

# Insider Threat (Database Intrusion Detection)



# Insider Threats: Motivation and Challenges

- Mission-critical information = High-value target
- Threatens Government organizations and large corporations
- Probability is “low”, but impact is **severe**
- Types of threat posed by malicious insiders
  - Denial of service
  - **Data leakage and compromise of confidentiality**
  - Compromise of integrity
- High complexity of problem
  - Increase in sharing of information, knowledge
  - Increased availability of corporate knowledge online
  - “Low and Slow” nature of malicious insiders

An “insider” is an individual who has currently or has previously had authorized access to information of an organization

# Some (old) Data

E-Crime Watch Survey 2004 (CERT and US Secret Service)  
<http://www.cert.org/archive/pdf/2004eCrimeWatchSummary.pdf>, 2004

- Insider threats identified as the 2<sup>nd</sup> biggest threat after hackers
- Majority of the incidents detected only AFTER the attack through MANUAL procedures
- 29% of attacks on survey respondents' organizations were from insiders
- Of these attacks, 34% involved “critical disruption” to the organization, its customers, or the larger critical infrastructure, which includes systems of government, telecommunications, finance, energy, etc.

# Some (new) Data

2010 CyberSecurity Watch Survey (\*) (CSO Magazine in cooperation with US Secret Service, CMU CERT and Deloitte)

- 26% of attacks on survey respondents' organizations were from insiders
  - (as comparison: 50% from outsiders, 24% unknown)
- Of these attacks, the most frequent types are:
  - Unauthorized access to/ use of information, systems or networks 23%
  - Theft of other (proprietary) info including customer records, financial records, etc. 15%
  - Theft of Intellectual Property 16%
  - *Unintentional exposure of private or sensitive information 29%*

(\*) [http://www.sei.cmu.edu/newsitems/cyber\\_sec\\_watch\\_2010\\_release.cfm](http://www.sei.cmu.edu/newsitems/cyber_sec_watch_2010_release.cfm)

INSIDER ATTACKS AND HUMAN ERROR: IS YOUR DATABASE SAFE?



**56%** of the non-financial services respondents believe that human error represents the greatest challenge or risk to database security.

channelinsider

INSIDER ATTACKS AND HUMAN ERROR: IS YOUR DATABASE SAFE?



**24%** of non-financial services firms state that abuse of privileges is the greatest threat.

channelinsider

INSIDER ATTACKS AND HUMAN ERROR: IS YOUR DATABASE SAFE?



**77%** of financial firms are mostly concerned with human error.

channelinsider

INSIDER ATTACKS AND HUMAN ERROR: IS YOUR DATABASE SAFE?

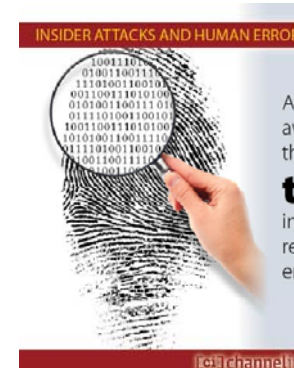


**48%** of database pros at financial firms are kept awake at night at the thought of insider privilege misuse.

channelinsider

## Insider Attacks and Human Error: Is Your Database Safe?

INSIDER ATTACKS AND HUMAN ERROR: IS YOUR DATABASE SAFE?



Among the respondents aware of a data breach that occurred recently, **two-thirds** indicate that it was a result of either human error or an insider attack.

channelinsider

INSIDER ATTACKS AND HUMAN ERROR: IS YOUR DATABASE SAFE?



In spite of these threats, only **38%** of organizations use role-based access control tools to manage access to databases.

channelinsider

INSIDER ATTACKS AND HUMAN ERROR: IS YOUR DATABASE SAFE?



And just **36%** have a database activity monitoring solution installed.

channelinsider

INSIDER ATTACKS AND HUMAN ERROR: IS YOUR DATABASE SAFE?



In fact, fewer than **half** of respondents even patch their database systems within a day of update availability.

channelinsider

# Intrusions vs Insider Attacks

Intrusions	Insider attacks
Outsider	Malicious insider
Masquerade	Masquerade/traitor (same as the malicious insider)
Unauthorized	Authorized
Outside organization's perimeter security	Inside or outside organization's perimeter security
May not be aware of the policies, procedures, and technology used organizations, and also often their vulnerabilities	Usually aware of the policies, procedures, and technology used organizations, and also often their vulnerabilities
Mostly technical	Technical or non-technical
Mostly anomalous	May or may not be anomalous

# Insider Attack Detection

- How are they currently detected by organizations?
  - Notification of a problem by a customer
  - Law enforcement officer, coworker, informant, auditor, or other external person who became suspicious
  - Sudden appearance of a competing business
  - Unable to perform daily operations
  - Accidental discovery during system/configuration upgrades
- How the insider identified after detection?
  - Mostly through various logs
- Can organizations do better?



# Remediation: Some Initial Ideas

- Distribute trust amongst multiple parties to force collusion
  - Most insiders act alone
- Question trust assumptions made in computing systems
  - Treat the LAN like the WAN
- Log all actions
- Isolate DBA from user data – *Oracle Database Vault*
- Create profiles of data access and monitor data accesses to detect anomalies



# Relevant Requirements

Regulatory Legislation	Regulation Requirement
Sarbanes-Oxley Section 302	Unauthorized changes to data
Sarbanes-Oxley Section 404	Modification to data, Unauthorized access
Sarbanes-Oxley Section 409	Denial of service, Unauthorized access
Gramm-Leach-Bliley	Unauthorized access, modification and/or disclosure
HIPAA 164.306	Unauthorized access to data
HIPAA 164.312	Unauthorized access to data
Basel II – Internal Risk Management	Unauthorized access to data
CFR Part 11	Unauthorized access to data
Japan Privacy Law	Unauthorized access to data
PCI – Requirement 7	Restrict access to cardholder data by business need-to-know
PCI – Requirement 8.5.6	Enable accounts used by vendors for remote maintenance only during the time period needed
PCI – Compensating Controls for Requirement 3.4	Provide ability to restrict access to cardholder data or databases based on the following criteria: <ul style="list-style-type: none"> <li>• IP address/Mac address</li> <li>• Application/service</li> <li>• User accounts/groups</li> </ul>
PCI - Requirement A.1: Hosting providers protect cardholder data environment	Ensure that each entity only has access to own cardholder data environment

# Database Intrusion Detection

- ID mechanisms have been extensively studied in OS and networks
- Why is it important to have an ID mechanism tailored for a DBMS?
  - Actions deemed malicious for a DBMS are not necessarily malicious for the underlying operating system or the network
    - A database user/application normally access data only from the human resources schema but submits a SQL command to the DBMS that accesses the financial records of the employees from the finance schema.
    - Such anomalous access pattern of the SQL command may be the result of a SQL Injection vulnerability or privilege abuse by an authorized user.
  - The key observation is that an ID system designed for a network or an operating system is ineffective against such database specific malicious actions

A. Kamra, E. Terzi, E. Bertino: Detecting anomalous access patterns in relational databases. VLDB J.  
B. 17(5): 1063-1077 (2008)

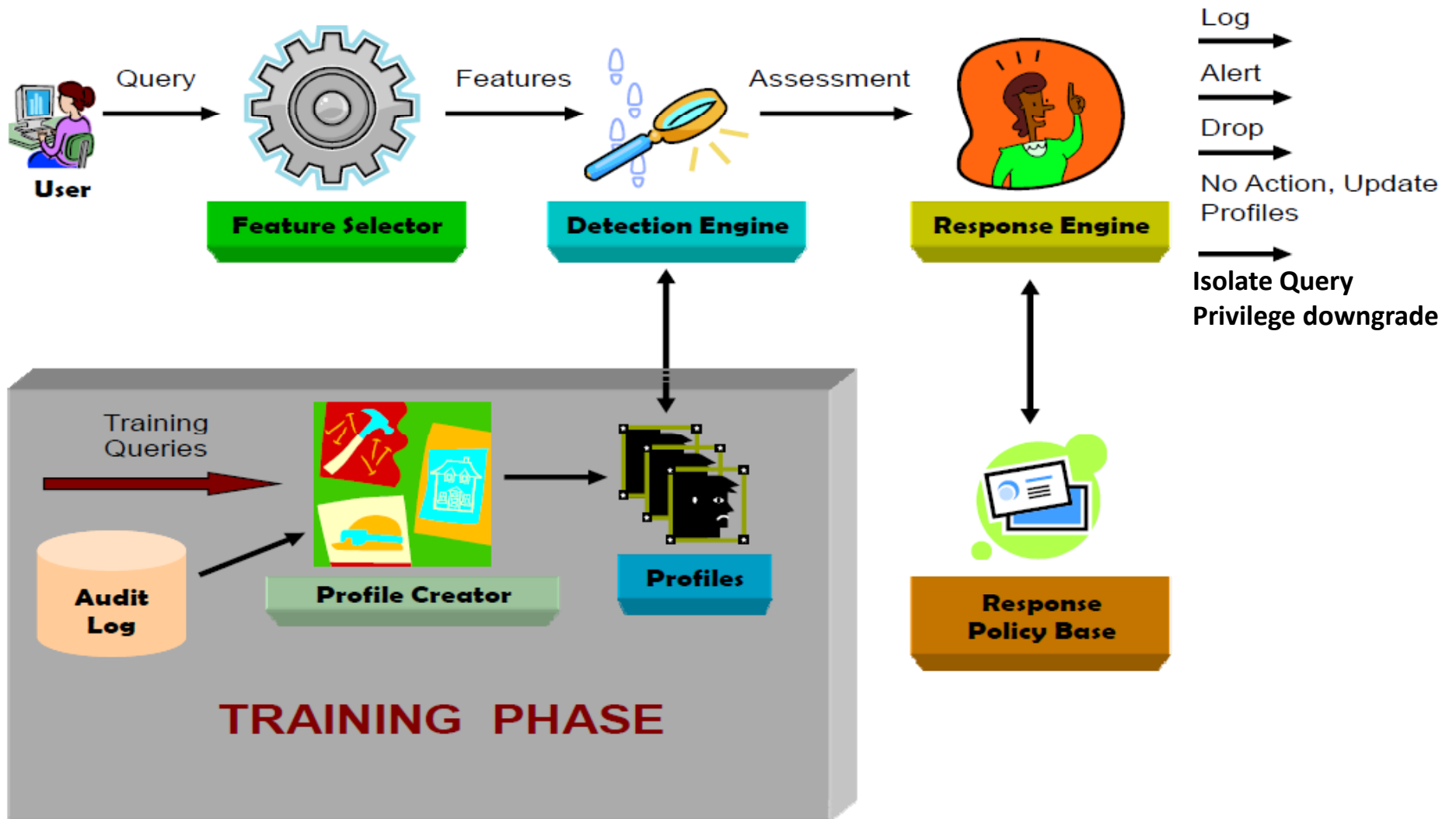
# Integrating ID and DBMS

- The intrusion detection is done as close to the target as possible (during query processing) thereby ruling out any chances of a backdoor entry to the DBMS that may bypass the ID mechanism.
- The physical location of the DBMS is not a constraint on obtaining the ID service.
  - Such requirement is critical in the current age of cloud computing if the organizations want to move their databases to a cloud service provider.
- Allows the ID mechanism to issue more versatile response actions to an anomalous request.

# Basic Framework

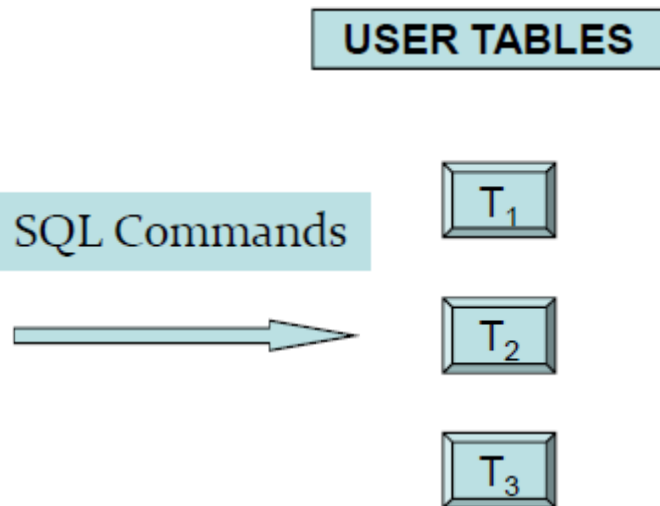
- Observation
  - A masquerader has stolen someone's credentials
  - He accesses what the victim is authorized to use
  - Unlikely to perform actions consistent with victim's typical behavior
  - Behavior is not something that can be easily stolen
- Framework
  - Extract patterns that are "normal"
  - Build profiles of these patterns
  - Build classifier or clusters
  - At runtime, if a pattern deviates from the classes/clusters, then it is **potentially** anomalous
- NOTE: "anomalous" does not necessarily mean "malicious"

# System Architecture

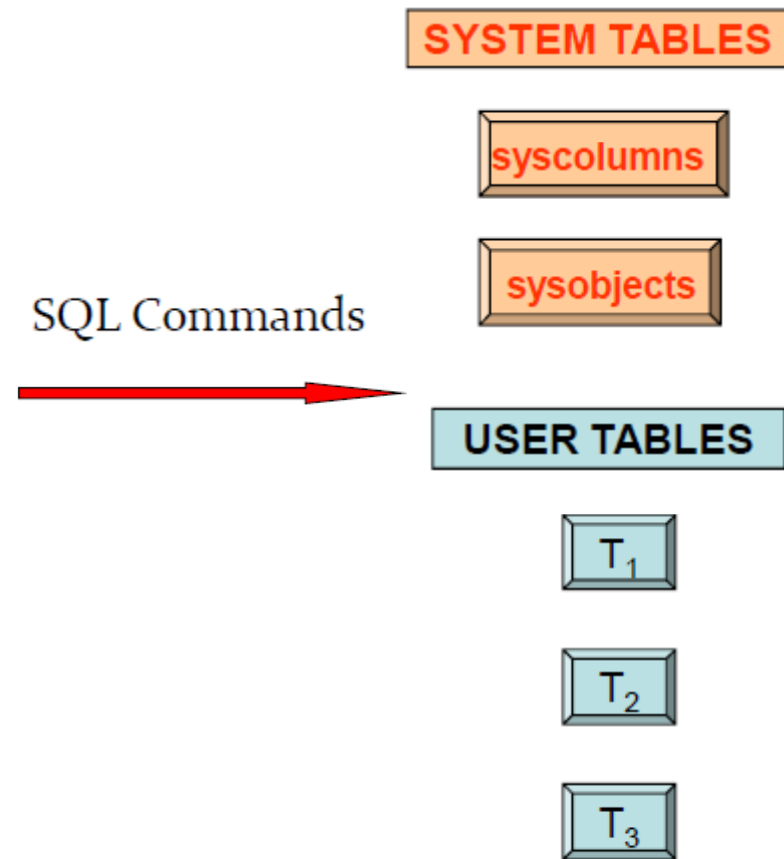


# Anomalous Access Patterns

## Normal Access Pattern



## Anomalous Access Pattern



# SQL Query Representation

- There is an assumption that users interact with the database through commands, where each command is **a different entry in the log file.**

SELECT [DISTINCT]	{TARGET-LIST}
FROM	{RELATION-LIST}
WHERE	{QUALIFICATION}

- Each log entry transforms to specific format that can be analyzed later. This format contains **5 fields** and thus called a ***quiplet***.
- Thus the log file can be viewed as **a list of quiplets.**
- Quiplets are basic units for forming profiles.
- **NOTE:** Quiplets are based on **syntax** only (not **semantics**)



# Data Representation

- Each quiplet is of the form  $Q(c; P_R; P_A; S_R; S_A)$ 
  1. SQL Command
  2. **Projection** Relation Information
  3. Projection Attribute Information
  4. **Selection** Relation Information
  5. Selection Attribute Information
- Can be represented by **one of three** different representation levels (each level is characterized by a different amount of recorded information).

# Coarse Quiplet

- Schema:

T1:{a1,b1,c1} T2:{a2,b2,c2} T3:{a3,b3,c3}

- Query:

~~SELECT T1.a1, T1.c1, T2.c2 FROM T1,T2,T3~~  
~~WHERE T1.a1=T2.a2 AND T1.a1=T3.a3~~

c-quiplet is  
sufficient in the  
case of a small  
number of well-  
separated roles

Field	Value
Command	SELECT
Num Projection Tables	2
Num Projection Columns	3
Num Selection Tables	3
Num Selection Columns	3

# Medium Quiplet

- Schema:

T1:{a1,b1,c1} T2:{a2,b2,c2} T3:{a3,b3,c3}

- Query:

SELECT T1.a1, T1.c1, T2.c2 FROM T1,T2,T3  
WHERE T1.a1=T2.a2 AND T1.a1=T3.a3

No attribute from T3  
being projected

Field	Value
Command	SELECT
Projection Tables	[1 1 0]
Projection Columns	[2 1 0]
Selection Tables	[1 1 1]
Selection Columns	[1 1 1]

# Medium Quiplet

- Schema:

T1:{a1,b1,c1} T2:{a2,b2,c2} T3:{a3,b3,c3}

- Query:

SELECT T1.a1, T1.c1, T2.c2 FROM T1,T2,T3  
WHERE T1.a1=T2.a2 AND T1.a1=T3.a3

No attribute from T3  
being projected

Field	Value
Command	SELECT
Projection Tables	[1 1 0]
Projection Columns	[2 1 0]
Selection Tables	[1 1 1]
Selection Columns	[1 1 1]

# Fine Quiplet

- Schema:

T1:{a1,b1,c1} T2:{a2,b2,c2} T3:{a3,b3,c3}

- Query:

**SELECT** T1.a1, T1.c1, T2.c2 **FROM** T1,T2,T3  
**WHERE** T1.a1=T2.a2 AND T1.a1=T3.a3

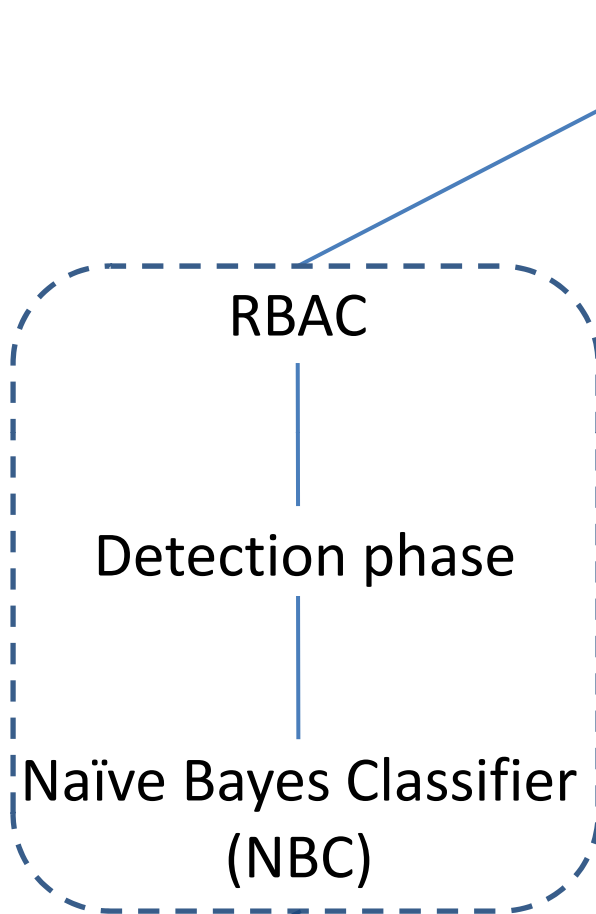
a1 is a projected column  
b1 is not  
c1 is

Field	Value
Command	SELECT
Projection Tables	[1 1 0]
Projection Columns	[[1 0 1] [0 0 1] [0 0 0]]
Selection Tables	[1 1 1]
Selection Columns	[[1 0 0] [1 0 0] [1 0 0]]

# Scenarios

- Two possible scenarios
  - Role-based
    - Queries are associated with roles
    - Supervised learning/data mining
    - Build a profile for each role
    - Build classifier to detect anomalous queries
  - Individual-based
    - Queries are associated with each user
    - Unsupervised learning/data mining
    - A lot more users than roles!
    - Cluster users into groups of similar behaviors to form concise profiles
    - Anomalous queries correspond to outliers

# Methodology



Classification problem



# Role-based Anomaly Detection

- Associate each query (from the audit files) with a role
- Build profiles per role
- Train a classifier with role as the class
  - Use Naïve Bayes Classifier
    - Low computational complexity
    - Ease of implementation
    - Works surprisingly well in practice even if the attributes independence condition is not met
- At runtime, declare a request as anomalous if classifier predicted role does not match the actual role

# NBC-based ID

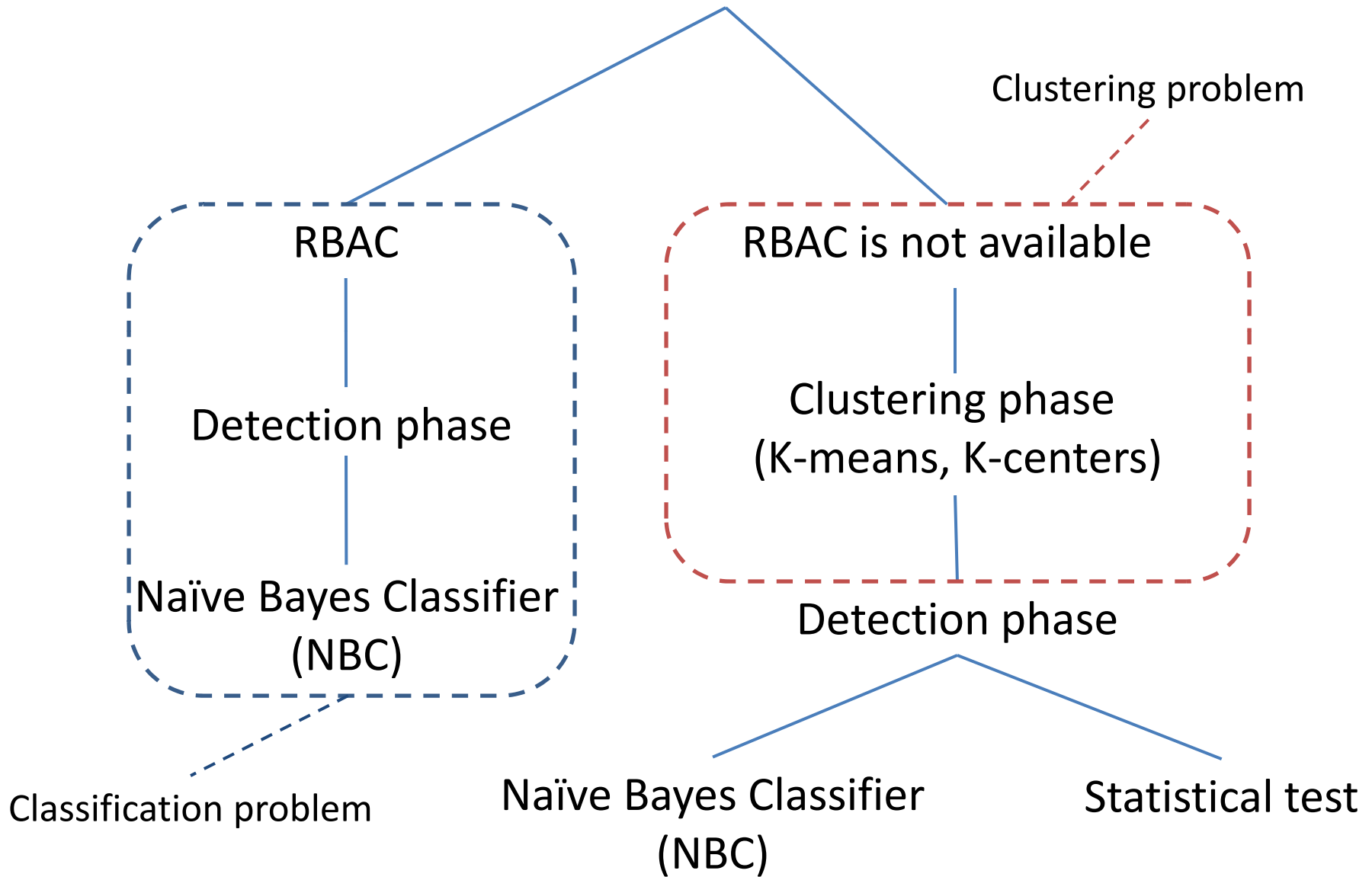
- Query traces submitted by live applications
- Results for the supervised case

Quiplet type	False negative (%)	False positive (%)
Coarse	2.6	19.2
Medium	2.4	17.1
Fine	2.4	17.9

## 8 roles. Real Dataset:

1. 8368 SQL Traces
2. 130 Tables
3. 1201 Columns
4. 7583 Select, 213 Insert and 572 Update Commands

# Methodology



# Un-supervised Anomaly Detection

- The role information is not available in the audit log files
  - Associate every query with a user
- Use clustering algorithms to partition training data into clusters
- The method maintains a mapping for every user to its representative cluster (the cluster that contains the maximum number of training records for that user after the clustering phase)

# Anomaly Detection

1. Find the representative cluster ( $C_z$ ) for query  $z$  issued by user  $U$ 
  - Look up the user-cluster mapping created during the clustering phase.
2. Two different approaches for the detection phase:
  - To use the naive Bayes classifier and treat the clusters as the classifier classes.
  - Determine if  $z$  is an outlier in cluster  $C_z$  with the help of a statistical test (in principle any test can be used)

# Limitations

- Two *similar looking* queries can produce completely *different* results

```
SELECT p.product_name, p.product_id  
FROM PRODUCT p  
WHERE p.cost = 100 and p.weight > 80
```

vs

```
SELECT p.product_name, p.product_id  
FROM PRODUCT p  
WHERE p.cost > 100 and p.weight = 80
```

- False negative – similar syntax, different results
- Two *different looking* queries can produce *similar* results

```
SELECT p.product_name, p.product_id  
FROM PRODUCT p  
WHERE p.cost = 100 and p.weight > 80
```

vs

```
SELECT p.product_name, p.product_id  
FROM PRODUCT p  
WHERE p.cost = 100 and p.weight > 80  
AND p.product_name is not null;
```

- False positive – different syntax, same results

# Database Intrusion Response

- Once intrusion has been detected, how should a database response?
- Three main types of responses
  - Conservative actions, e.g., sending an alert and allow the anomalous requests to go through
  - Aggressive actions, e.g., block the anomalous requests
  - Fine-grained response actions
    - Neither conservation or aggressive
    - **Suspend** an anomalous request – put it on hold until users perform specific actions, e.g., further authentication steps
    - **Taint** the request – marked it as a potential suspicious request resulting in further monitoring or possibly suspension or dropping of subsequent requests by the same user



# Database Response Policy

- Which response measure to take?
  - Non-trivial to develop an automated response mechanism
  - E.g., a user who submits a query that is not captured in his/her profile
  - Given that query is “anomalous”, if it accesses sensitive data, strong response action may be needed (e.g., revoke user’s privilege)
    - What if it’s a false alarm? E.g., one time action required to accomplish some task
  - Need to look at details of the requests and the context surrounding it (e.g., time of the day, origin of requests, etc)
- A **response policy** is required by database security administrator to specify appropriate response actions for different circumstances

# Policy Language

- We can view the detection of an anomaly as an event, and the attributes of the anomaly (user, role, SQL command) as the environment surrounding the event
- A policy can be specified taking into account the attributes to guide the response engine to take a **suitable** action
- An **ECA policy** can be specified as a **ECA rule** that drives the response action:

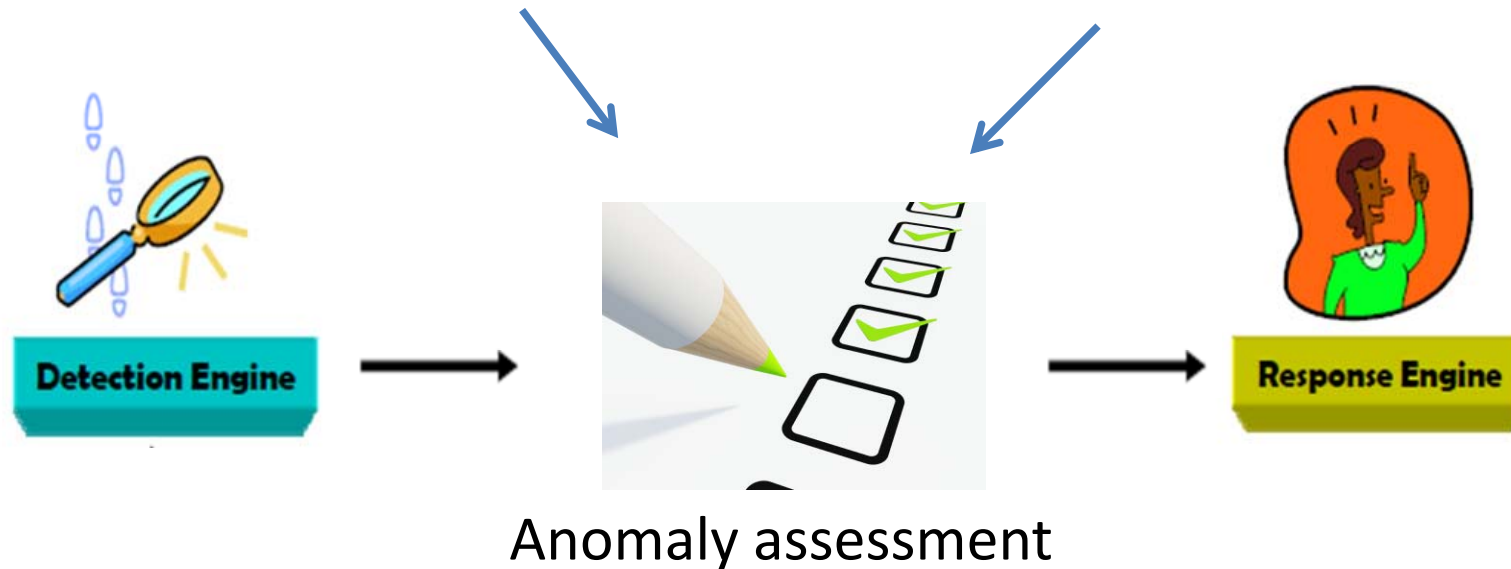
ON {Event} IF {Condition} THEN {Action}

# ECA Policy

Conditions specified on  
the anomaly attributes

CONTEXTUAL	
User	The user associated with the request.
Role	The role associated with the request.
Client App	The client application associated with the request.
Source IP	The IP address associated with the request.
Date Time	Date/Time of the anomalous request.

STRUCTURAL	
Database	The database referred to in the request.
Schema	The schema referred to in the request.
Obj Type	The object types referred to in the request such as table, view, stored procedure
Obj(s)	The object name(s) referred in the request
SQLCmd	The SQL Command associated with the request
Obj Attr(s)	The attributes of the object(s) referred in the request.



# Policy Conditions

## Anomaly Attributes

$A_1 = \text{Source IP}$ ,  $A_2 = \text{SQLCmd}$ ,  $A_3 = \text{User}$ ,  $A_4 = \text{Role}$

## Policy Predicates

$Pr_1$ : Source IP IN 192.168.0.0/16

$Pr_2$ : Source IP IN 128.10.0.0/16

$Pr_3$ : SQLCmd IN {Insert, Delete, Update}

$Pr_4$ : SQLCmd = 'exec'

$Pr_5$ : Role != 'DBA'

$Pr_6$ : User = 'appuser'

## Policy Conditions

$Pol_1(C) = Pr_1 \wedge Pr_3$

$Pol_2(C) = Pr_2 \wedge Pr_6$

$Pol_3(C) = Pr_4 \wedge Pr_5$

$Pol_4(C) = Pr_1 \wedge Pr_3 \wedge Pr_6$

# Response Actions

<b>CONSERVATIVE: low severity</b>	
NOP	No OPeration. This option can be used to filter unwanted alarms.
LOG	The anomaly details are logged.
ALERT	A notification is sent.
<b>FINE-GRAINED: medium severity</b>	
TAINT	The request is audited.
SUSPEND	The request is put on hold till execution of a confirmation action.
<b>AGGRESSIVE: high severity</b>	
ABORT	The anomalous request is aborted.
DISCONNECT	The user session is disconnected.
REVOKE	A subset of user-privileges are revoked.
DENY	A subset of user-privileges are denied.

- Response can also be a sequence of actions, e.g., LOG followed by ALERT

# Response Policy: Example 1

- If there is an anomalous access to tables in the 'dbo' schema (system catalogue) from un-privileged users inside the organization's internal network, the user should be disconnected

*ON ANOMALY DETECTION*

*IF Role != DBA and*

*SourceIP IN 192.168.0.0/16 and*

*ObjType = table and*

*Objs IN dbo.\* and*

*SQLCmd IN {Select, Update, Insert}*

*THEN DISCONNECT*

## Response Policy: Example 2

- If there is an anomalous access originating from a privileged user during usual business hours, then take no action if it originates from the internal network of the organization (this policy prevents false alarms)

```
ON ANOMALY DETECTION
IF Role = DBA and
   SourceIP IN 192.168.0.0/16 and
   ObjType = table and
   DateTime BETWEEN 0800-1700
THEN NOP
```



# Interactive ECA Response Policies

- Sometimes, upon anomaly detection, the system may want to engage in interactions with the users,
  - SUSPEND anomalous requests
  - Request user to authenticate with a second authentication factor as the next section
  - Upon authentication failure, DISCONNECT user; otherwise, resume normal processing
- Interactive ECA response policy:

ON	{Event}
IF	{Condition}
THEN	{Initial Action}
CONFIRM	{Confirmation Action}
ON SUCCESS	{Resolution Action}
ON FAILURE	{Failure Action}

**CONFIRM** - Second course of action after the initial response action – interact with user to resolve effects of Initial action

# Interactive Response Policy: Example

- Re-authenticate un-privileged users who are logged from inside the organization's internal network for write anomalies to tables in the dbo schema. If re-authentication fails, drop the request and disconnect the user else do nothing

```
ON ANOMALY DETECTION
IF Role != DBA and
  SourceIP IN 192.168.0.0/16 and
  ObjType = table and
  Objs IN dbo.* and
  SQLCmd IN {Select, Update, Insert}
THEN SUSPEND
CONFIRM re-authenticate
ON SUCCESS NOP
ON FAILURE ABORT, DISCONNECT
```

# Summary

- It is challenging to deal with insider threat.
- Intrusion detection mechanisms can be used
  - Determine the profiles of users/roles, and then check for anomaly during querying
  - Besides syntax, semantics of queries need to be considered to minimize false negatives/positives
- Response also has to be carefully managed, and accessed.
- Fine-grained response is important, and integrating with access control offers greater flexibility