of violating data owner's expectation**
  - *Untrusted cloud providers*
  - *Untrusted data users*
- Data under **privacy regulations**

Clinical data
- genome, medication history, …

Financial data
- Credit history, transactions, …

GDPR, CCPA, HIPAA

Problem: Possible Data Breach while Analysis



- **Risk of violating**
- **s expectation**
- *d providers*
- *users*

Data under privacy regulations

Clinical data
- genome, medication history, …

Financial data
- Credit history, transactions, …

GDPR, CCPA, HIPAA

Problem: Possible Data Breach while Analysis

DNA testing com
violating privacy i

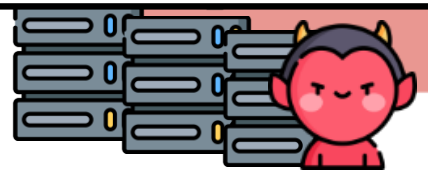
By **Mike Scarcella**

October 11, 2024 6:43 PM EDT · Updated 4 months

Patients sue Walgreens for making money on their data

By **Diana Manos** | March 18, 2011 | 09:26 AM

Walgreen Company customers have filed a lawsuit in California this week accusing the national drug-store chain of unlawfully selling medical information gleaned from patient prescriptions, Reuters Legal reports.



Cloud Providers

Data Users

Clinical data

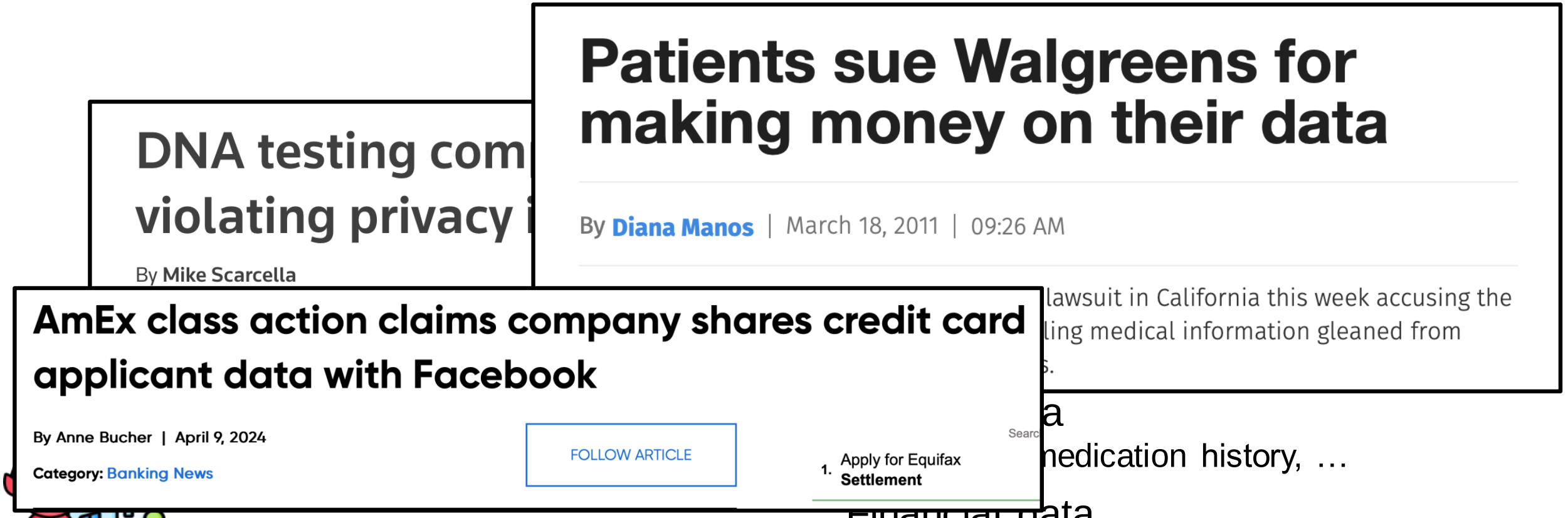
- genome, medication history, …

Financial data

- Credit history, transactions, …

GDPR, CCPA, HIPAA

Problem: Possible Data Breach while Analysis



Data Users

a

nedication history, …

Financial data

- Credit history, transactions, …

GDPR, CCPA, HIPAA

Problem: Possible Data Breach while Analysis

DNA testing com
violating privacy i

By Mike Scarcella

Patients sue Walgreens for making money on their data

By Diana Manos | March 18, 2011 | 09:26 AM

lawsuit in California this week accusing the
ling medical information gleaned from

AmEx class action claims company shares credit card applicant data with F

By Anne Bucher | April 9, 2024

Category: Banking News

First multi-million Euro GDPR fine: Google LLC fined €50 million under GDPR for transparency and consent infringements in relation to use of personal data for personalized ads

Data Users

Problem: Possible Data Breach while Analysis

The Costs of an Unnecessarily Stringent Federal Data Privacy Law

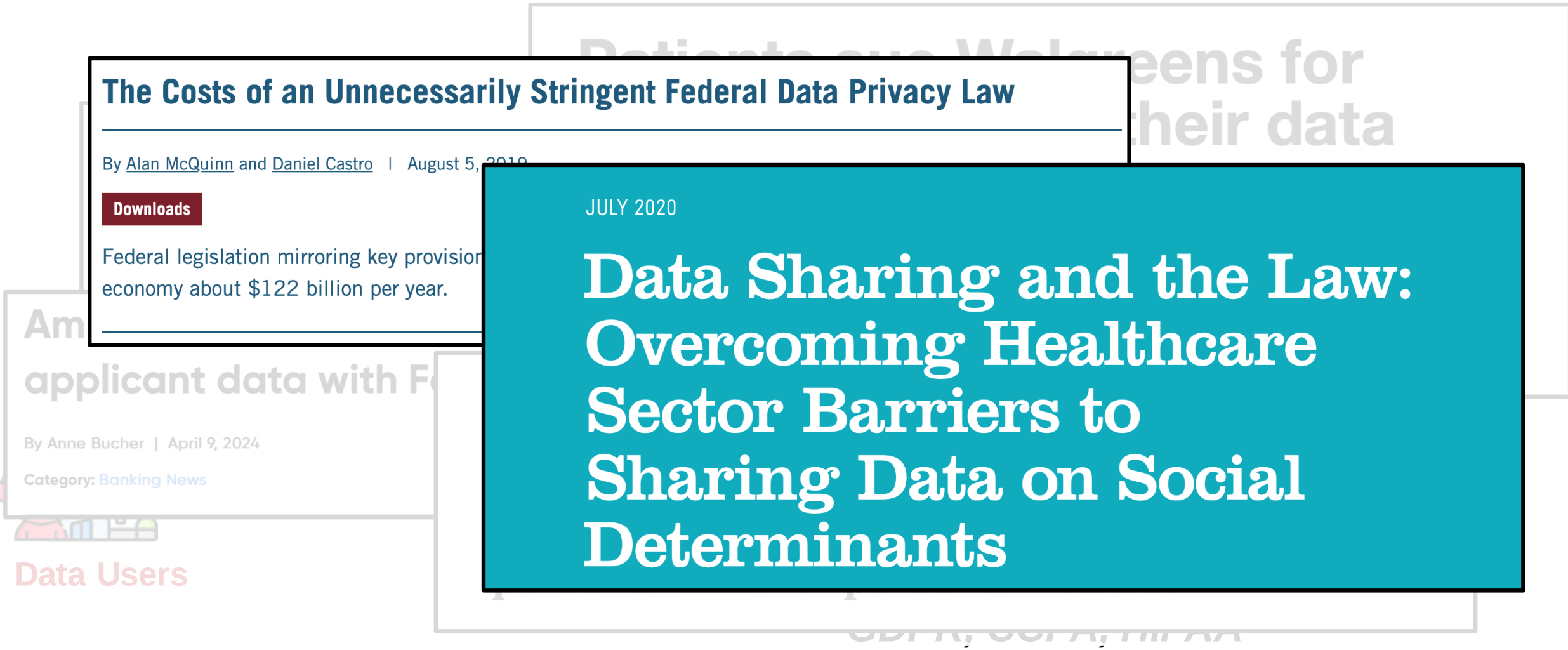
By Alan McQuinn and Daniel Castro | August 5, 2019

Downloads

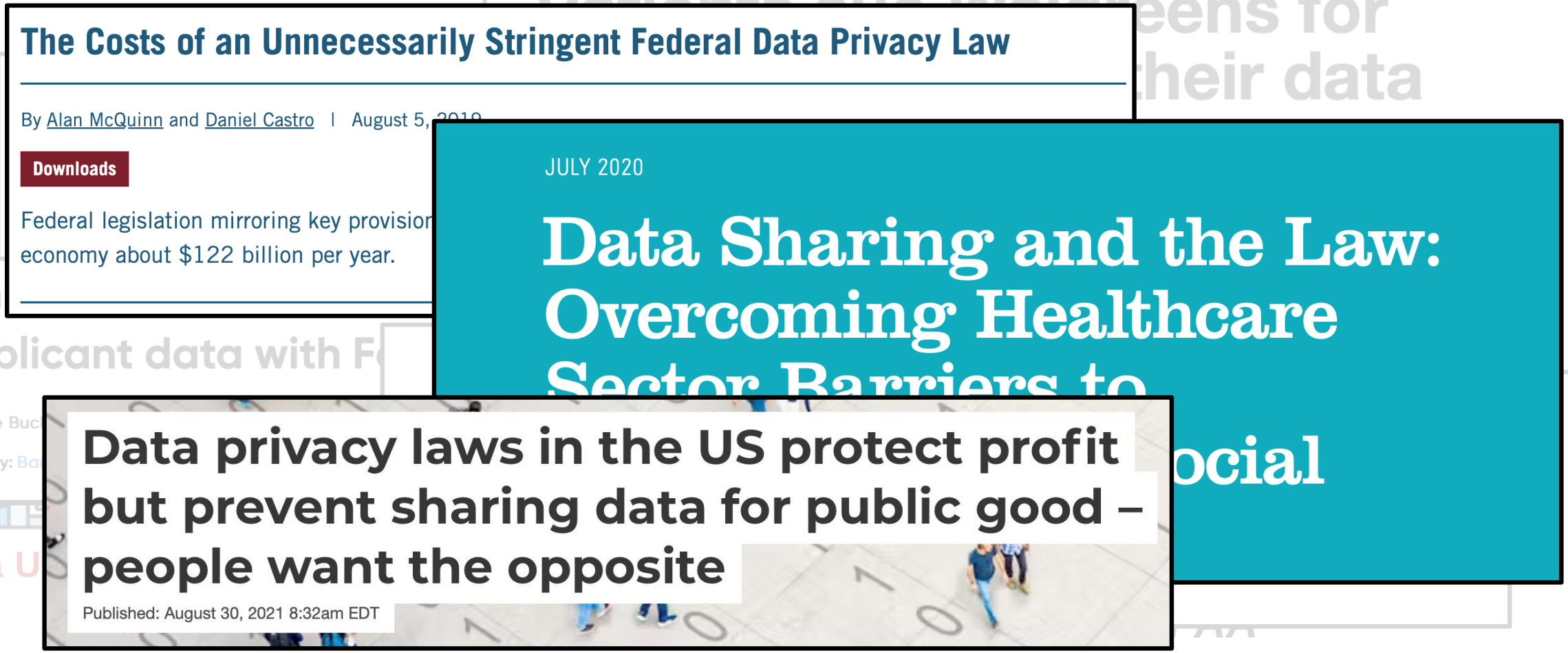
Federal legislation mirroring key provisions of privacy laws in Europe or California could cost the U.S. economy about $122 billion per year.

Data Users

Problem: Possible Data Breach while Analysis



Problem: Possible Data Breach while Analysis



Problem: Possible Data Breach while Analysis

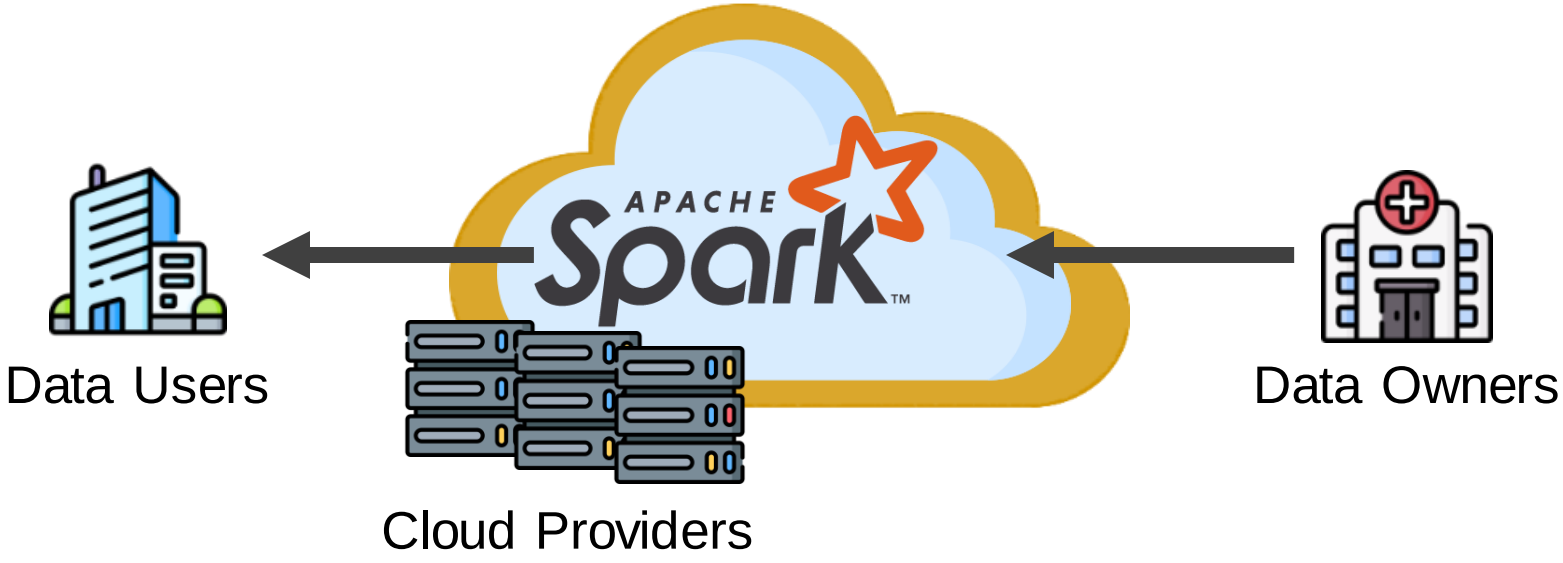
What We Want

Allow the data owners to avoid regulatory violations while sharing their data

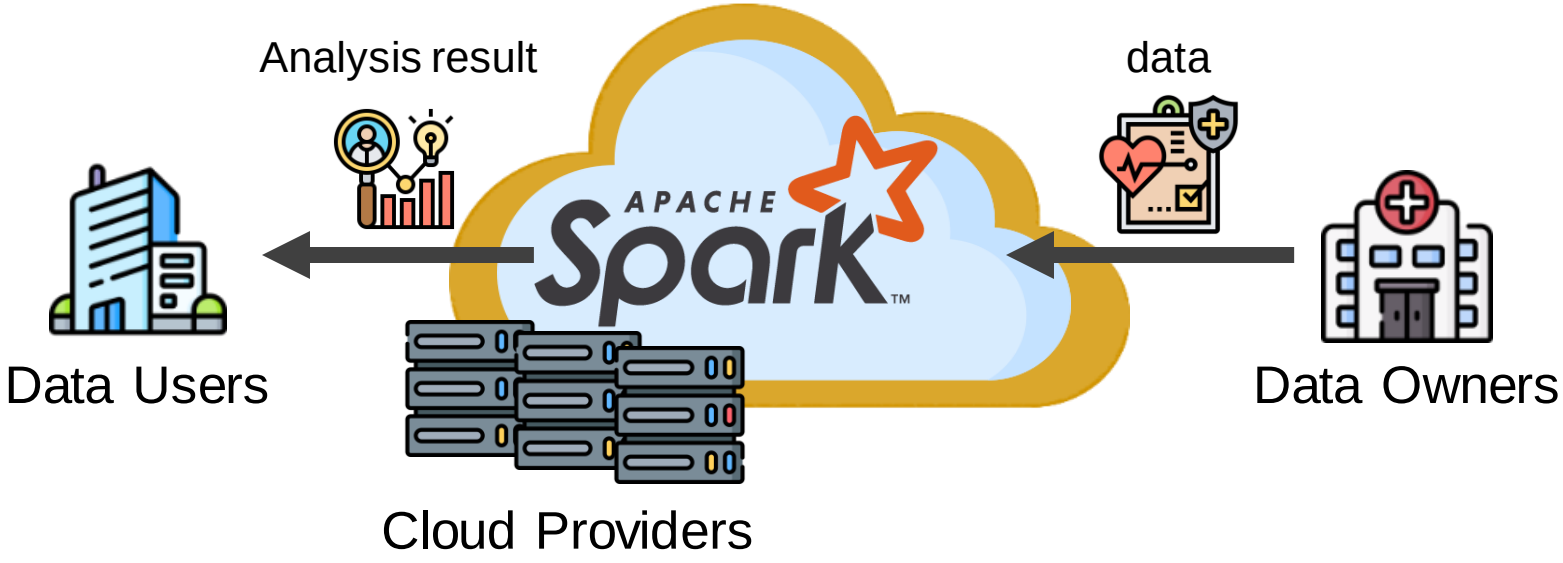
So, we propose a **New Architecture** of

Cloud-based Spark for Secure Data Analytics

Rearchitecting Spark for Secure Data Analytics

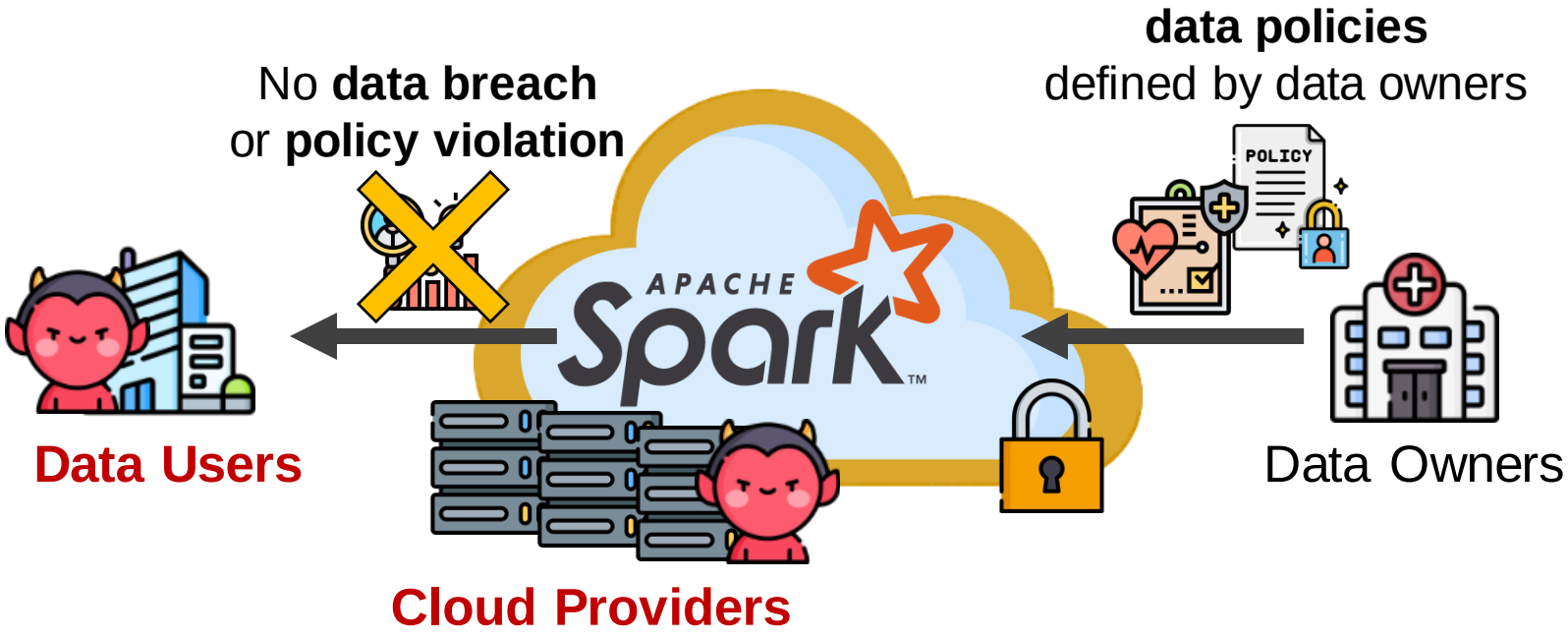


Rearchitecting Spark for Secure Data Analytics



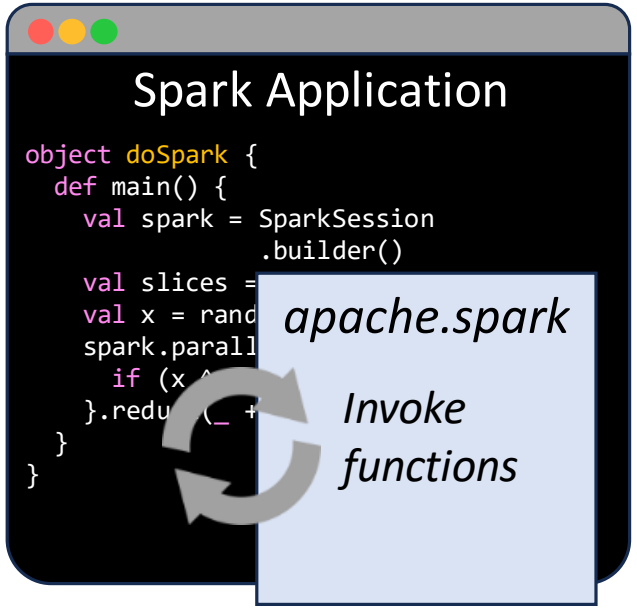
- **Usability:** We keep data to be analyzed as before

Rearchitecting Spark for Secure Data Analytics



- **Usability:** We keep data to be analyzed as before
- **Security:** but, prevent data breach or violating data policies

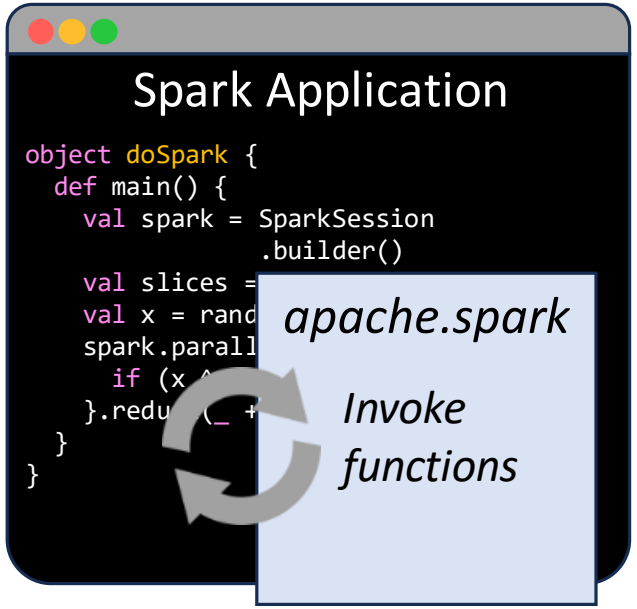
Security Requirements for Protecting Data



Data analysis procedure of Spark application

- User's code interacting with **Spark Libraries**

Security Requirements for Protecting Data

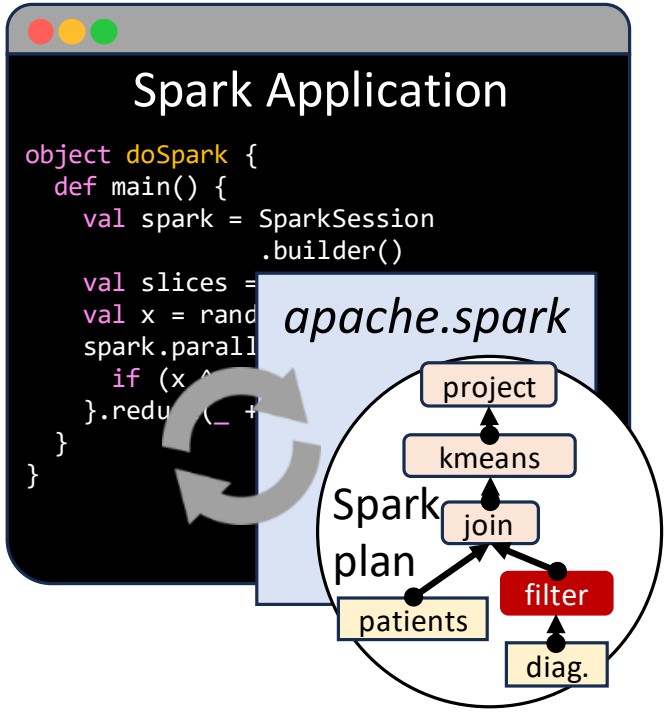


Data analysis procedure of Spark application

- User's code interacting with **Spark Libraries**

For simplifying distributed computation

Security Requirements for Protecting Data



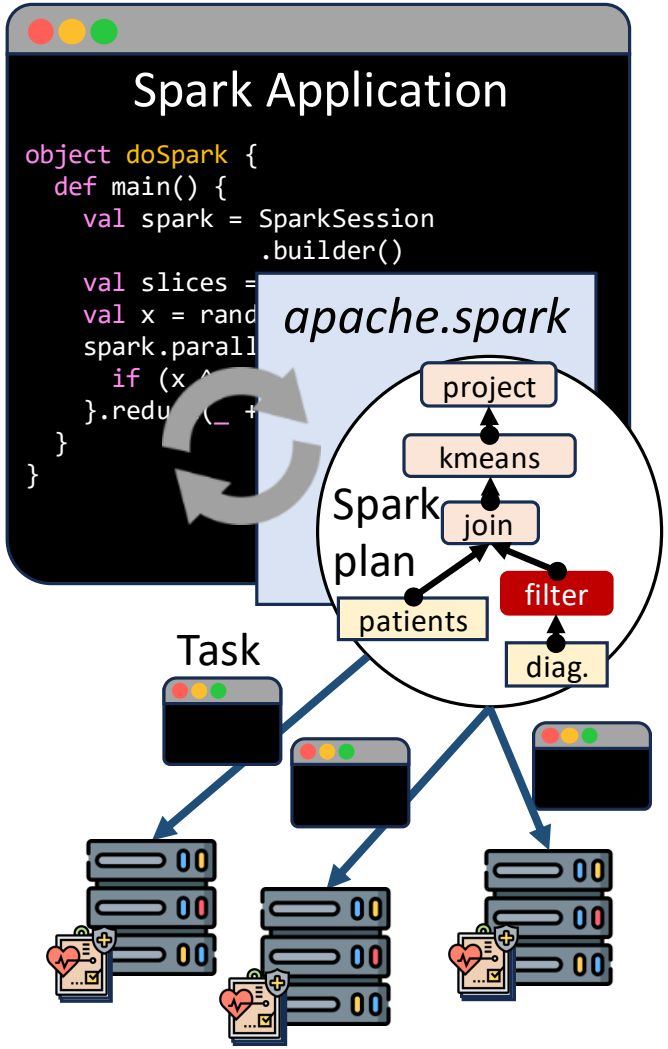
Data analysis procedure of Spark application

- User's code interacting with **Spark Libraries**

For simplifying distributed computation

1. ***Spark plan*** is internally constructed by the library.

Security Requirements for Protecting Data



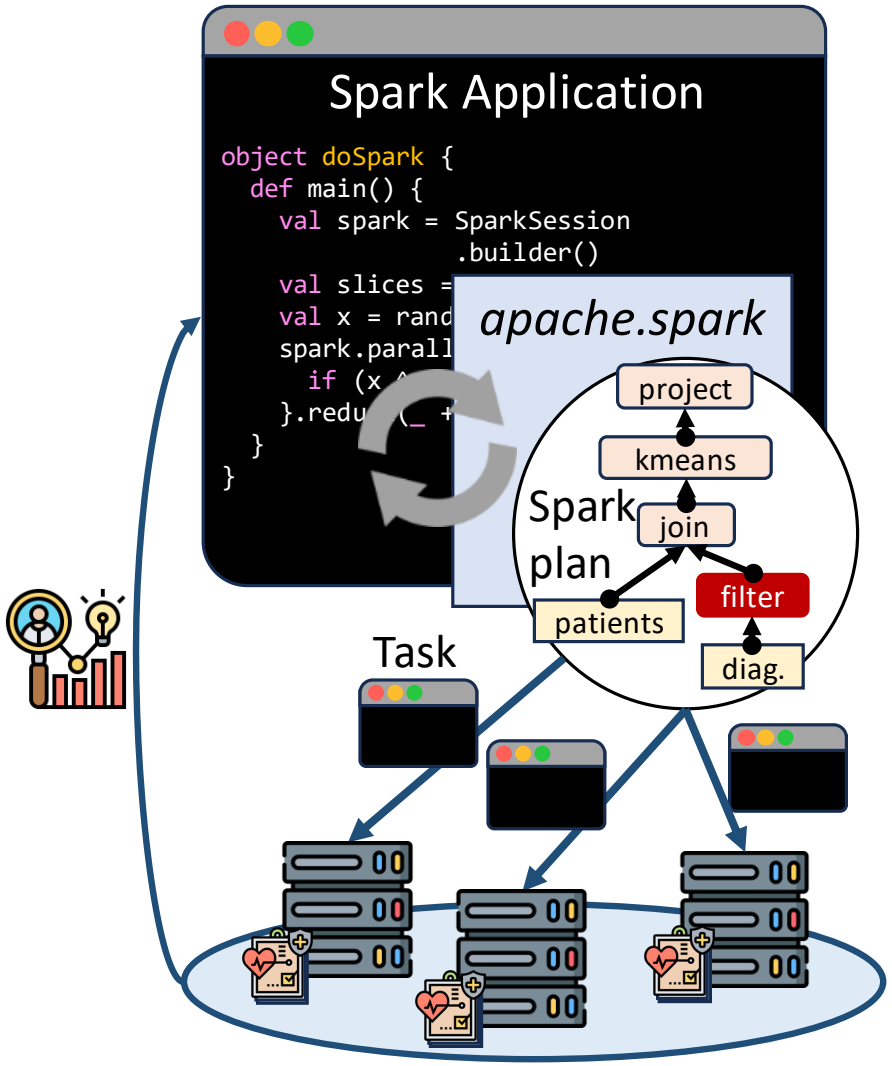
Data analysis procedure of Spark application

- User's code interacting with **Spark Libraries**

For simplifying distributed computation

1. ***Spark plan*** is internally constructed by the library.
2. **Tasks** are constructed from the plan and executed on the data in distributed nodes.

Security Requirements for Protecting Data



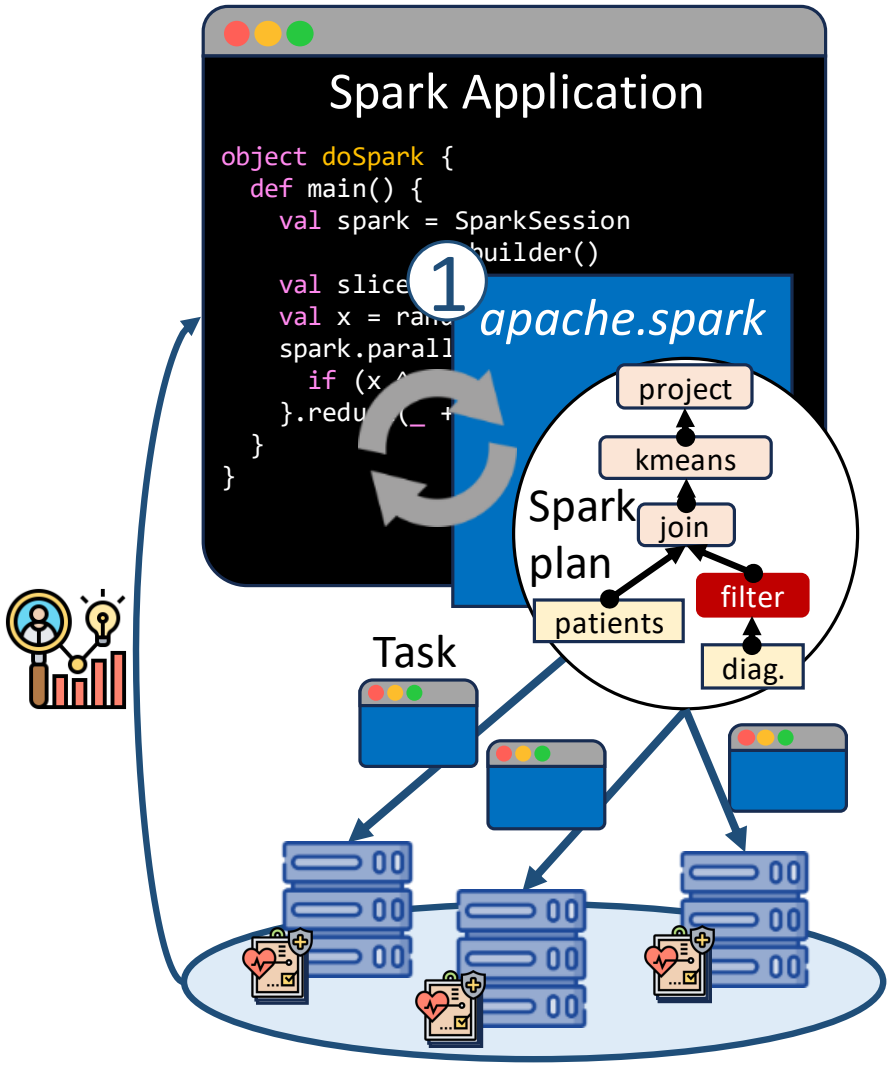
Data analysis procedure of Spark application

- User's code interacting with **Spark Libraries**

For simplifying distributed computation

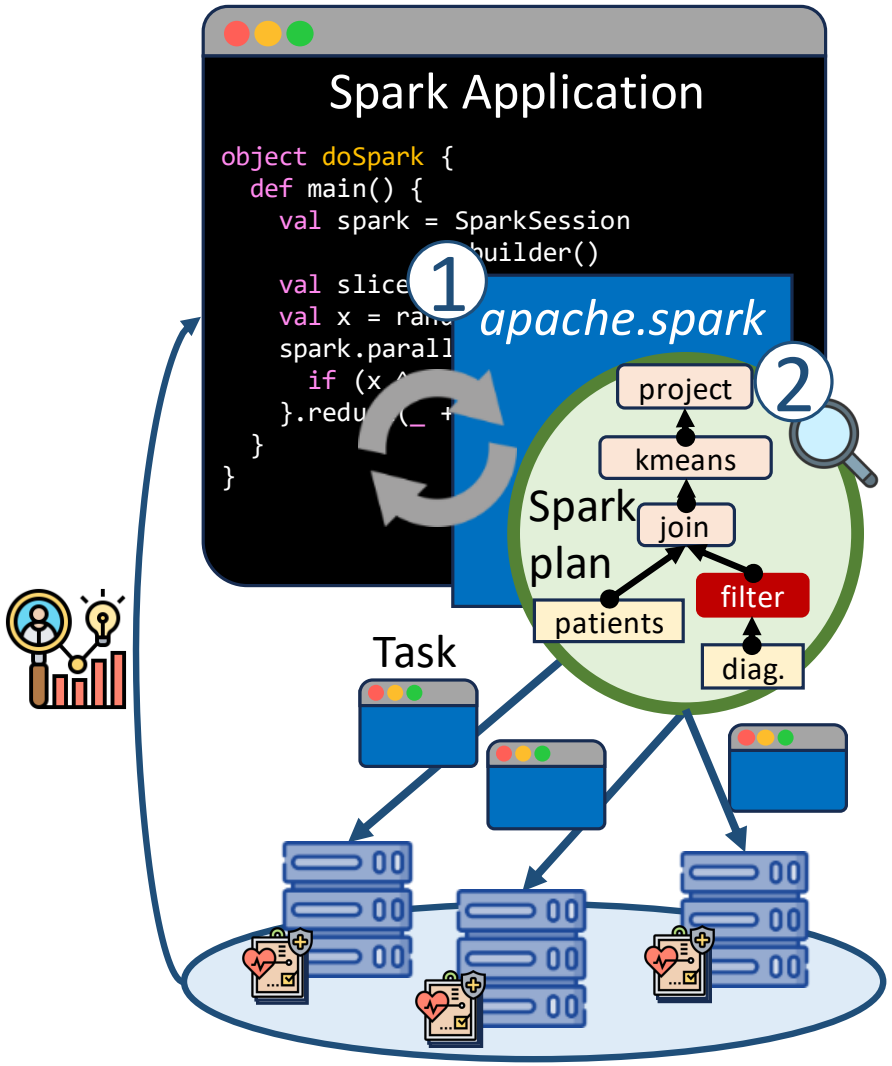
1. ***Spark plan*** is internally constructed by the library.
2. **Tasks** are constructed from the plan and executed on the data in distributed nodes.
3. **Analysis result** is returned.

Security Requirements for Protecting Data



1. Ensure the confidentiality and integrity of the entire **data analysis pipeline**

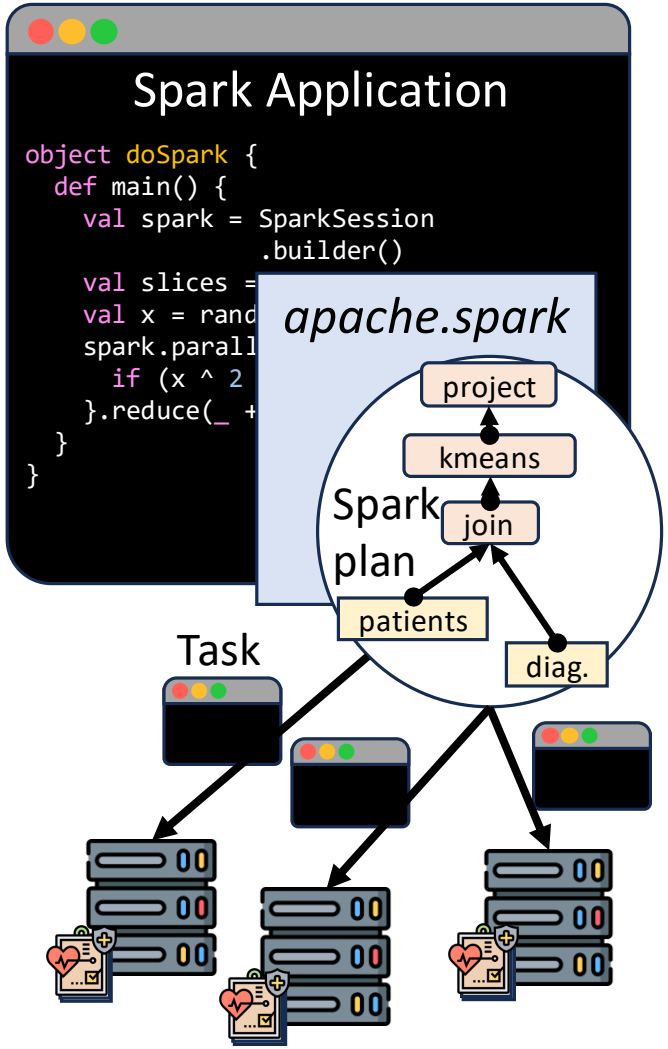
Security Requirements for Protecting Data



1. Ensure the confidentiality and integrity of the entire **data analysis pipeline**

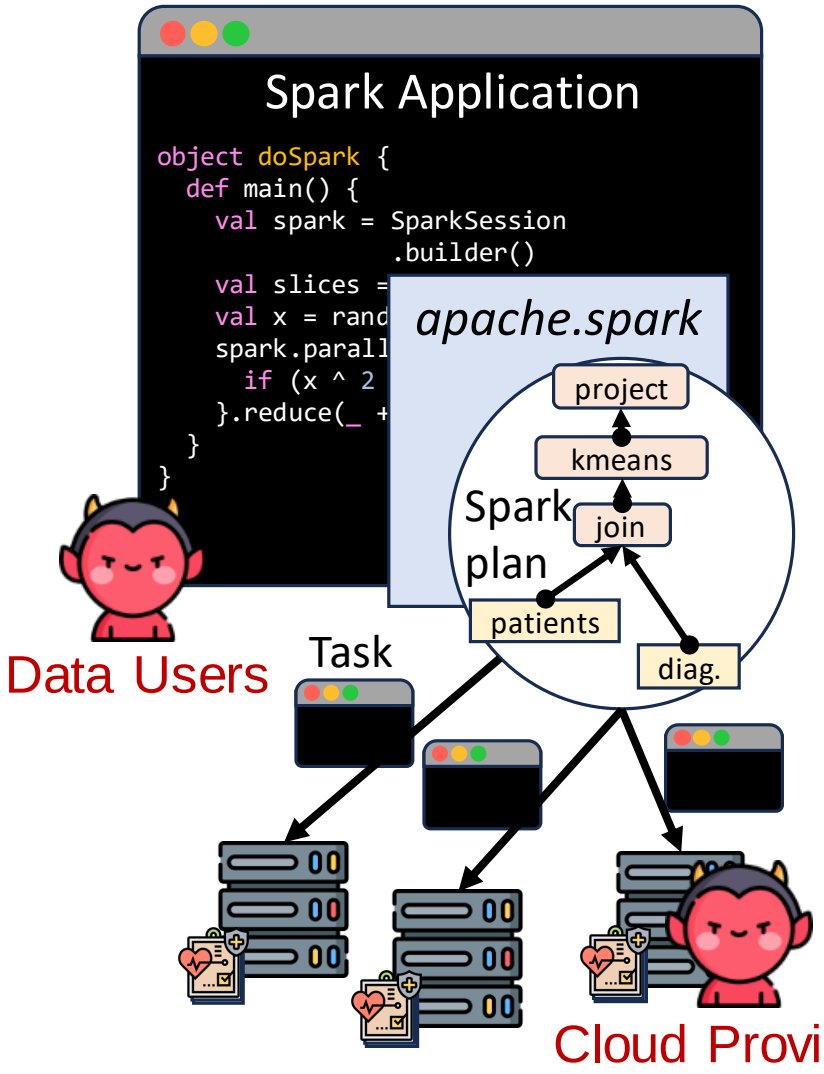
2. Ensure the Spark plans by data users respect the **owner-defined policies**

Attack 1. Compromising Data Analysis Pipeline



- Spark application is fully controlled by data users and cloud providers

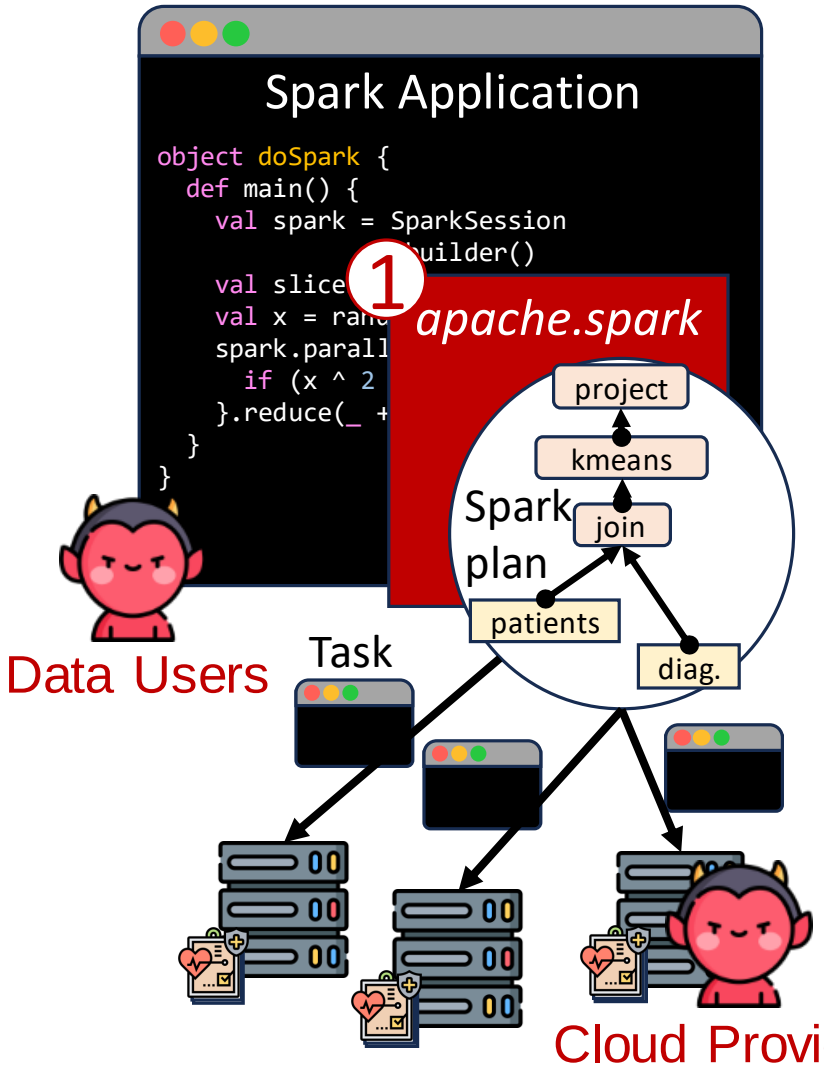
Attack 1. Compromising Data Analysis Pipeline



- Spark application is fully controlled by data users and cloud providers

Possible Attack Vectors, when
Data Users and **Cloud Providers** are compromised

Attack 1. Compromising Data Analysis Pipeline

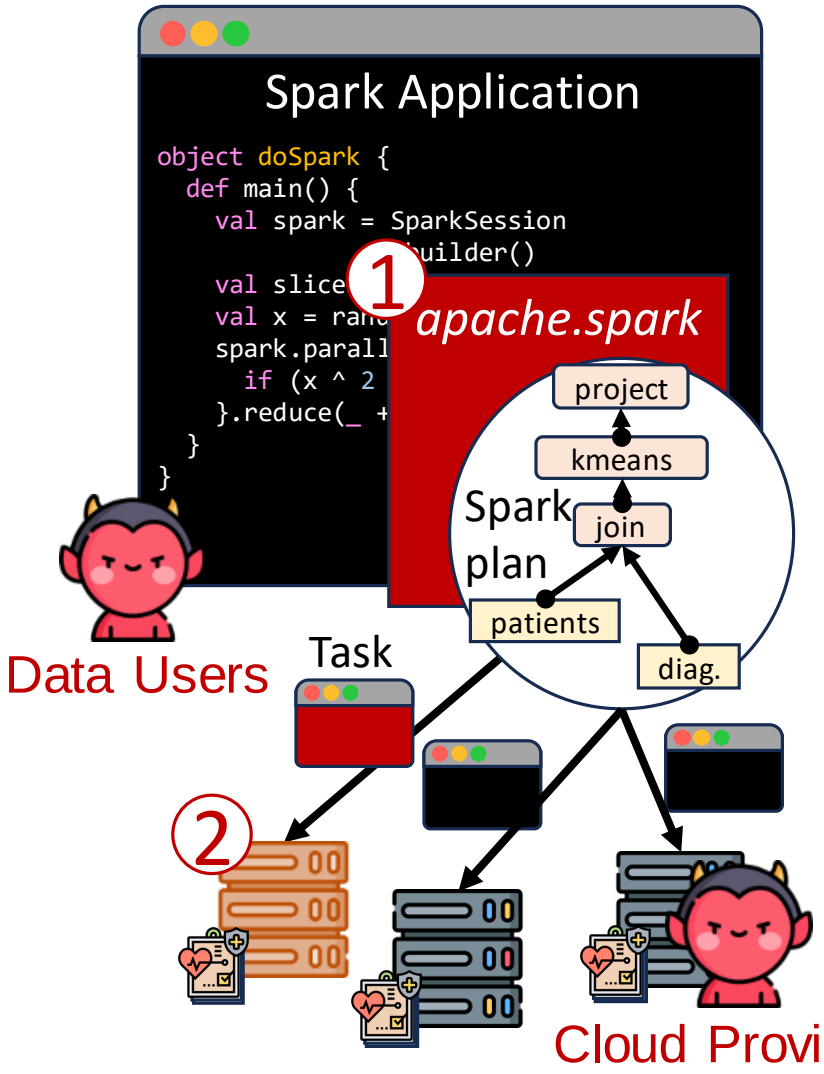


- Spark application is fully controlled by data users and cloud providers

Possible Attack Vectors, when
Data Users and **Cloud Providers** are compromised

1. Compromising **Spark library**

Attack 1. Compromising Data Analysis Pipeline

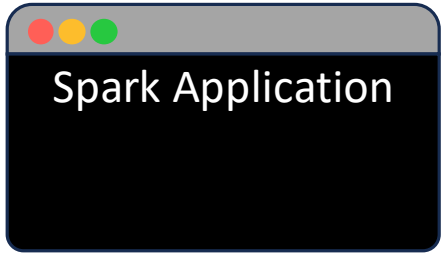


- Spark application is fully controlled by data users and cloud providers

Possible Attack Vectors, when **Data Users** and **Cloud Providers** are compromised

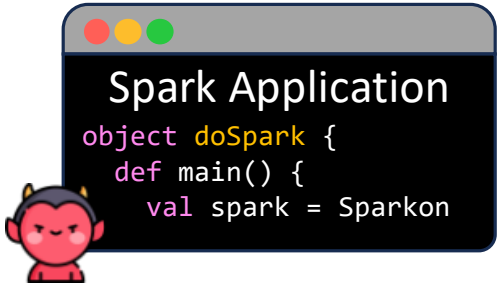
1. Compromising **Spark library**
2. Compromising the **distributed nodes**

Defense 1. Securing the Data Analysis Pipeline



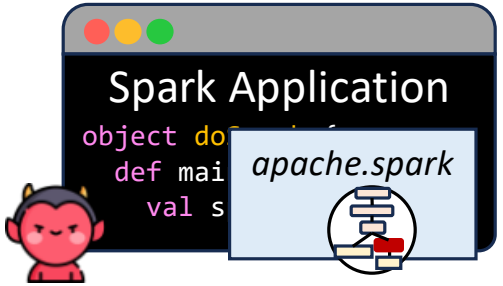
- **Compartmentalization**

Defense 1. Securing the Data Analysis Pipeline



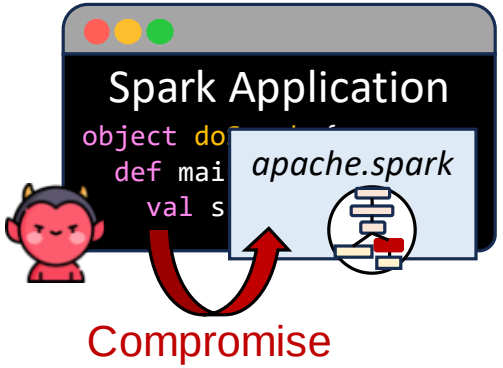
- **Compartmentalization**

Defense 1. Securing the Data Analysis Pipeline



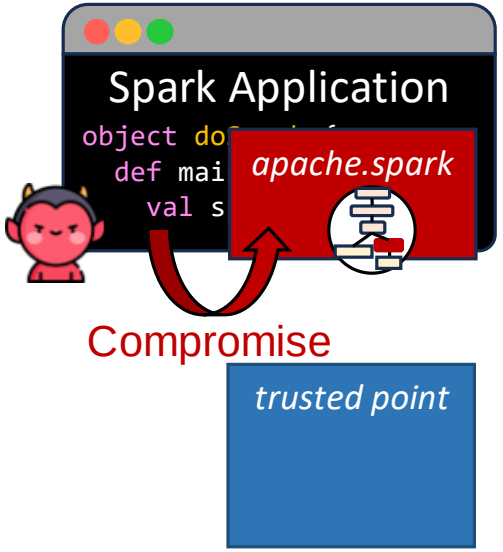
- **Compartmentalization**

Defense 1. Securing the Data Analysis Pipeline



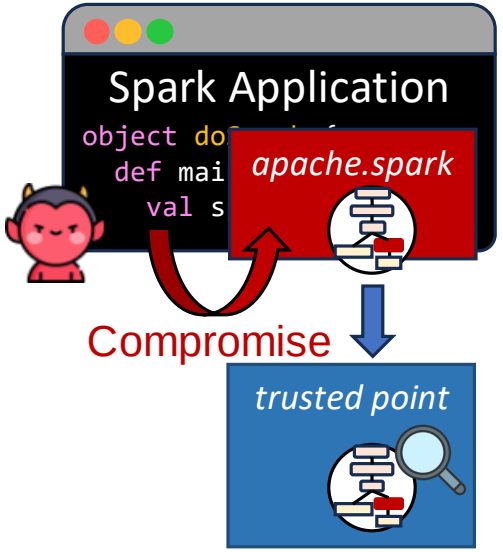
- **Compartmentalization**

Defense 1. Securing the Data Analysis Pipeline



- **Compartmentalization**
  Separate the address space and isolate Spark core components from user's code

Defense 1. Securing the Data Analysis Pipeline

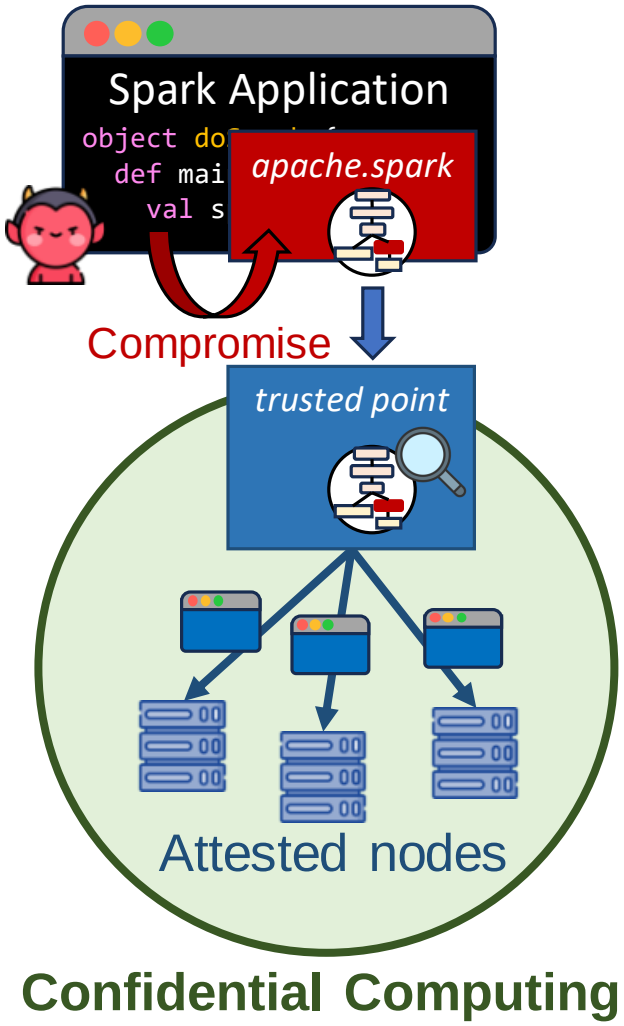


- **Compartmentalization**

  Separate the address space and isolate Spark core components from user's code

  Enforce the Spark plans to be relayed to a ***trusted point*** to be actually executed on data

Defense 1. Securing the Data Analysis Pipeline



- **Compartmentalization**

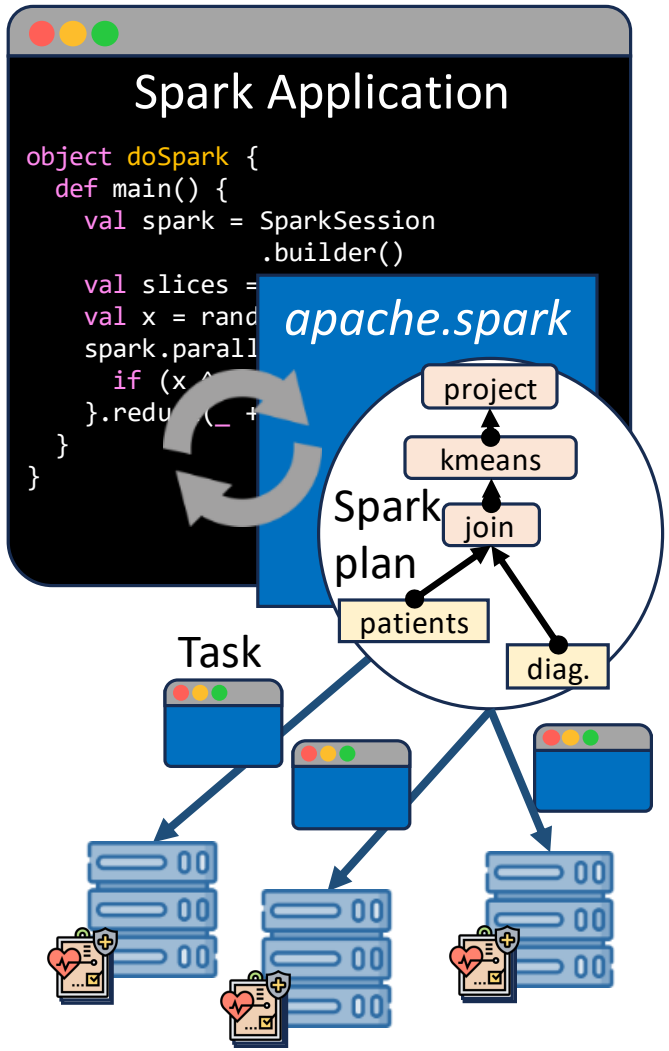
  Separate the address space and isolate Spark core components from user's code

  Enforce the Spark plans to be relayed to a ***trusted point*** to be actually executed on data

- **Distributed Confidential Computing**

  Entire Spark plan execution is protected by the confidential computing environment

Attack 2. Building Malicious Spark Plan

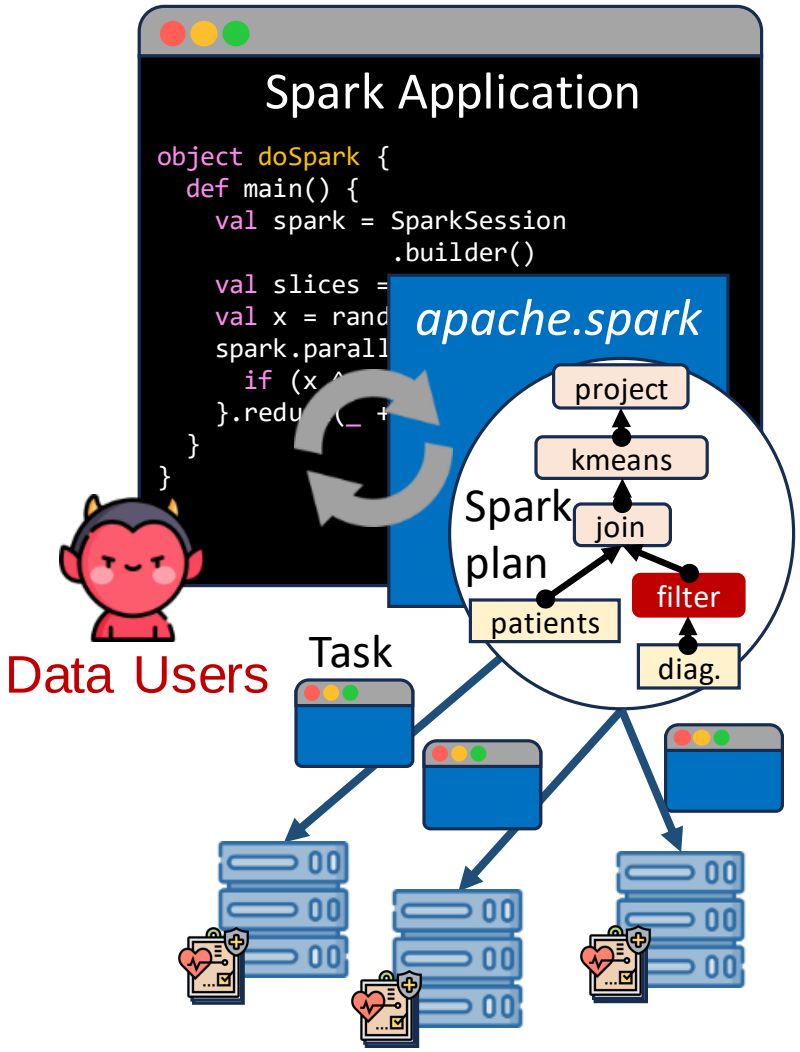


- Spark application is fully controlled by data users and cloud providers

Possible Attack Vectors, when
Data Users and **Cloud Providers** are compromised

~~1. Compromising **Spark library**~~
~~2. Compromising the **distributed nodes**~~

Attack 2. Building Malicious Spark Plan

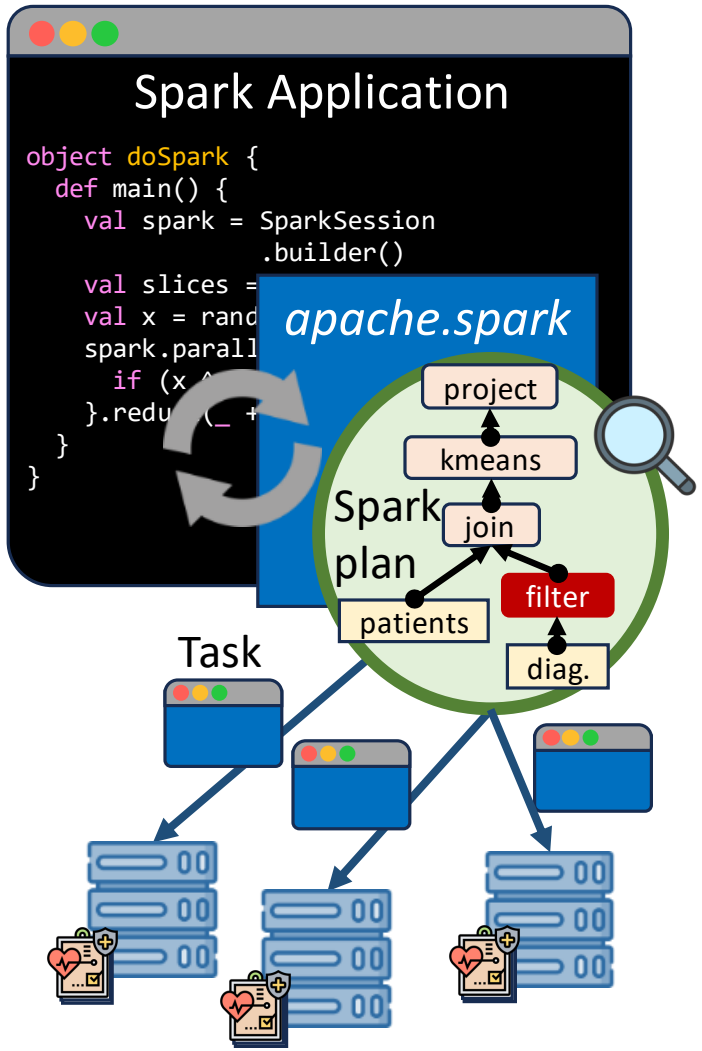


- Spark application is fully controlled by data users and cloud providers

Possible Attack Vectors, when
Data Users and **Cloud Providers** are compromised

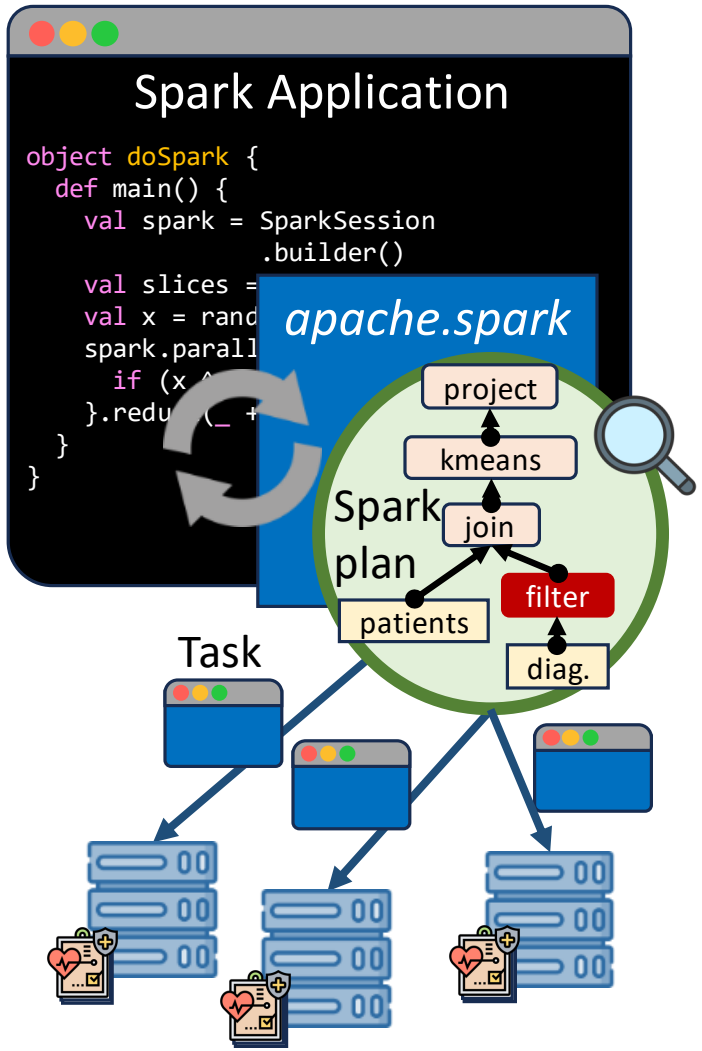
1. ~~Compromising **Spark library**~~
2. ~~Compromising the **distributed nodes**~~
3. Building a policy violating **Spark plan**

Defense 2. Enforcing Policy on Spark Plans



- **New policy check mechanism based on pattern-matching**

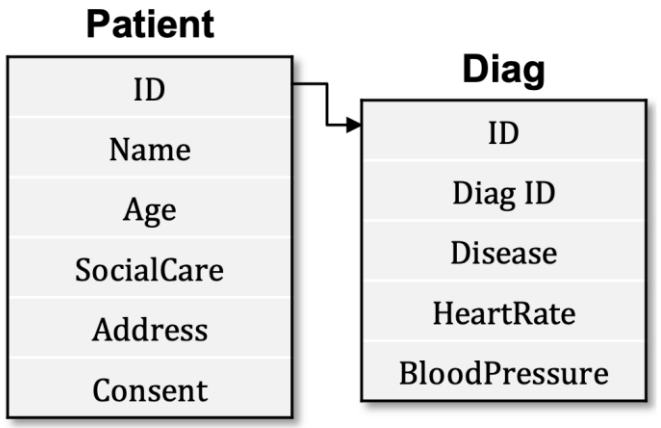
Defense 2. Enforcing Policy on Spark Plans



- **New policy check mechanism based on pattern-matching**
  - Provide a policy language for data owners to define their expectations into policies

Motivating Scenario: Targeted Clinical Trials

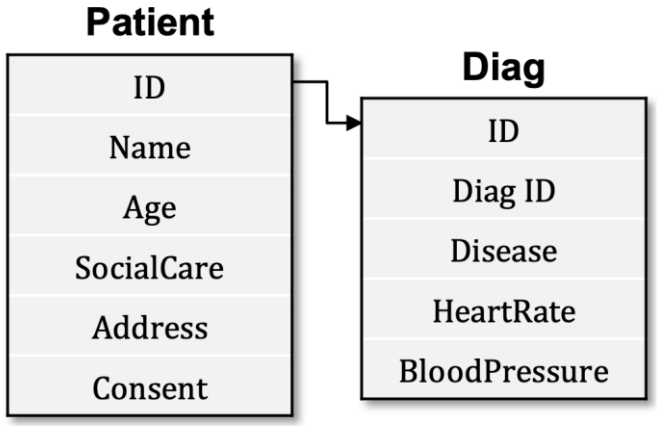
- **Hospital** wants to share medical dataset to **pharmaceutical company**



(a) DB schema of the medical dataset.

Motivating Scenario: Targeted Clinical Trials

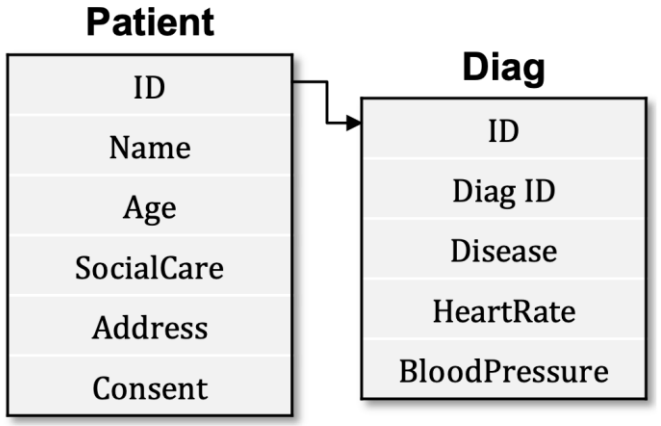
- **Hospital** wants to share medical dataset to **pharmaceutical company**
- Goal of **pharmaceutical company** (i.e., data user)
  - *Get the **Name** and **Address** of patients for their targeted drug testing*



(a) DB schema of the medical dataset.

Motivating Scenario: Targeted Clinical Trials

- **Hospital** wants to share medical dataset to **pharmaceutical company**
- Goal of **pharmaceutical company** (i.e., data user)
  - *Get the **Name** and **Address** of patients for their targeted drug testing*
- Expectation of **hospital** (i.e., data owner)
  - ***who** has been diagnosed with **which disease** should not be revealed*



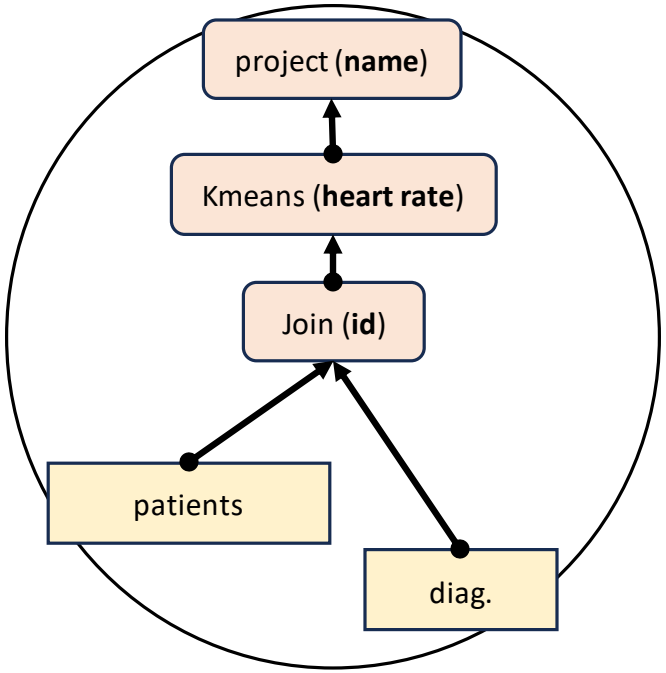
(a) DB schema of the medical dataset.

	Example policies
$\mathbf{P_1}$	Only the records of the patients who have consented can be used in machine learning.
$\mathbf{P_2}$	Patient's social care status and name must not be used in machine learning.
$\mathbf{P_3}$	When the `patient` and `diag` tables are joined, the patients' disease must not be revealed.

(b) Policies that the hospital wants to enforce.

Motivating Scenario: Targeted Clinical Trials

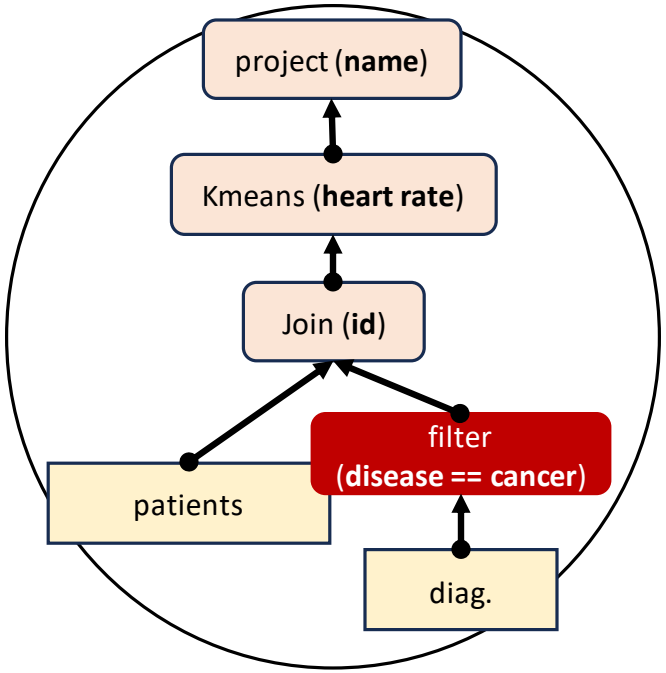
- Benign Spark plan



Benign Spark plan

Motivating Scenario: Targeted Clinical Trials

- Policy violating Spark plan
  - ***Filter diag*** on **disease** (== **cancer**) first, ***join***, and then ***project*** **name**

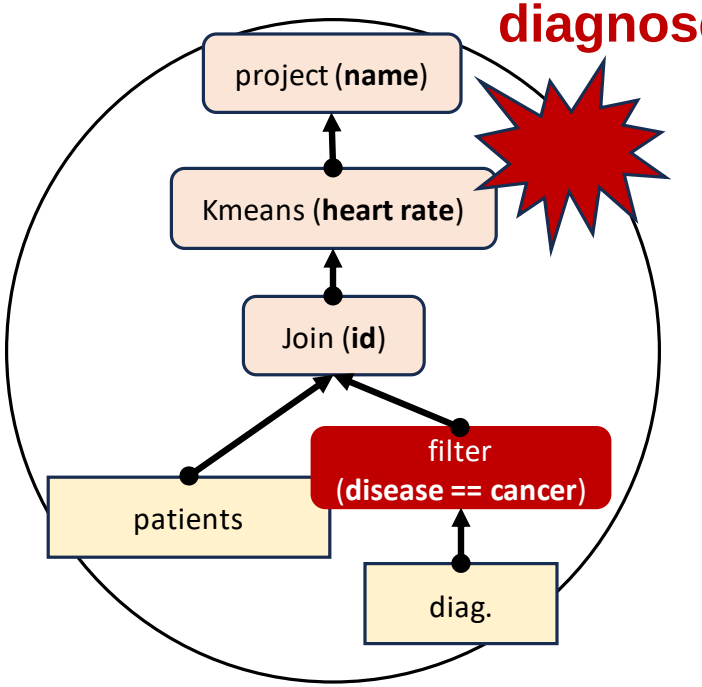


Policy breaking plan

Motivating Scenario: Targeted Clinical Trials

- Policy violating Spark plan
  - ***Filter `diag`*** on **disease** (== **cancer**) first, ***join***, and then ***project*** **name**

Exposing who has been diagnosed with cancer!

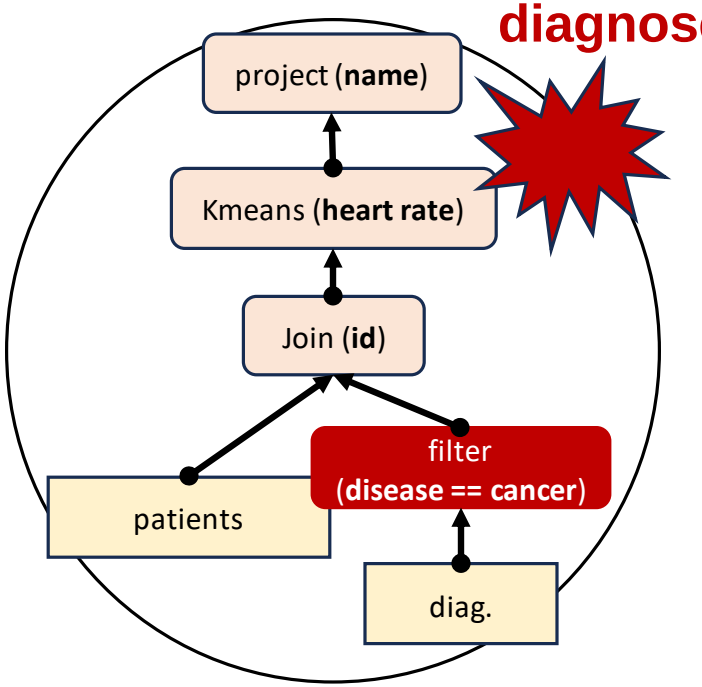


Policy breaking plan

Motivating Scenario: Targeted Clinical Trials

- Policy violating Spark plan
  - ***Filter diag*** on **disease** (== **cancer**) first, ***join***, and then ***project*** **name**

Exposing who has been diagnosed with cancer!



Policy breaking plan

Policy preventing this case

For table **diag.**

Disallow

$\bullet^*S_3\bullet^*S_1\bullet^* | \bullet^*S_1\bullet^*S_3\bullet^*$

S_3 = <**filter**, {**disease**}>

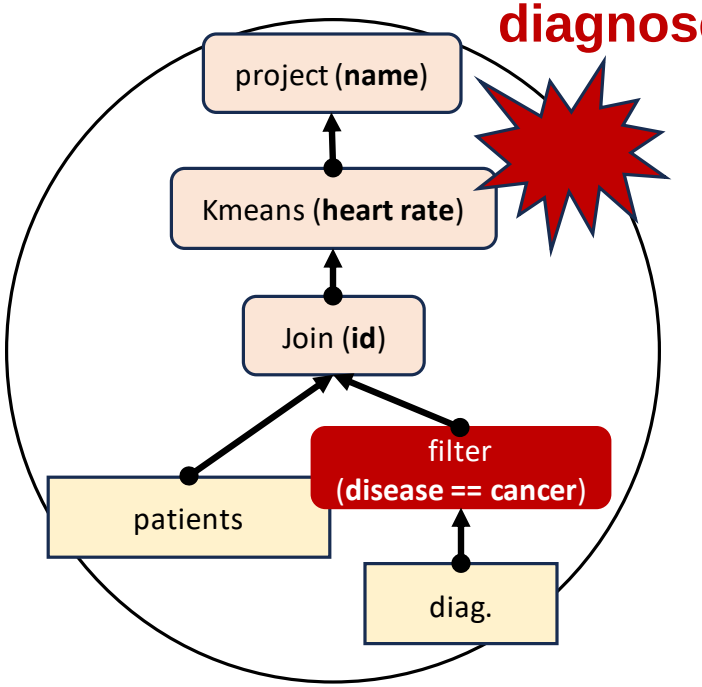
S_1 = <**join**, {**id**}>

Owner defined policy

Motivating Scenario: Targeted Clinical Trials

- Policy violating Spark plan
  - ***Filter diag*** on **disease** (== **cancer**) first, ***join***, and then ***project*** **name**

Exposing who has been diagnosed with cancer!



Policy breaking plan

Policy preventing this case

For table **diag.**

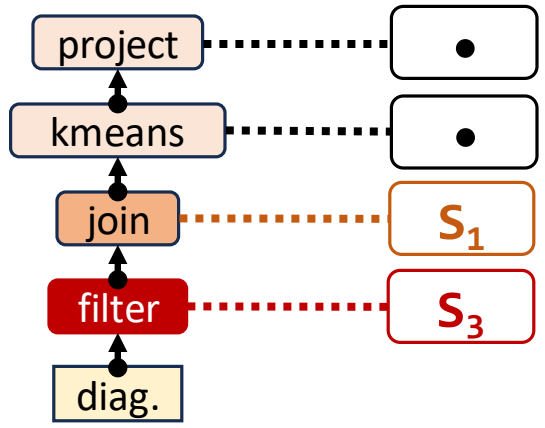
Disallow

$\bullet^* S_3 \bullet^* S_1 \bullet^* | \bullet^* S_1 \bullet^* S_3 \bullet^*$

S_3 = <**filter**, {**disease**}>

S_1 = <**join**, {**id**}>

Owner defined policy

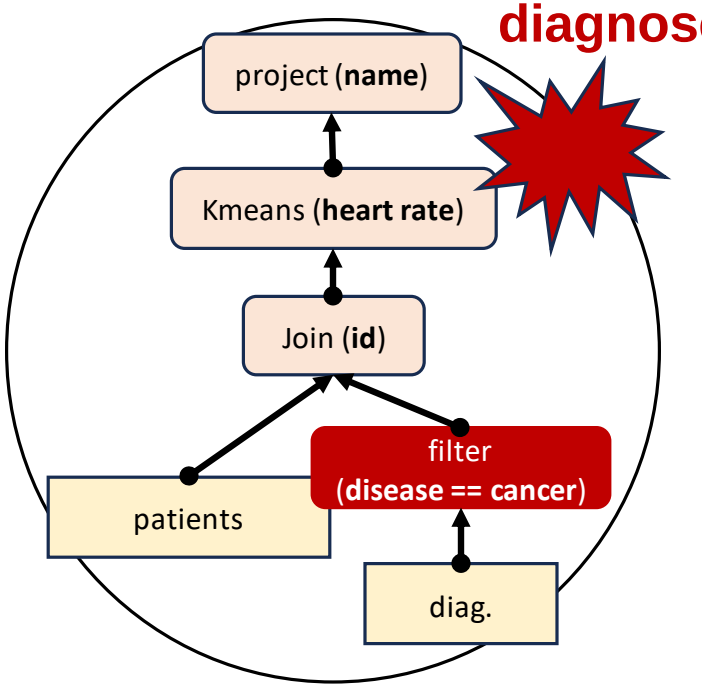


Pattern-matching

Motivating Scenario: Targeted Clinical Trials

- Policy violating Spark plan
  - ***Filter `diag`*** on **disease** (== **cancer**) first, ***join***, and then ***project*** **name**

Exposing who has been diagnosed with cancer!



Policy breaking plan

Policy preventing this case

For table **diag.**

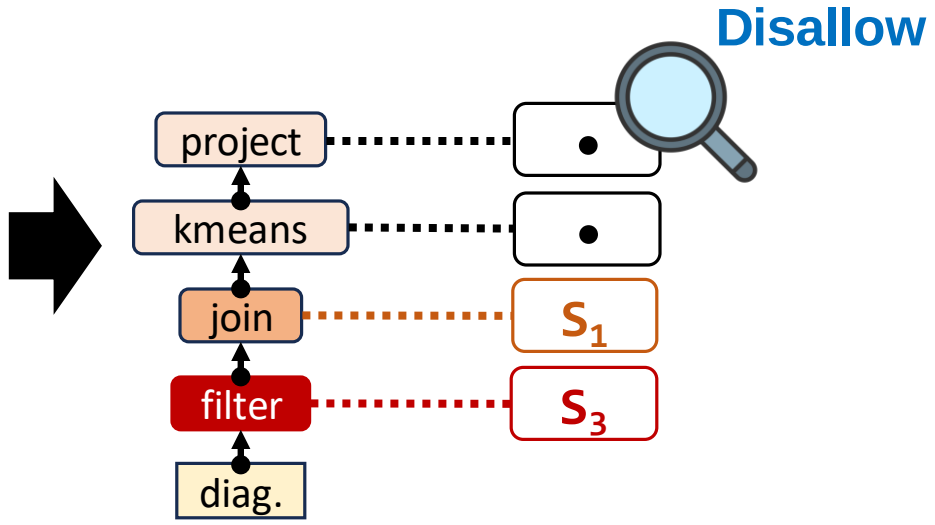
Disallow

$\bullet^* S_3 \bullet^* S_1 \bullet^* | \bullet^* S_1 \bullet^* S_3 \bullet^*$

S_3 = <**filter**, {**disease**}>

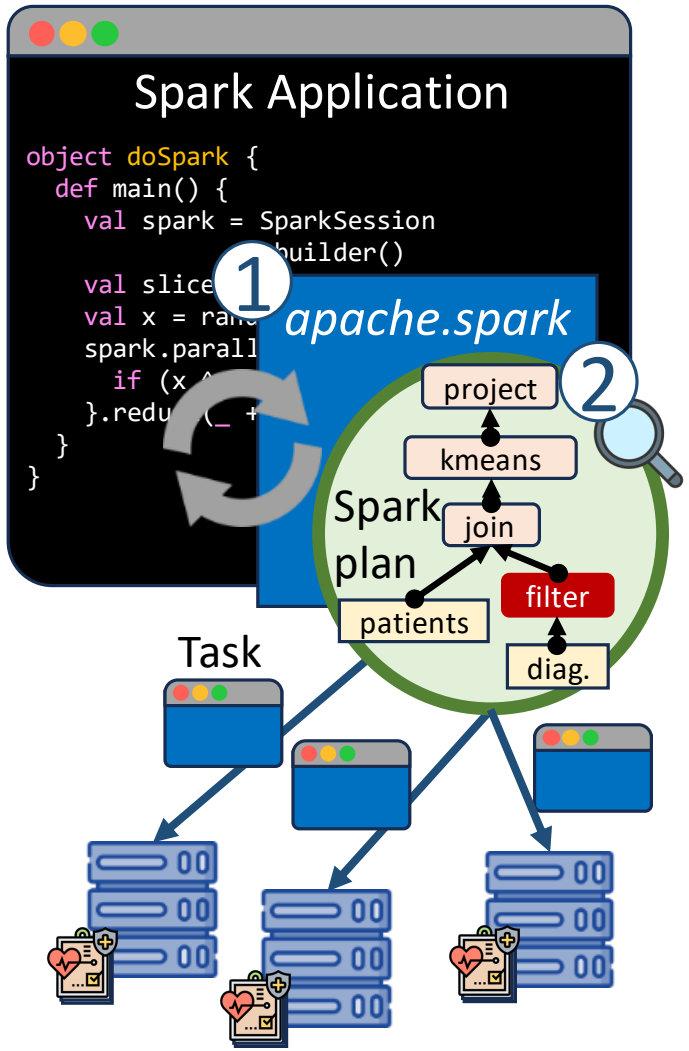
S_1 = <**join**, {**id**}>

Owner defined policy



Pattern-matching

Security Requirements for Protecting Data



- ✓ Ensure the confidentiality and integrity of the entire **data analysis pipeline**
- ✓ Ensure the Spark plans by data users respect the **owner-defined policies**

Implementation

- Compartmentalization
  - Each *SparkContext* on untrusted data user's side and ***trusted point***
  - ***Trusted point*** and distributed nodes protected by AMD SEV-SNP

Implementation

- Compartmentalization
  - Each *SparkContext* on untrusted data user's side and ***trusted point***
  - ***Trusted point*** and distributed nodes protected by AMD SEV-SNP

- Pattern-matching based policy check
  - Policy language defined on top of regular expression
  - Spark plans matched against the policies based on Regex matching

Evaluation

- Security Evaluation
  - Enforced 7 custom-defined policies and checked correctness on 22 queries from *TPC-H benchmark*

  - Example Policies
    I. *Personally identifiable information (e.g., name) must not be revealed*
    II. *Private information (e.g., account balance) must not be obtained after filtering on PII*
    III. *Sensitive identifiers (i.e., primary keys) can only be used for joining tables*

Evaluation

- Performance Evaluation
  - *TPC-H*, *BDB benchmark* and Spark ML applications
  - 35% latency & 25% throughput overheads on average

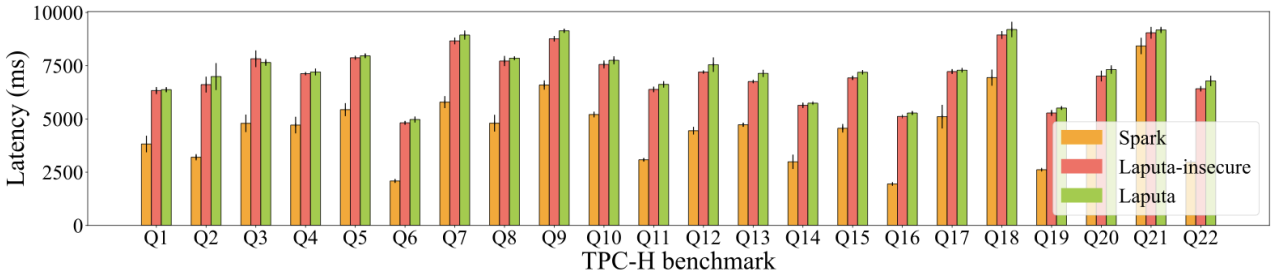
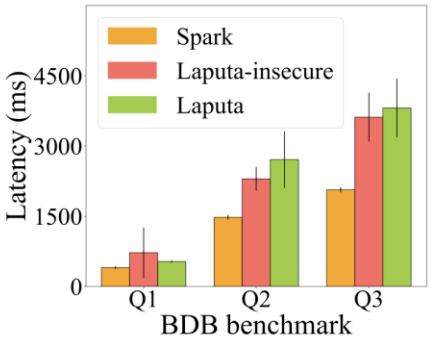
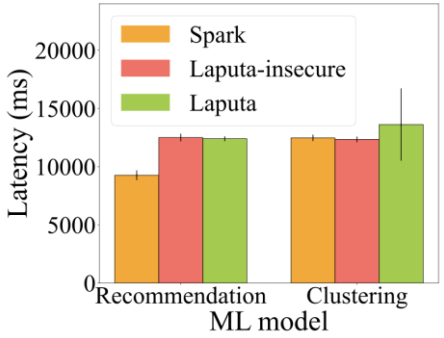


Figure 9: Increased latency on TPC-H benchmark.



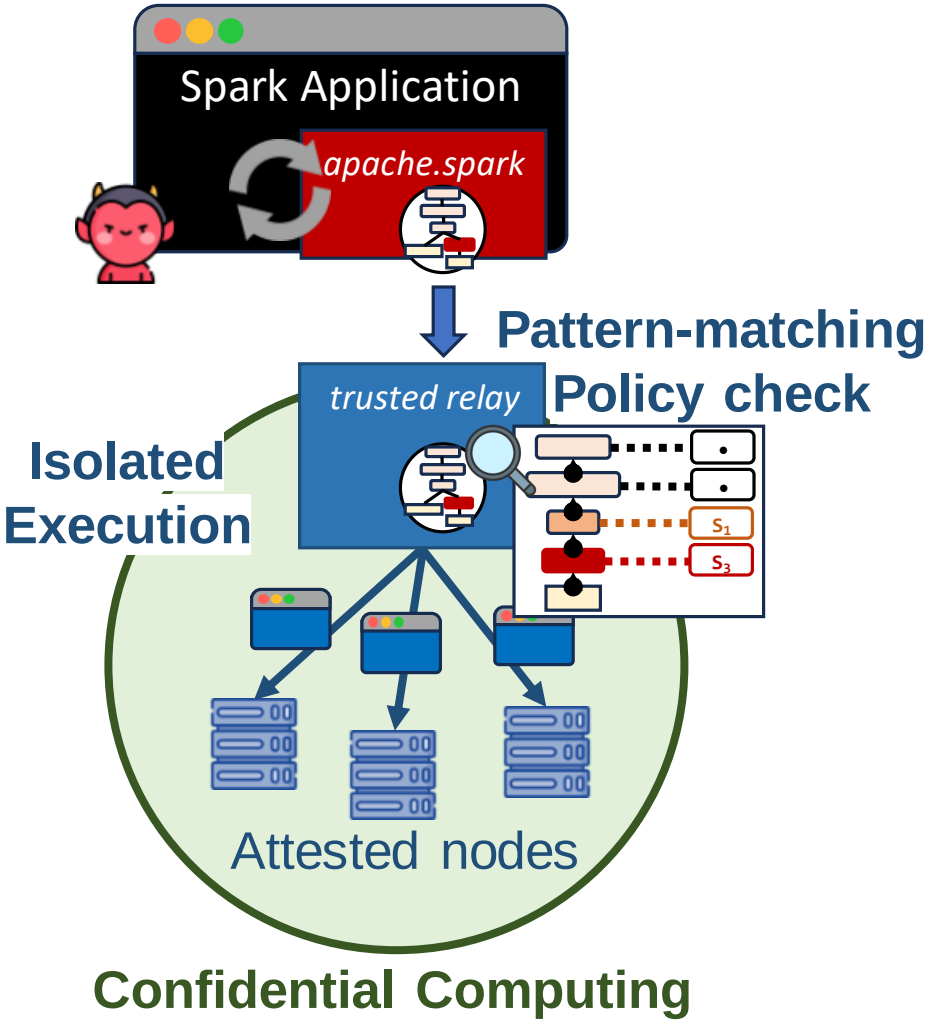
(a) Increased latency.



(a) Increased latency.

Conclusion

- We propose a new secure data analytics framework on Spark.
- Our framework can be used for **data owners to share their data without concerning the regulatory violation**.



Thank you