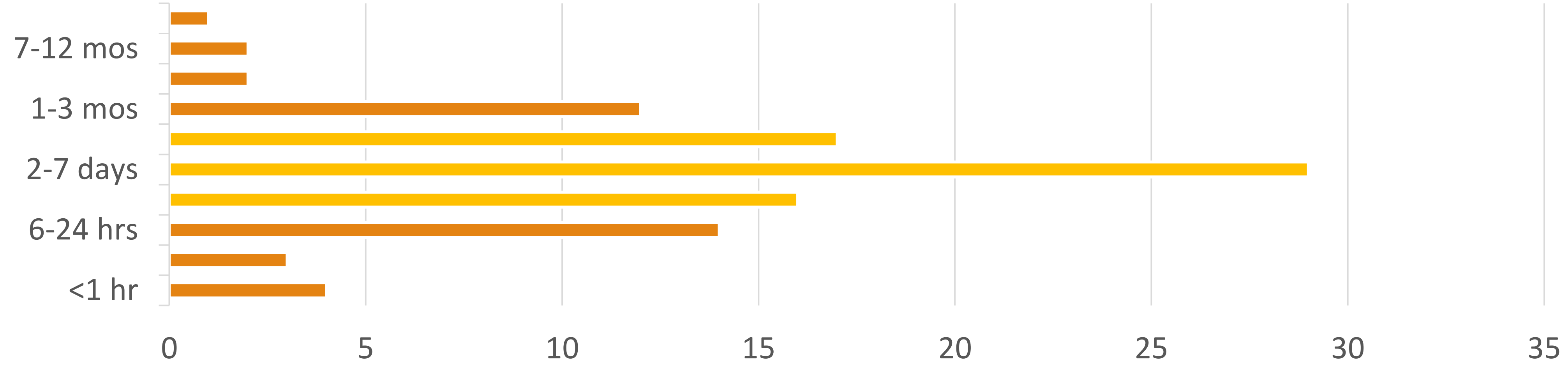
Prioritization of Cloud System Monitoring for Incident Response

UTTAM THAKORE

MARCH 2, 2017

Problem

Average time between detection & response to incidents in enterprises is staggering:



Rapid identification of and response to incidents is necessary to ensuring reliability and security of large-scale enterprise cloud systems

- Requires *efficient analysis* of heterogeneous monitor and log data

Source: *Incident Response Capabilities in 2016: The SANS 2016 Incident Response Survey*, SANS Institute, June 2016

Current state of the art

Decisions on **what to monitor** and **where to focus data analysis** are driven by domain expertise and default settings of logging and monitoring tools

Issues are discovered by clients or administrators observing undesirable functionality or critical alerts, and managed through ticketing at a helpdesk

Log exploration is performed manually, sometimes aided by dashboards and incident tracking tools

Example log exploration process

1. Client or administrator identifies issue and reports a ticket.
2. Incident response team (IRT) member looks at ticket and must determine which data sources to look at first.
3. IRT member constructs log trail to identify root cause of issue. This can require:
   a) Googling error codes
   b) Ad hoc examination of related data sources
   c) Manual analysis of historical incident records to identify related past incidents
   d) Calling on collective prior experience of other administrators
4. Once the root cause is identified, IRT member reports root cause to system admins/developers to resolve the issue.

Challenges

Each ticket takes significant time to investigate

- According to SANS 2016 Incident Response survey[1], the majority of security incidents took over 24 hours to detect and over 24 hours to investigate and respond to

Root causes can be very far from where visible alerts are generated

Collected data may be:

- Insufficient to support investigation
- Excessive and not useful

[1] *Incident Response Capabilities in 2016: The SANS 2016 Incident Response Survey*, SANS Institute, June 2016

Our contribution

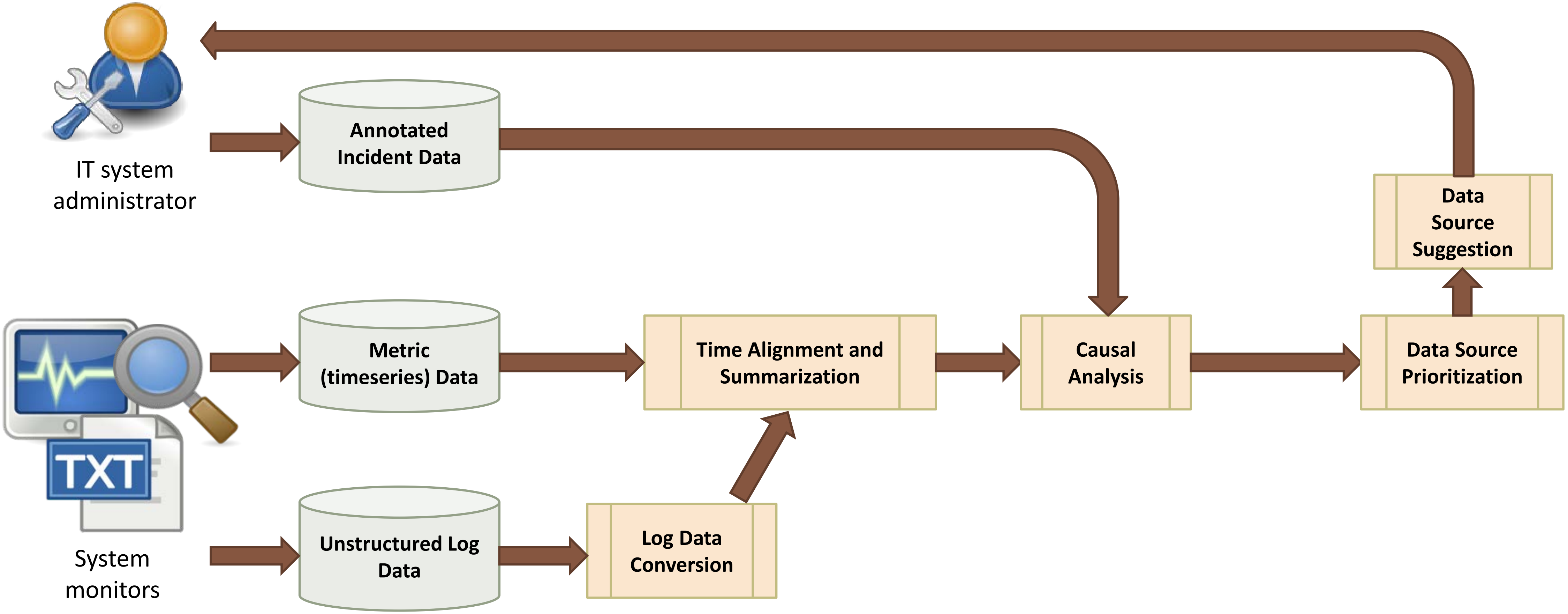
Prioritize data sources for collection and analysis based on their utility in detecting events

Use *statistical correlation analysis* to:

- Identify relationships between data sources at the time of an incident
- Quantify utility of data sources for incident detection

Approach

Overview


IT system administrator
Annotated Incident Data
Data Source Suggestion
Metric (timeseries) Data
Time Alignment and Summarization
Causal Analysis
Data Source Prioritization
TXT
System monitors
Unstructured Log Data
Log Data Conversion

Incident data analysis

Incident data: human-written incident reports, log and metric alerts, etc.

From these, we identify the data points that signal the incident

- Data sources containing these points are *terminal data sources (TDS)*
- Example: When a web application crashes, TDS may be application

Log data conversion

Log data must be converted from unstructured/semi-structured format to numerical time series

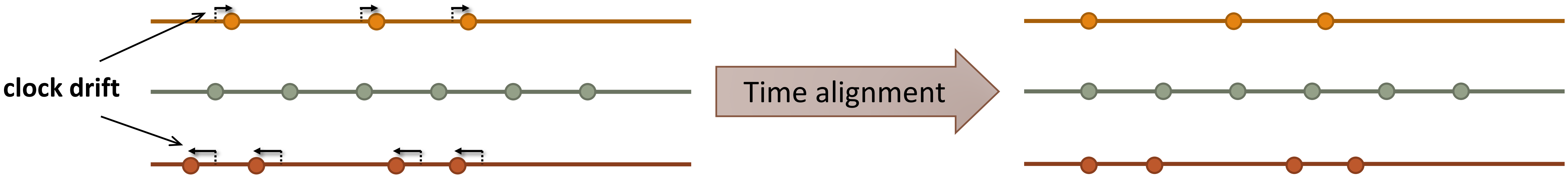
```
12.23.34.45 - - [04/Jan/2015:05:22:03 +0000] "GET
  /images/web/img1 HTTP/1.1" 200 1234 "-"
  "Mozilla/5.0 (compatible; MSIE 10.0; Windows NT
  6.1; WOW64; Trident/6.0)”
12.23.34.45 - - [04/Jan/2015:05:22:04 +0000] "GET
  /favicon.ico HTTP/1.1" 200 31415 "-" "Mozilla/5.0
  (compatible; MSIE 10.0; Windows NT 6.1; Win64; x64;
  Trident/6.0)”
23.34.45.56 - - [04/Jan/2015:05:22:13 +0000] "GET
  /doc/index.html?org/elasticsearch/action/search/Sea
  rchResponse.html HTTP/1.1" 404 294 "-" "Mozilla/5.0
  (compatible; Googlebot/2.1;
  +http://www.google.com/bot.html)”
```

Feature	Timestamped Values
HTTP Response Sizes	22:03: 1234 22:04: 31415 22:13: 294
HTTP Error Responses	22:03: 0 22:04: 0 22:13: 1
HTTP GET requests	22:03: 1 22:04: 1 22:13: 1

Time alignment and feature reduction

Time series may have clock drift or timestamp delays across processes and machines

- Can be on the order of tens of seconds per day!



May also need to reduce number of features to remove redundant/low relevance features

- Necessary to improve scalability and of analysis
- Currently evaluating different information criteria (e.g., types of entropy)

Statistical correlation analysis

We want to identify *important features*

- Features are unidimensional time series of data (metric data, processed log data)
- Important features help detect incident or anomalous behavior

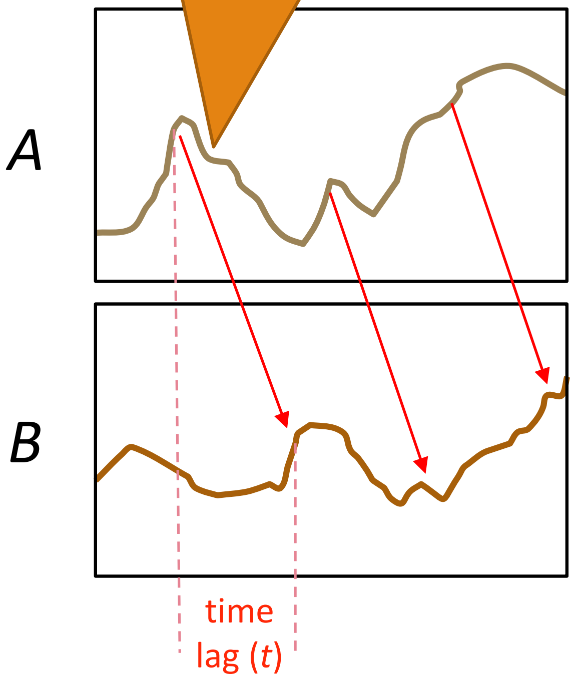
Our current approach: Multivariate Granger causality analysis

- Look at pairs of features (X, Y) in time window $T = \{t_1, t_2, \ldots, t_n\}$
- Determine if regression using time-lagged values of X and Y taken together correlates more strongly to current behavior of Y than regression with just past behavior of Y

Formally: $X \to_G Y \;\text{ iff }\; P\big[Y(t_{n+1}) \mid \mathcal{I}_{X,Y}(T)\big] > P\big[Y(t_{n+1}) \mid \mathcal{I}_Y(T)\big]$

Intuition: If particular behavior of Y is known to correspond to an undesirable incident, then collection of all such X would be desirable

Strong correlation from *A* → *B* with time lag *t*



[2] Spirtes, Peter, Clark N. Glymour, and Richard Scheines. "*Causation, prediction, and search.*" MIT press, 2000.

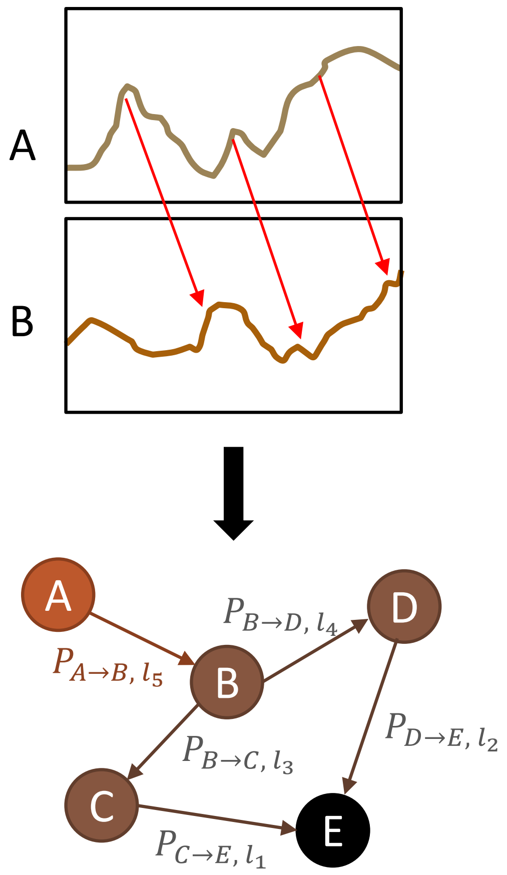
Statistical correlation analysis

Anchor at terminal data source (TDS) and perform correlation analysis across all other data sources

- Fixed max time lag and analysis window

Repeat process backwards from TDS to form correlation relationship graph

- Nodes: data sources
- Edges: time-lagged correlation relationships, labeled with **strength of correlation** ($P_{X \to Y}$) and **time lag** (l)
  - Can prune edges with weak correlation



Data source prioritization

We define a *prioritization score* (*PS*) to define priority for each data source

- For example, weighting data sources with highest correlation to incident and smallest time lag

Result: ranking of all data sources by prioritization score

How can this be used by an admin?

Incident analysis (post hoc): When incident is detected, run our approach and investigate data sources in decreasing order of *PS*

Monitor selection (pre hoc): Given list of incidents, determine which data sources to collect

- E.g., union of top *k* data sources for each of the incidents

Evaluation

We are currently evaluating the work on an IBM cloud dataset

- Contains multiple performance and reliability incidents that occurred in pre-production systems
- Detailed descriptions of incident root causes and evidence in data sources are provided
- Diverse data sources:
  - Web server logs
  - OS performance metrics
  - Application performance metrics and error logs

This work is also generalizable to performance-related security incidents, like flooding attacks (e.g., denial of service)

Ongoing and future work

Current evaluation indicates that better feature selectionand graph pruning are needed to improve performance

- Redundant and low information features can cloud the correlation relationship graphs obtained

A paper submission based on the work is planned for Spring 2017