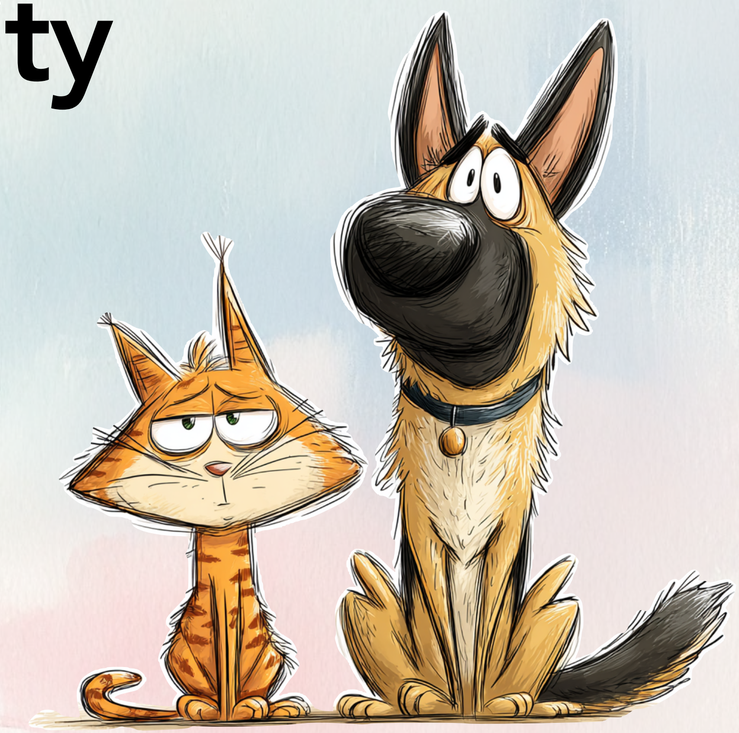
Deep dive into
data streaming security

by Olena Kutsenko,

CONFLUENT




Flink
CONFLUENT
Developer
ICEBERG
kafka

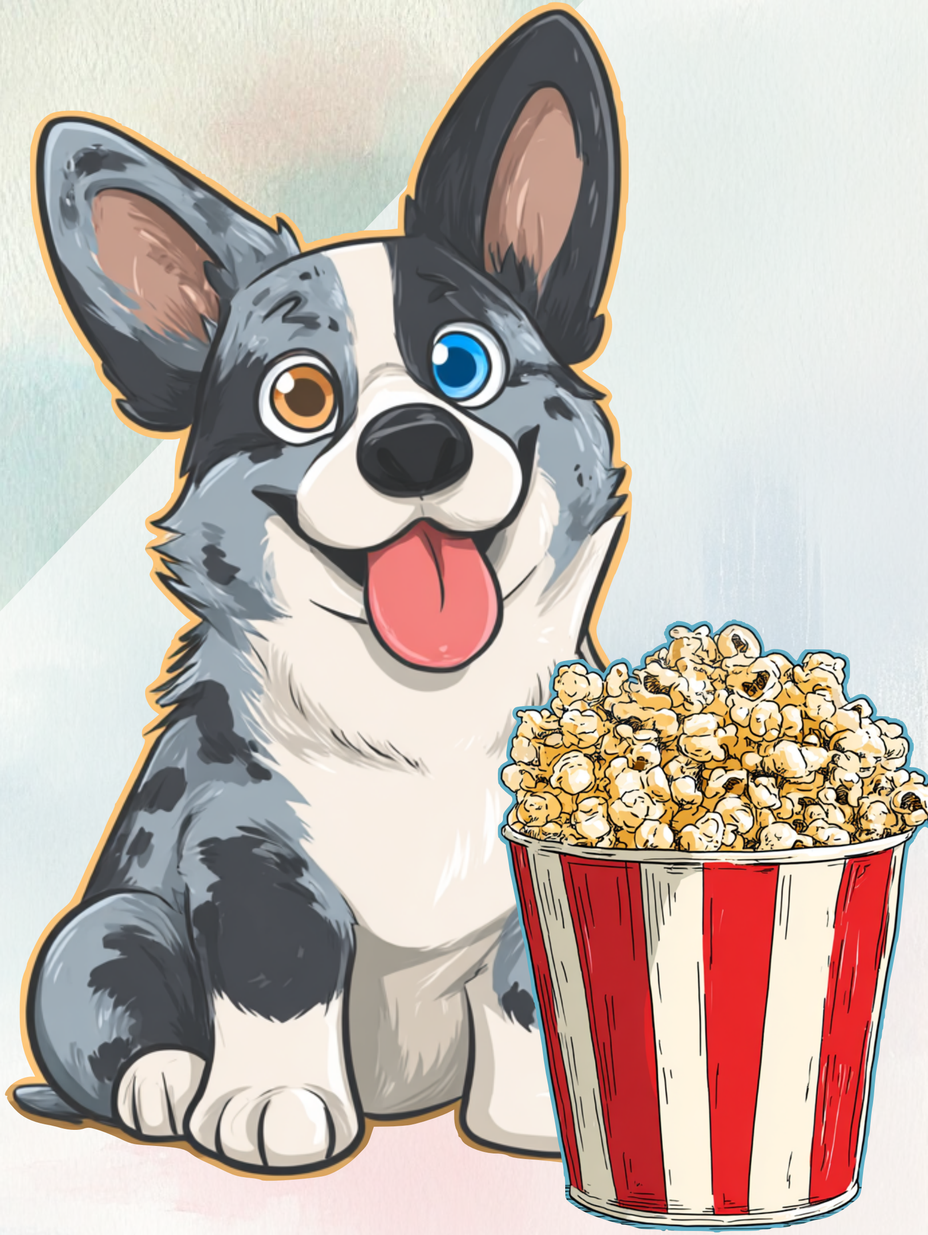
Data streaming

it’s powerful, but it’s **messy**

- Huge volumes
- peed
- ds of data
- ources
- scale
- It has to adapt
- It has to work everywhere

What can go wrong?

Security bloopers awards



Healthcare data leak through chatbot (2025)

The "Who needs authentication anyway?" award

- major player in Brazil's healthcare sector with an estimated 15 million clients
- sent over 14 million messages in an insecure way:
  - pictures
  - documents
  - messages
  - names
  - phone numbers
  - email addresses
  - company card numbers

https://cybernews.com/security/unimed-brazil-healthcare-data-leak/

Food delivery platform data leak (2025)

The "Data à la Carte" award

Cybernews team estimates the following details were exposed:

- Customer orders
- Restaurants and hotels where orders were made
- Customer phone numbers
- Email addresses
- Home addresses
- Delivery notes
- Payment methods used

https://cybernews.com/security/gonnaorder-food-delivery-data-leak/

Misconfigured Kafdrop (2021)

The "Peekaboo Panel" award

- exposed data due to misconfigurations
- not caused by the Kafdrop project itself
- supply chain vulnerabilities (external libraries, tools, services)

https://spectralops.io/blog/misconfigured-kafdrop-puts-companies-apache-kafka-completely-exposed/

Cryptominers targeting misconfigured Apache Hadoop and Apache Flink (2024)

The "Crypto Creep" award

- unauthenticated app deployment and then POST request to execute arbitrary code
- drops ‘dca’ binary, rootkits, and Monero cryptominer is written to the disk
- defense evasion: Packed ELF, rootkits, file deletion, config tampering, cron jobs redeploy ‘dca’ binary

https://www.aquasec.com/blog/threat-alert-apache-applications-targeted-by-stealthy-attacker/

Critical flaw in Apache Parquet allows remote attackers to execute arbitrary code (2025)

The "Schema Screamer" award

- Schema parsing in the parquet-avro module of Apache Parquet 1.15.0 and previous versions allows bad actors to execute arbitrary code
- CVE-2025-30065
- score of 9.8

https://avd.aquasec.com/nvd/2025/cve-2025-30065/

Vulnerability in Apache Pulsar allowed manipulator-in-the-middle attacks (2022)

The "Inner Trust Betrayal" award

- Pulsar’s components shipped with hostname verification disabled by default

https://pulsar.apache.org/security/CVE-2022-33682/

CVE-2024-27309 Potential incorrect access control during migration from ZK mode to KRaft mode

CVE-2023-25194 Possible RCE/Denial of service attack via SASL JAAS JndiLoginModule configuration using Apache Kafka Connect API #

CVE-2025-27818 Apache Kafka: Possible RCE attack via SASL JAAS LdapLoginModule configuration

CVE-2022-23302 Deserialization of Untrusted Data Flaw in JMSSink of Apache Log4j logging library in versions 1.x #

CVE-2024-56128 SCRAM authentication vulnerable to replay attacks when used without encryption #

CVE-2025-27819 Apache Kafka: Possible RCE/Denial of service attack via SASL JAAS JndiLoginModule configuration #

CVE-2025-27819 Apache Kafka: Possible RCE/Denial of service attack via SASL JAAS JndiLoginModule configuration #

CVE-2024-56128 SCRAM authentication vulnerable to replay attacks when used without encryption #

CVE-2022-34917 Unauthenticated clients may cause OutOfMemoryError on brokers

CVE-2023-34455 Clients using Snappy compression may cause out of memory error on brokers

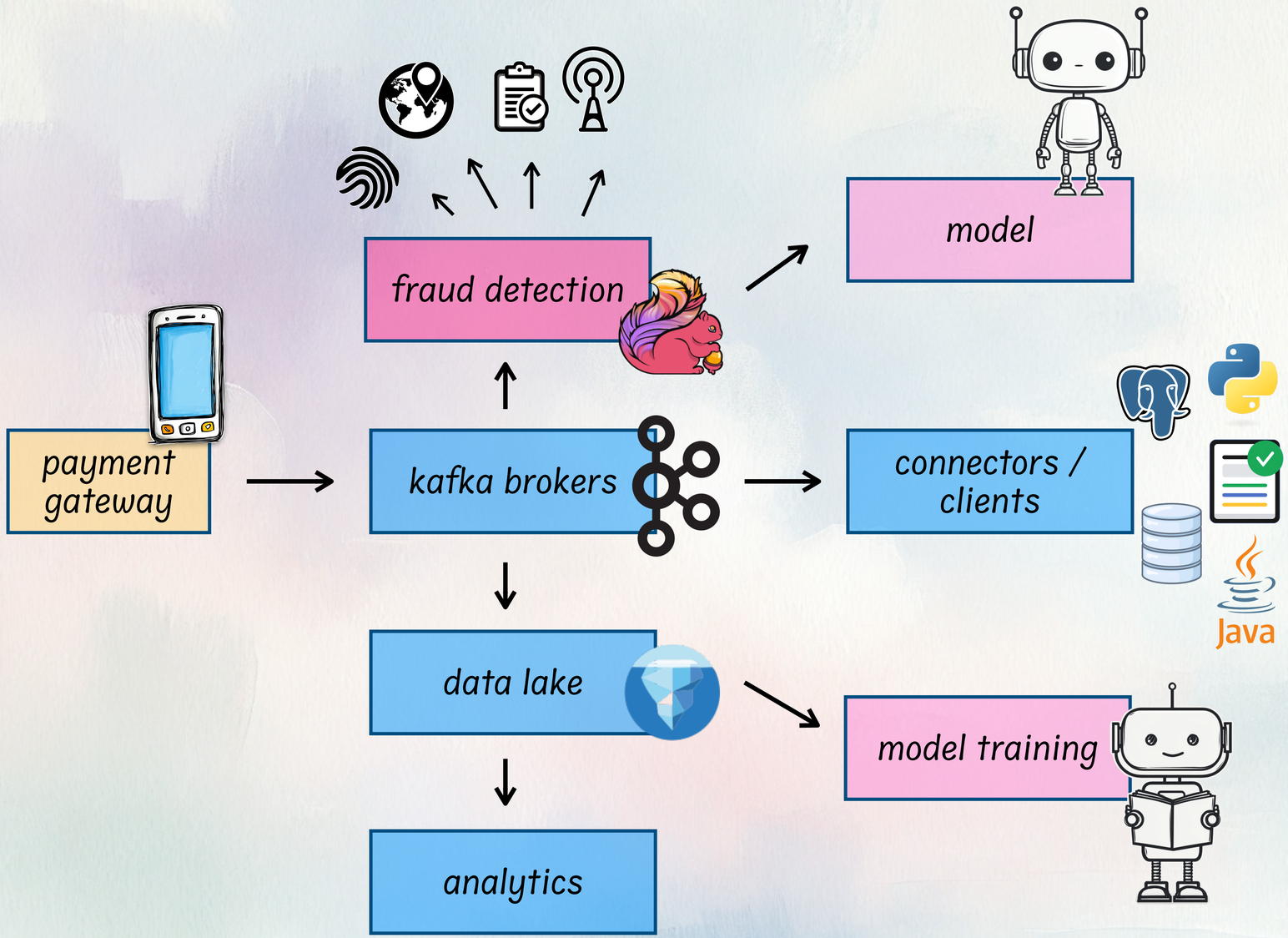
CVE-2024-31141 Files or Directories Accessible to External Parties, Improper Privilege Management vulnerability in Apache Kafka Clients

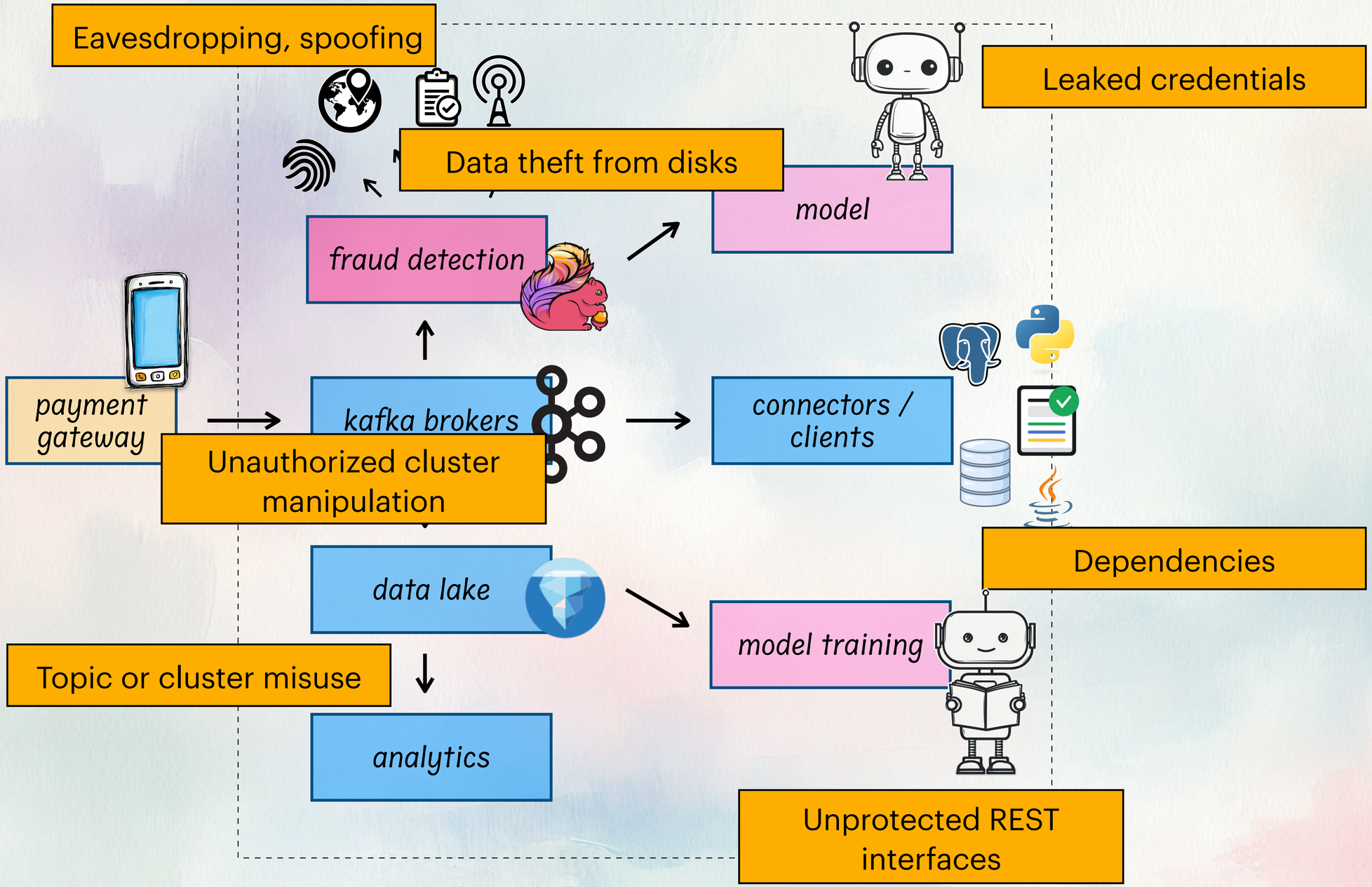
The stream defender’s checklist

Don't get into the news

- Avoid misconfigurations
- Patch & update fast
- Verify your dependencies & supply chain integrity

Use case


payment gateway
fraud detection
kafka brokers
data lake
analytics
model
connectors / clients
model training
Java


Eavesdropping, spoofing
Data theft from disks
Leaked credentials
model
fraud detection
payment gateway
kafka brokers
connectors / clients
Unauthorized cluster manipulation
Dependencies
data lake
model training
Topic or cluster misuse
analytics
Unprotected REST interfaces

kafka - open-source distributed event streaming platform

- Millions of messages per second
- With millisecond latency
- At petabyte scale
- With built-in fault tolerance

produce

topic

partitions

1
2
3
...

brokers

message

leader

followers

1
4
5
...

1
7
8
...

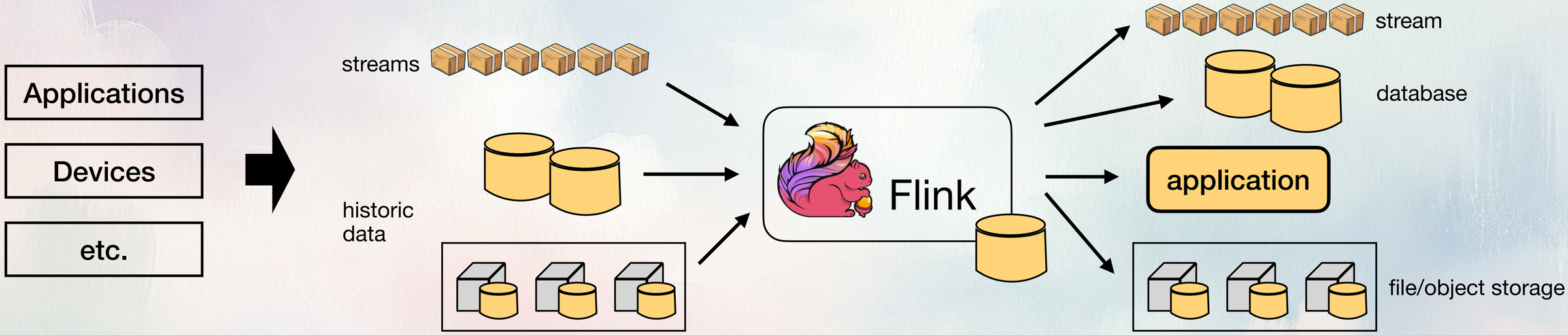
consume

Kafka Connect

Java

MongoDB.

Apache Flink is a framework and distributed processing engine for stateful computations



Apache Iceberg

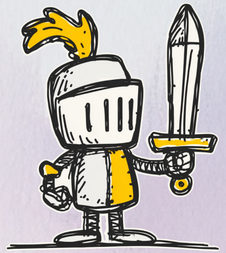
Data Warehouse

- structured
- cleaned data
- BI reporting

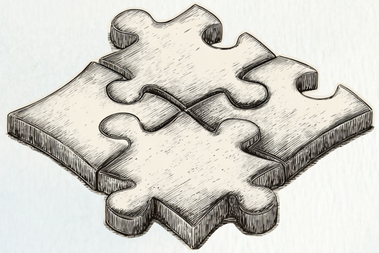
Data Lake

- raw
- diverse data
- ML/AI
- large-scale storage

Security principles in action



Defense in depth



Separation of duties

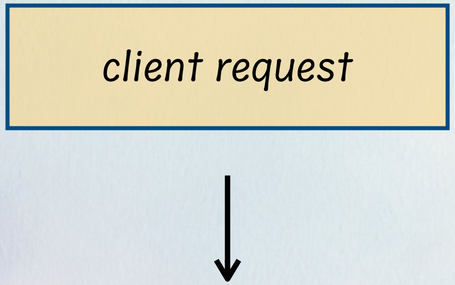
Least privilege

Security by design

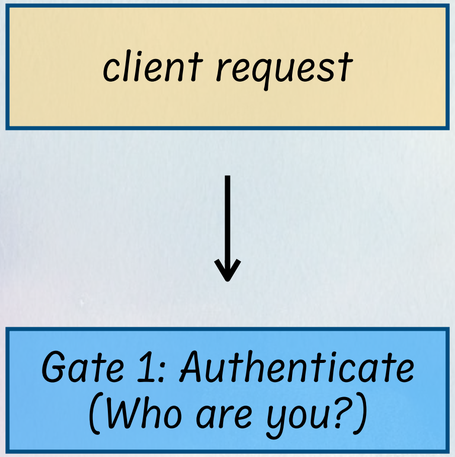
CIA

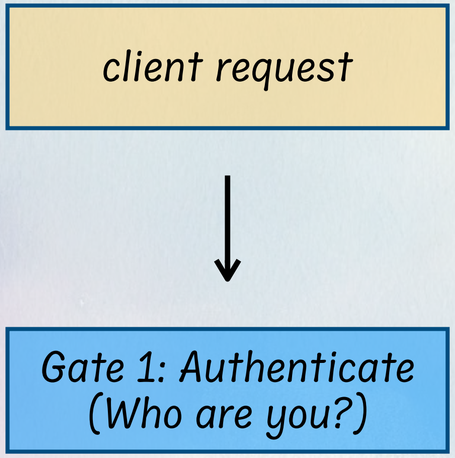
Transparency over obscurity

Tools: least privilege


client request

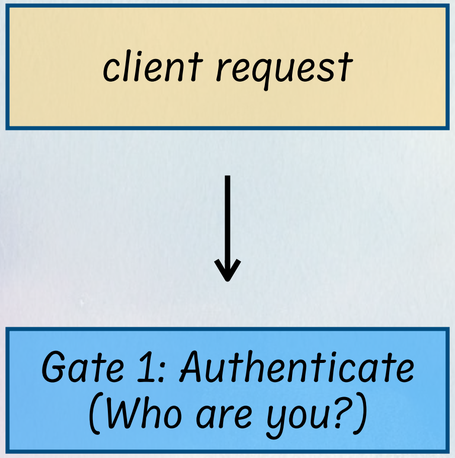
Establish principal identity





Establish principal identity

- mTLS
  - mutual authentication by certificates
  - no passwords, self-contained
  - good against token theft
  - no internet/no external dependency



Establish principal identity

- mTLS
  - mutual authentication by certificates
  - no passwords, self-contained
  - good against token theft
  - no internet/no external dependency

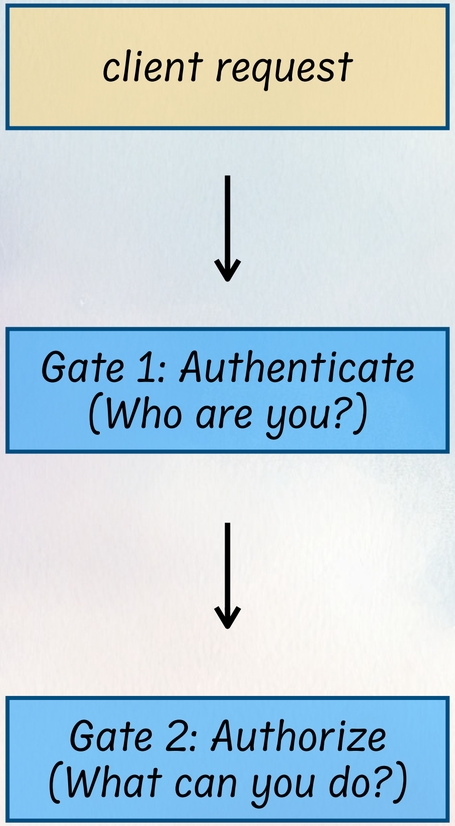
- OAuth 2.0 (SASL/OAUTHBEARER)
  - short-lived tokens
  - cloud-native
  - integrates with IdPs (Okta, etc)
  - stronger in dynamic, modern systems
  - fine-grained scopes
  - allows revocation

client request

Gate 1: Authenticate
(Who are you?)

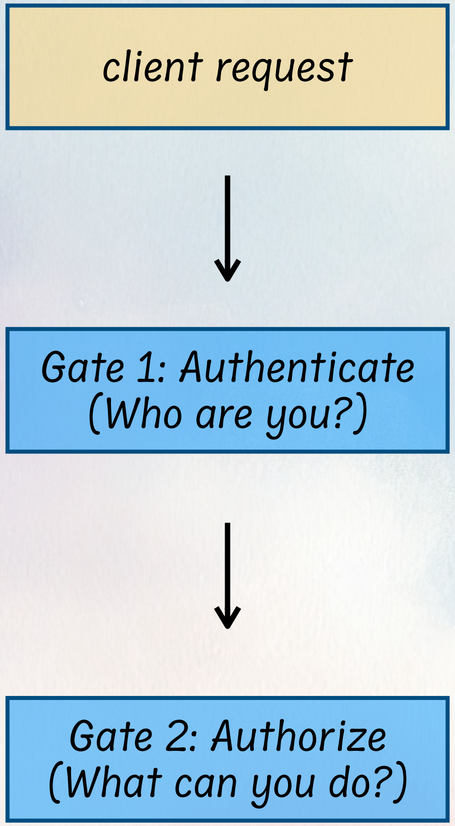
Gate 2: Authorize
(What can you do?)

check ACLs for operation+resource



check ACLs for operation+resource

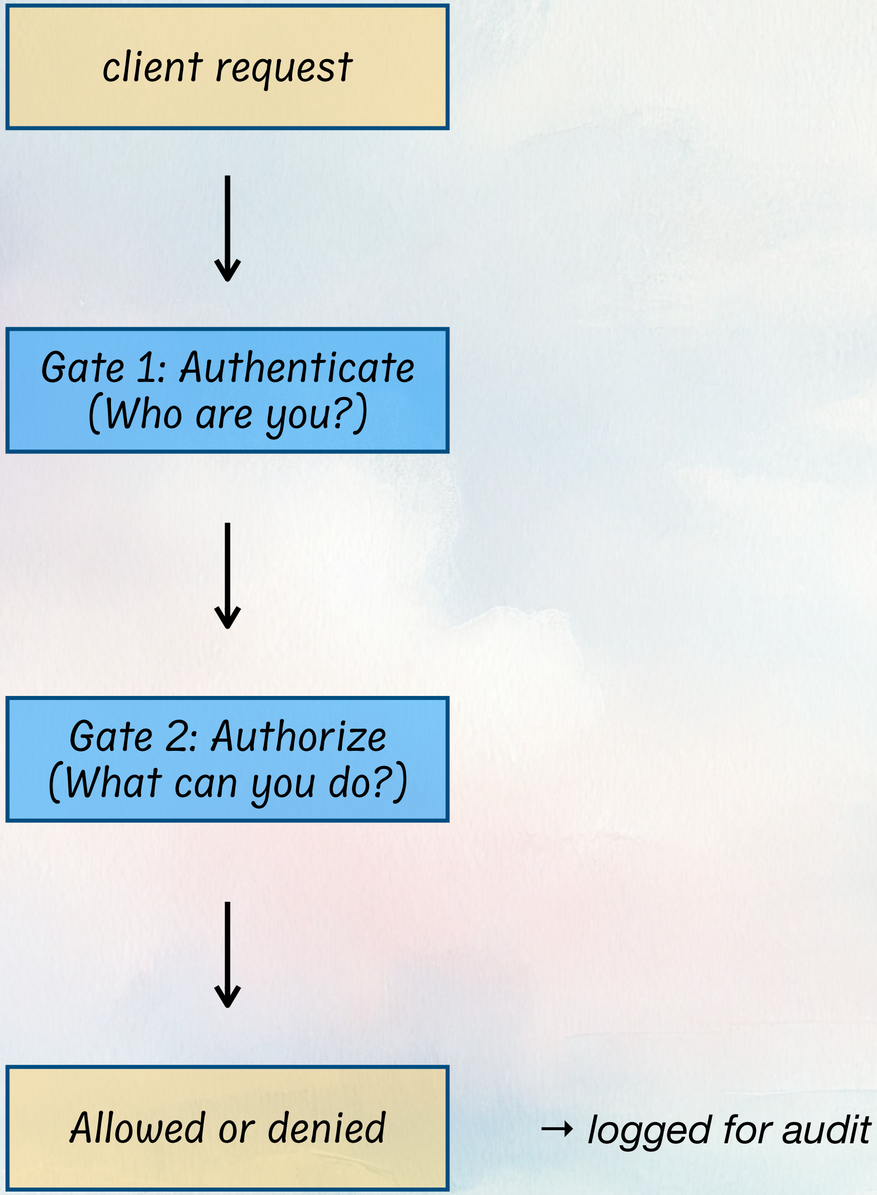
- ACLs (Access Control Lists)
- define permissions for:
  - Principal (User:alice)
  - Operation
  - Resource
  - Permission type (Allow/Deny)

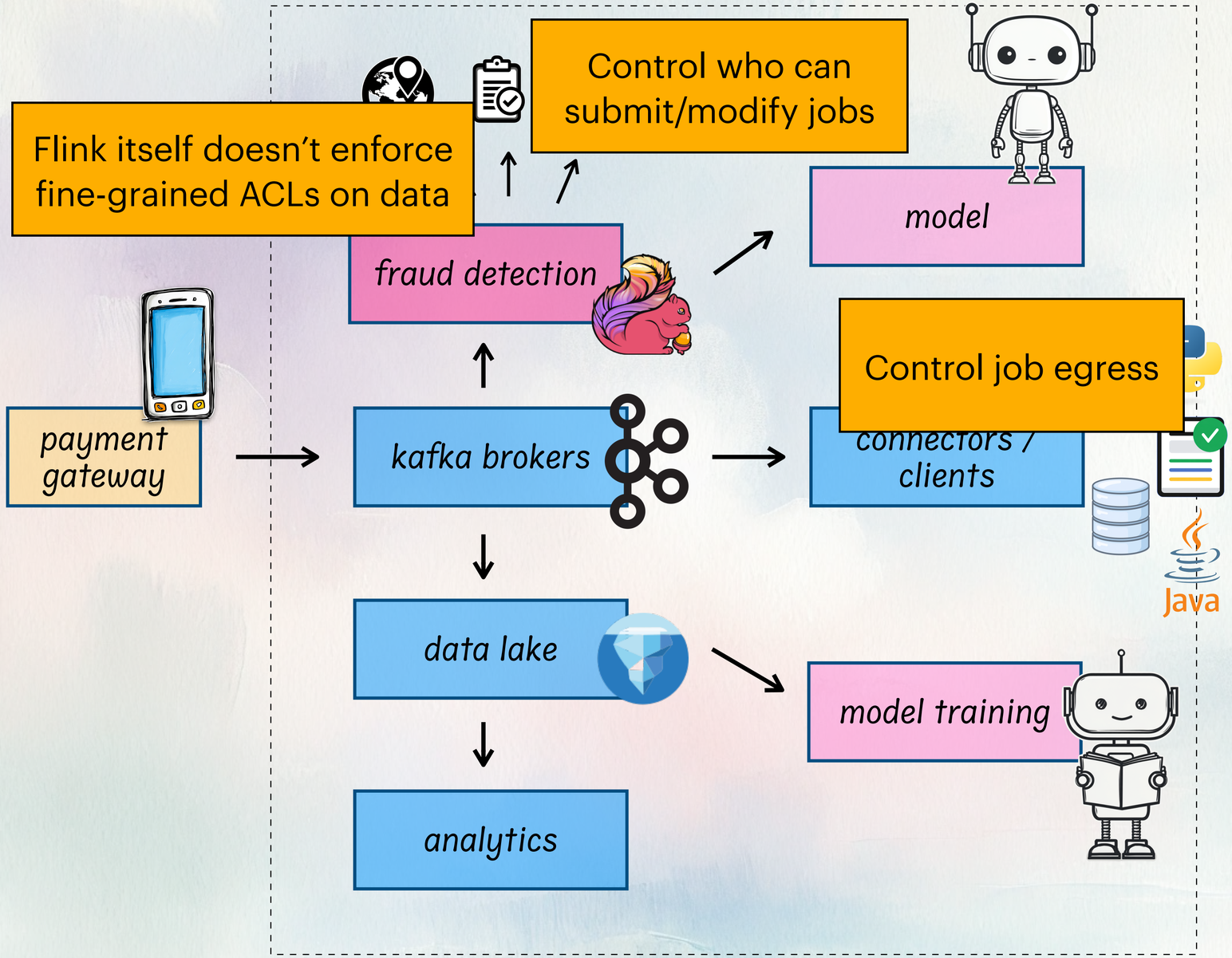


check ACLs for operation+resource

- ACLs (Access Control Lists)
- define permissions for:
  - Principal (User:alice)
  - Operation
  - Resource
  - Permission type (Allow/Deny)

```
kafka-acls --add \
 --allow-principal User:fraud-service \
 --operation WRITE \
 --topic payments
```


client request
Gate 1: Authenticate
(Who are you?)
Gate 2: Authorize
(What can you do?)
Allowed or denied
→ logged for audit



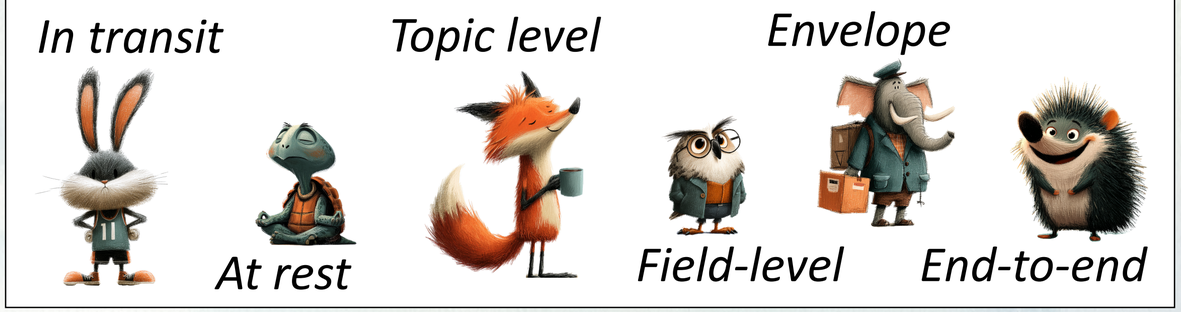
- who can deploy jobs?
- under what identity they run?
- what systems those jobs are allowed to reach?

The stream defender’s checklist

Don't get into the news

- Avoid misconfigurations
- Patch & update fast
- Verify your dependencies & supply chain integrity

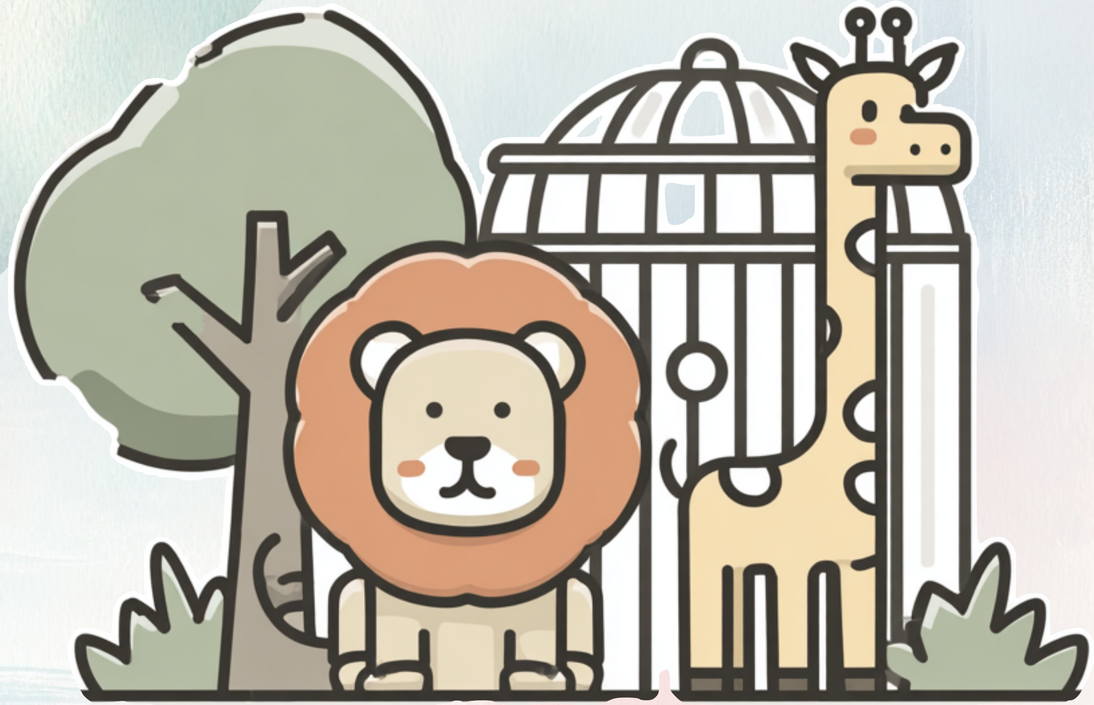
Encryption - select your favorite animals

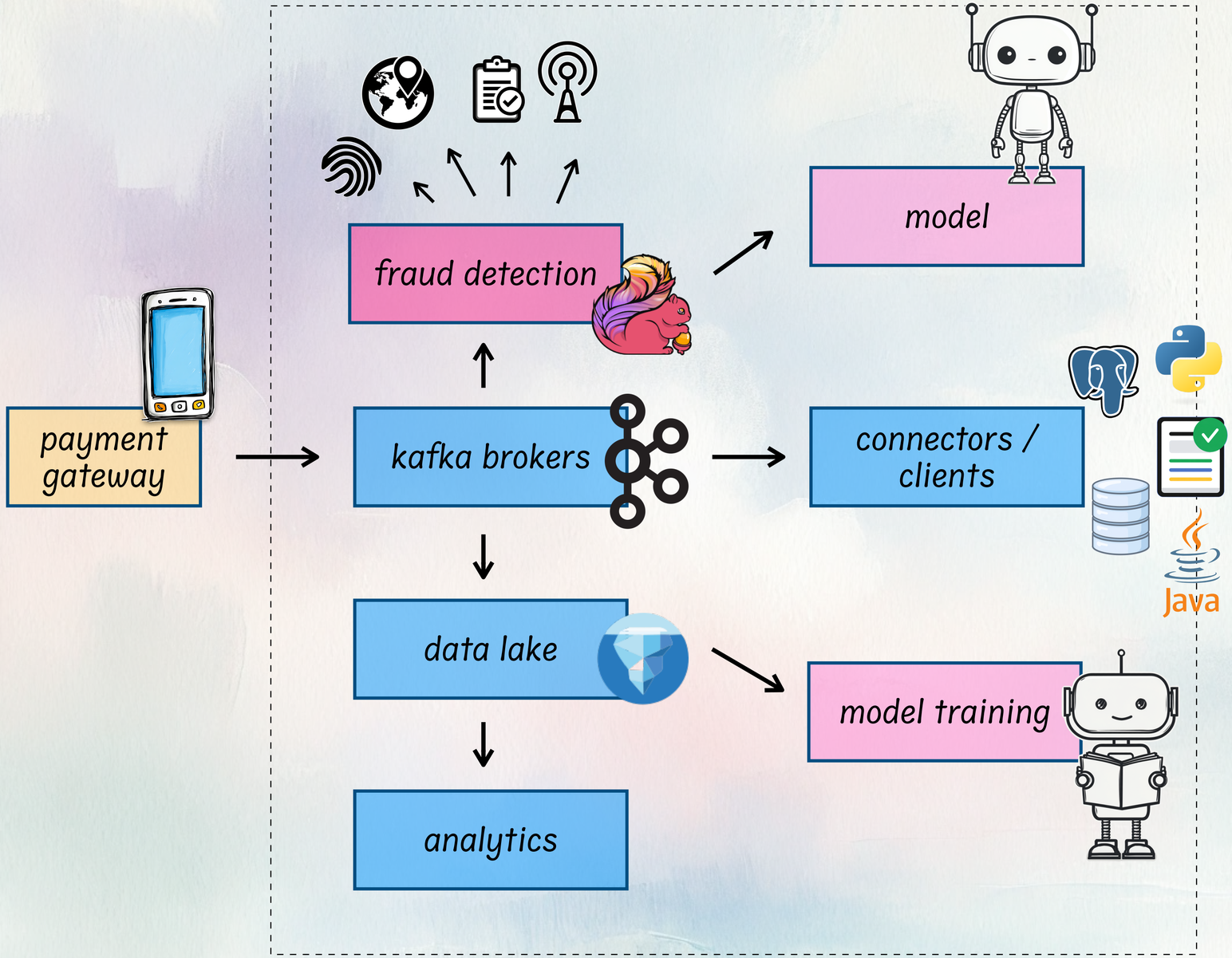


Least privilege

- Go for mTLS or OAuth over older protocols
- Access control defines the blast radius
- Every job has its own identity.
- Authorization without auditing is blind

Tools: encryption




payment gateway
fraud detection
kafka brokers
data lake
analytics
model
connectors / clients
model training
Java

1. Encryption in transit

2. Encryption at rest

3. Topic-level encryption

4. Field-level encryption

5. Envelope encryption

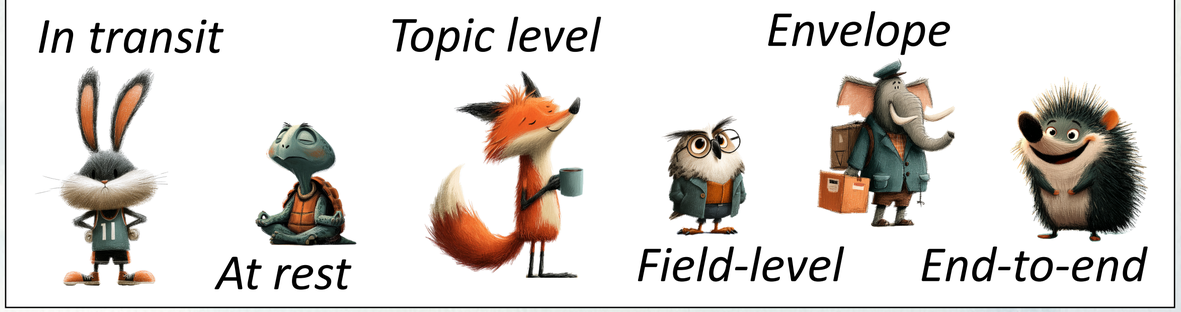
6. End-to-end encryption

The stream defender’s checklist

Don't get into the news

- Avoid misconfigurations
- Patch & update fast
- Verify your dependencies & supply chain integrity

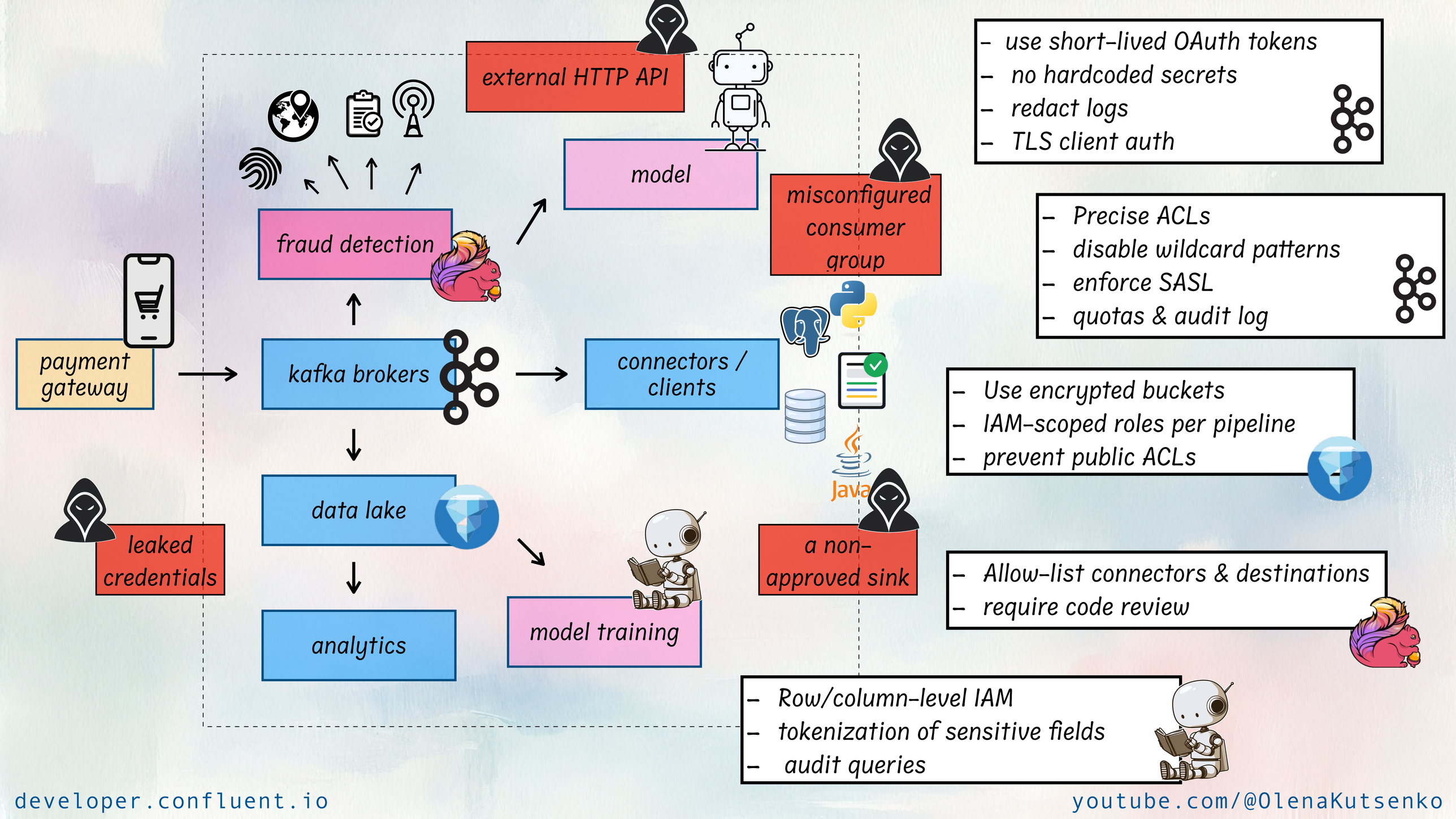
Encryption - select your favorite animals



Least privilege

- Go for mTLS or OAuth over older protocols
- Access control defines the blast radius
- Every job has its own identity.
- Authorization without auditing is blind

Risk:
data exfiltration

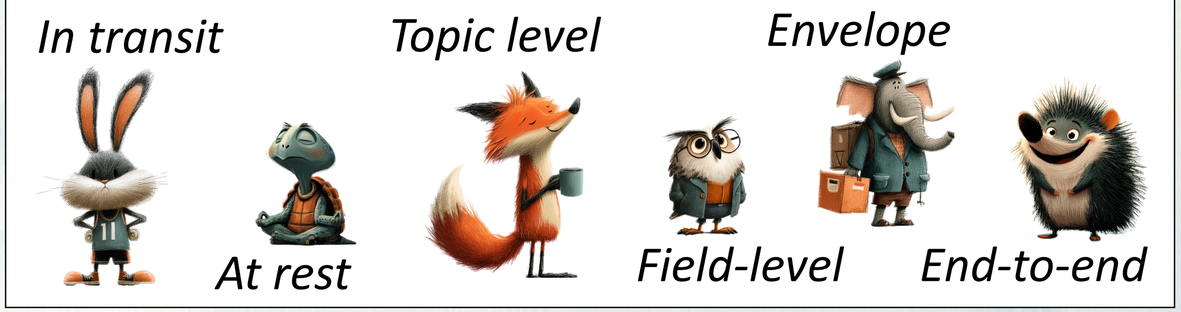

external HTTP API
model
fraud detection
misconfigured consumer group
payment gateway
kafka brokers
connectors / clients
data lake
leaked credentials
analytics
model training
a non-approved sink
Java
- use short-lived OAuth tokens
- no hardcoded secrets
- redact logs
- TLS client auth
- Precise ACLs
- disable wildcard patterns
- enforce SASL
- quotas & audit log
- Use encrypted buckets
- IAM-scoped roles per pipeline
- prevent public ACLs
- Allow-list connectors & destinations
- require code review
- Row/column-level IAM
- tokenization of sensitive fields
- audit queries

The stream defender’s checklist

Don't get into the news

- Avoid misconfigurations
- Patch & update fast
- Verify your dependencies & supply chain integrity

Encryption - select your favorite animals



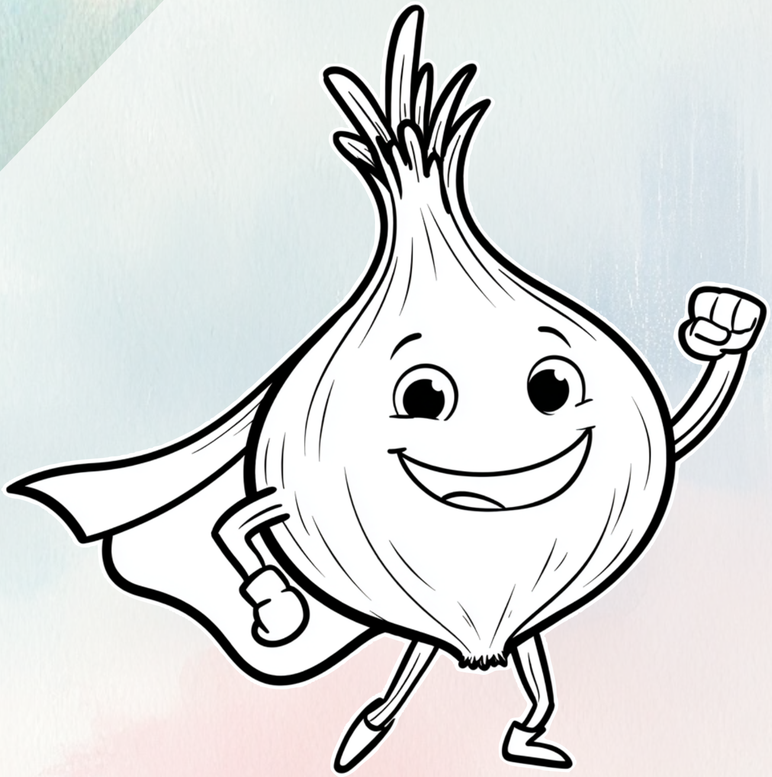
Data exfiltration

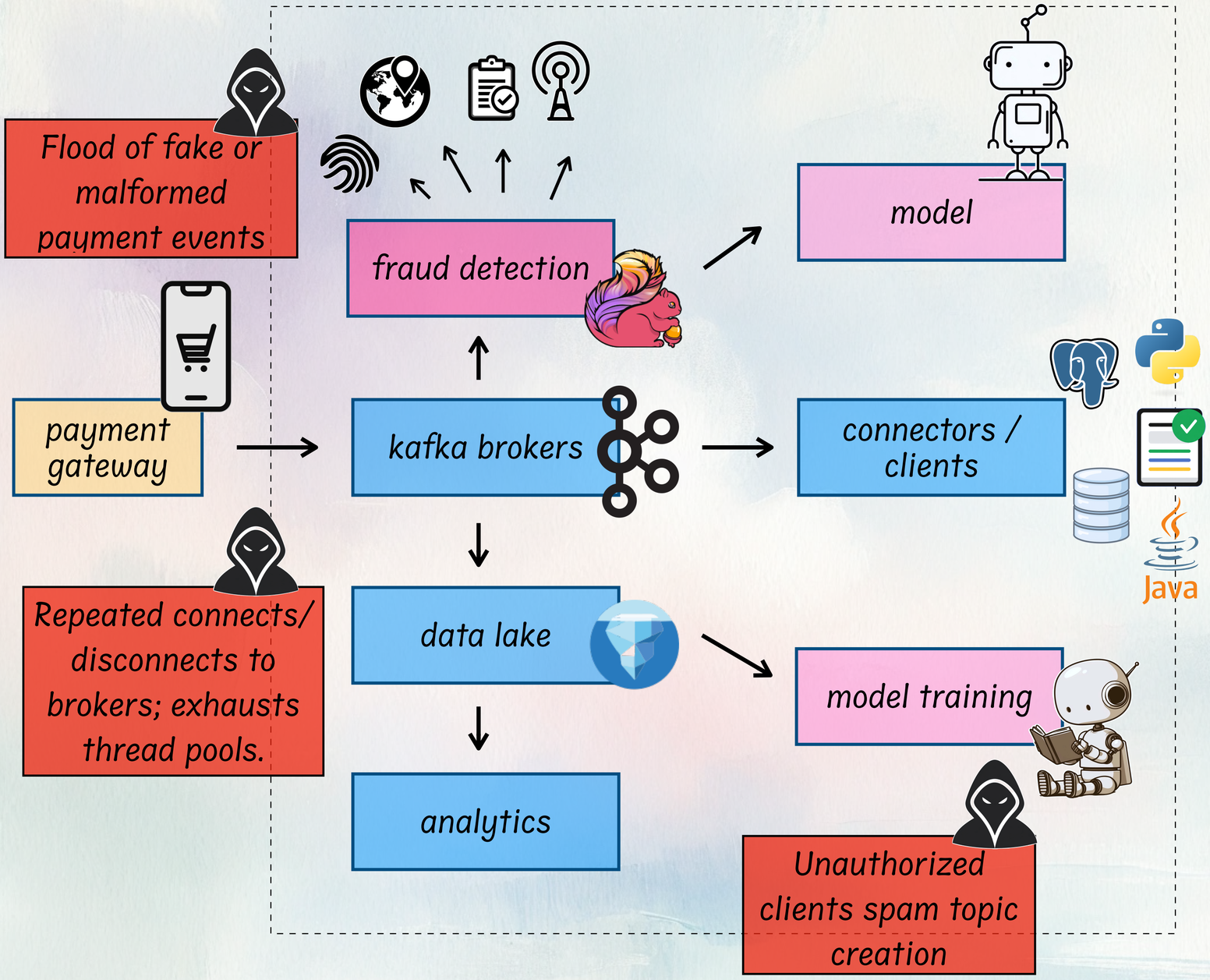
- The real threat may already be authenticated
- Shrink what can be stolen
- Every read leaves a trail

Least privilege

- Go for mTLS or OAuth over older protocols
- Access control defines the blast radius
- Every job has its own identity.
- Authorization without auditing is blind

Risk: DDoS





- too many TCP connections / TLS handshakes
- too many metadata requests / auth attempts
- too many fetch/produce requests from abusive clients

- Don't expose brokers to the Internet
- Controlled entry point (NLB + proxy/ gateway) that can shed abusive connections early
- Enforce mTLS/SASL, connection quotas, and sane broker/network timeouts so floods don't starve real traffic

The stream defender’s checklist

Don't get into the news

- Avoid misconfigurations
- Patch & update fast
- Verify your dependencies & supply chain integrity

DDoS

- A DDoS in streaming isn’t about downtime - it’s about delay
- Quotas, private links, and client isolation for defense
- You can’t fight what you don’t see

Least privilege

- Go for mTLS or OAuth over older protocols
- Access control defines the blast radius
- Every job has its own identity.
- Authorization without auditing is blind

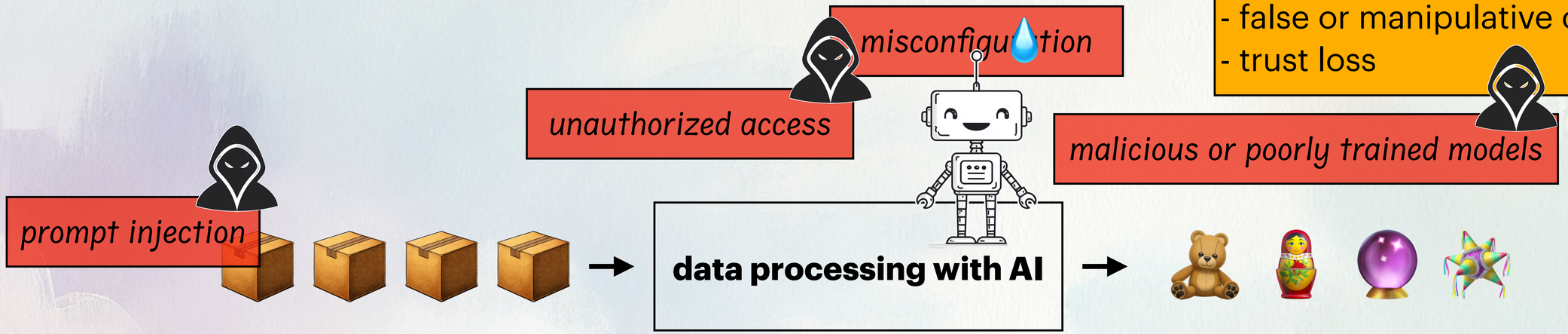
Encryption - select your favorite animals

Data exfiltration

- The real threat may already be authenticated
- Shrink what can be stolen
- Every read leaves a trail

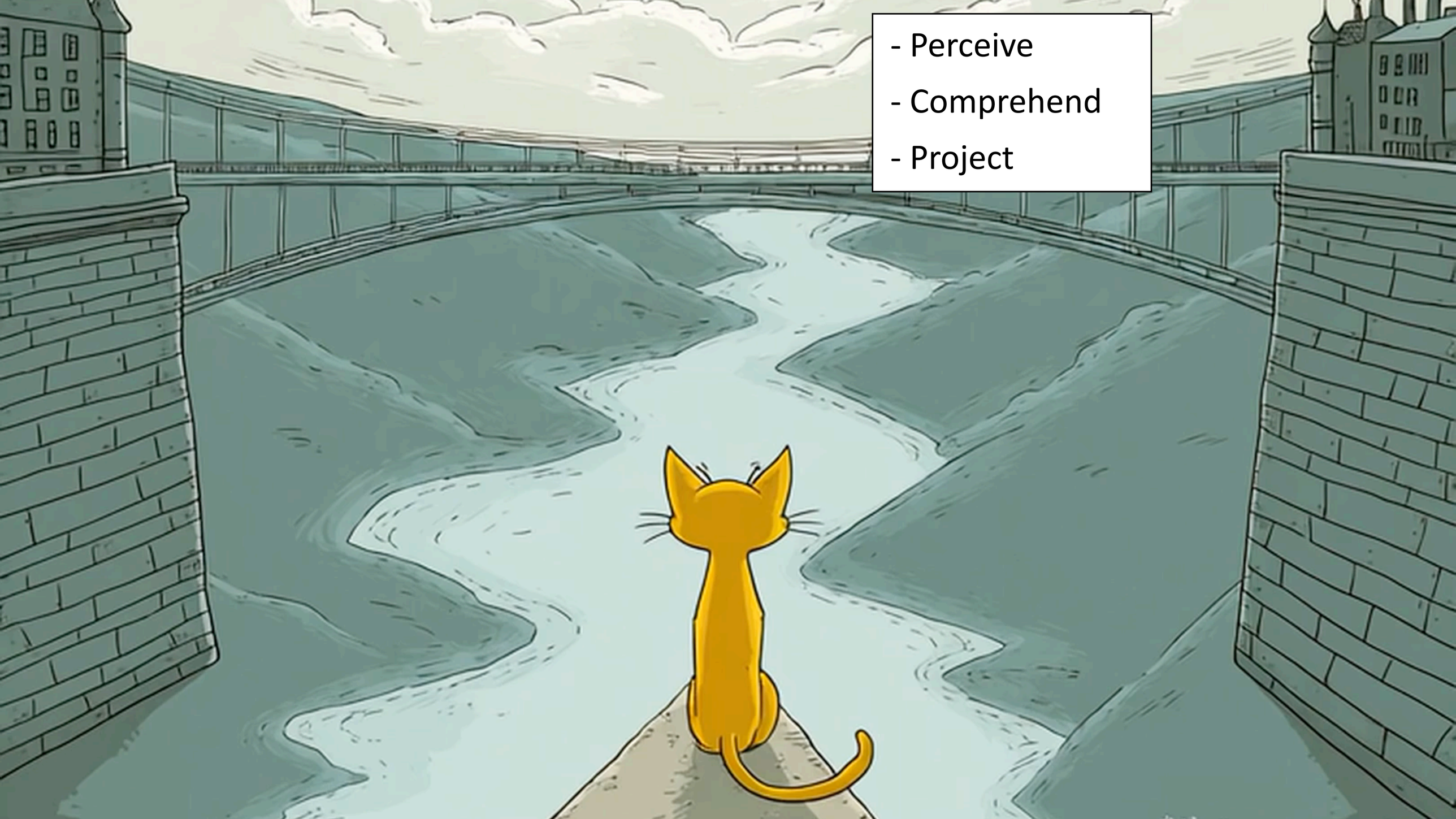
Risk: AI

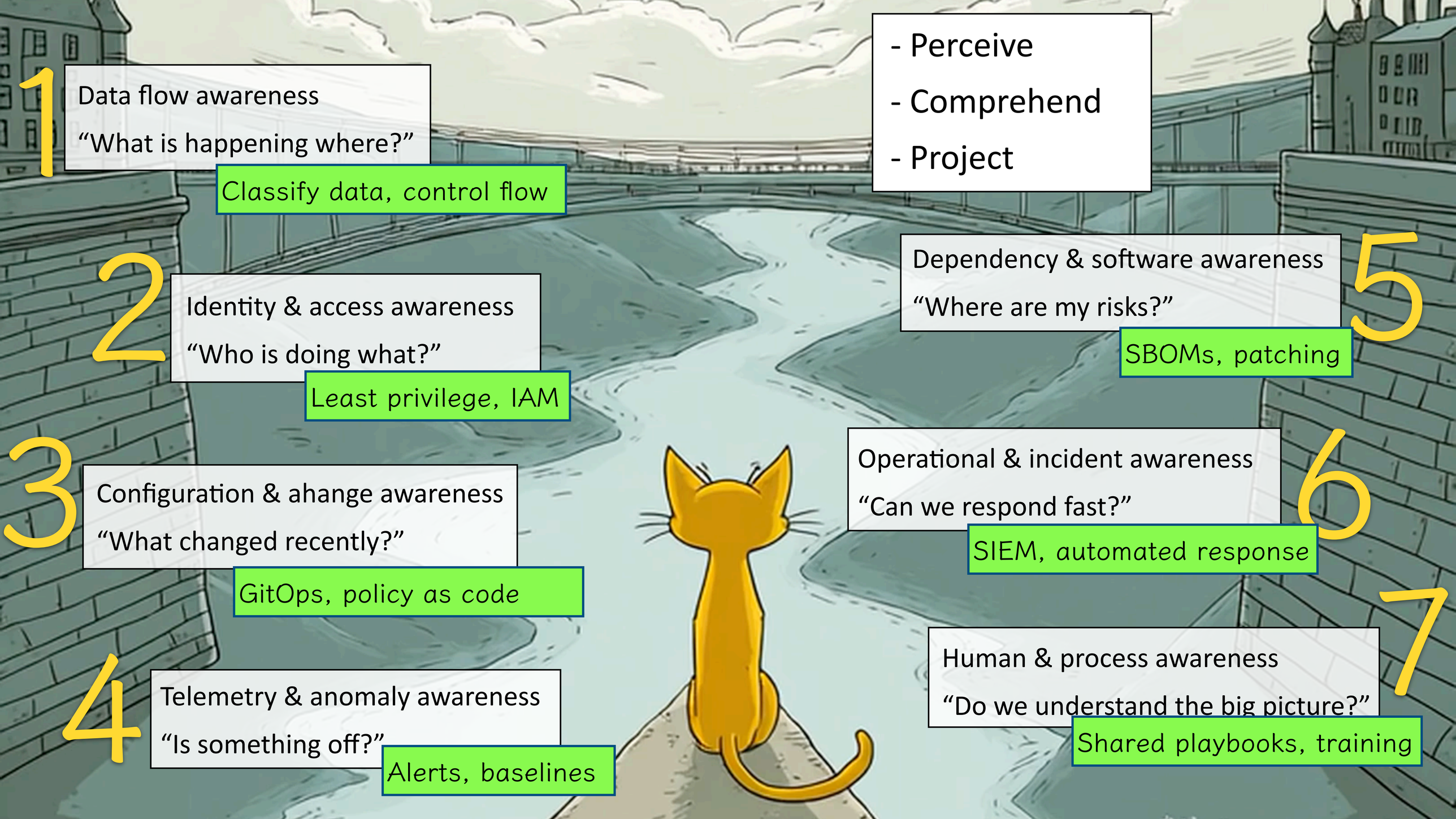
- data leaks
- intellectual property theft
- false or manipulative outputs
- trust loss



- Verify -> trust
- Just in case -> Just in time
- Perimeter-based control -> throughout the system
- Assume that bad guy is already in your system

Tools: situational awareness


- Perceive
- Comprehend
- Project


- Perceive
- Comprehend
- Project
1
Data flow awareness
“What is happening where?”
Classify data, control flow
2
Identity & access awareness
“Who is doing what?”
Least privilege, IAM
3
Configuration & ahange awareness
“What changed recently?”
GitOps, policy as code
4
Telemetry & anomaly awareness
“Is something off?”
Alerts, baselines
5
Dependency & software awareness
“Where are my risks?”
SBOMs, patching
6
Operational & incident awareness
“Can we respond fast?”
SIEM, automated response
7
Human & process awareness
“Do we understand the big picture?”
Shared playbooks, training

The stream defender’s checklist

Don't get into the news

- Avoid misconfigurations
- Patch & update fast
- Verify your dependencies & supply chain integrity

Encryption - select your favorite animals

In transit *At rest* *Topic level* *Field-level* *Envelope* *End-to-end*

DDoS

- A DDoS in streaming isn’t about downtime - it’s about delay
- Quotas, private links, and client isolation for defense
- You can’t fight what you don’t see

Data exfiltration

- The real threat may already be authenticated
- Shrink what can be stolen
- Every read leaves a trail

Least privilege

- Go for mTLS or OAuth over older protocols
- Access control defines the blast radius
- Every job has its own identity.
- Authorization without auditing is blind

AI

- Verify -> trust
- Just in case -> Just in time
- Perimeter-based control -> throughout the system

Situational awareness

- Know your data in motion
- Watch for the unexpected in real time
- Build for visibility and rapid response

youtube.com/@OlenaKutsenko