Probabilistic Graph Models: from Bayesian to Factor Graphs

ECE/CS 498 DS U/G

Lecture 16: Conceptual Discussion of PGMs

Ravi K. Iyer

Dept. of Electrical and Computer Engineering

University of Illinois at Urbana Champaign

Announcements – Movement to Online Course

- Course will now be **entirely online** - no more face-to-face meetings
- Lectures:
  - Will be conducted online via Zoom
  - Will be more discussion-based
  - Attendance will be tracked
  - Lecture recordings will still be uploaded to our Media Space channel
  - Please follow Zoom etiquette: mute microphones when not speaking and turn off videos
- Quizzes:
  - To better gauge course understanding, quizzes will now occur **once a week**
  - As before, they will be short, conceptual, and occur during the last 5-10 minutes of class on Compass2G

Announcements – Movement to Online Course

- Discussion Sections/TA Office Hours:
  - Will be online via Zoom
  - Will occur at the same scheduled times
- ICAs:
  - Will occur during regularly scheduled times
  - Will be conducted via Zoom "Breakout Rooms"
  - Students may still work in groups, but everyone needs to individually submit their group's work in a PDF file on Compass2G
- MPs/HWs: No changes
- Final Project: Generally no changes, except that final presentations will need to be done remotely via Zoom
- Refer to Piazza for more detailed information, including how to access Zoom: https://piazza.com/class/k5js5bkktry6cu?cid=194

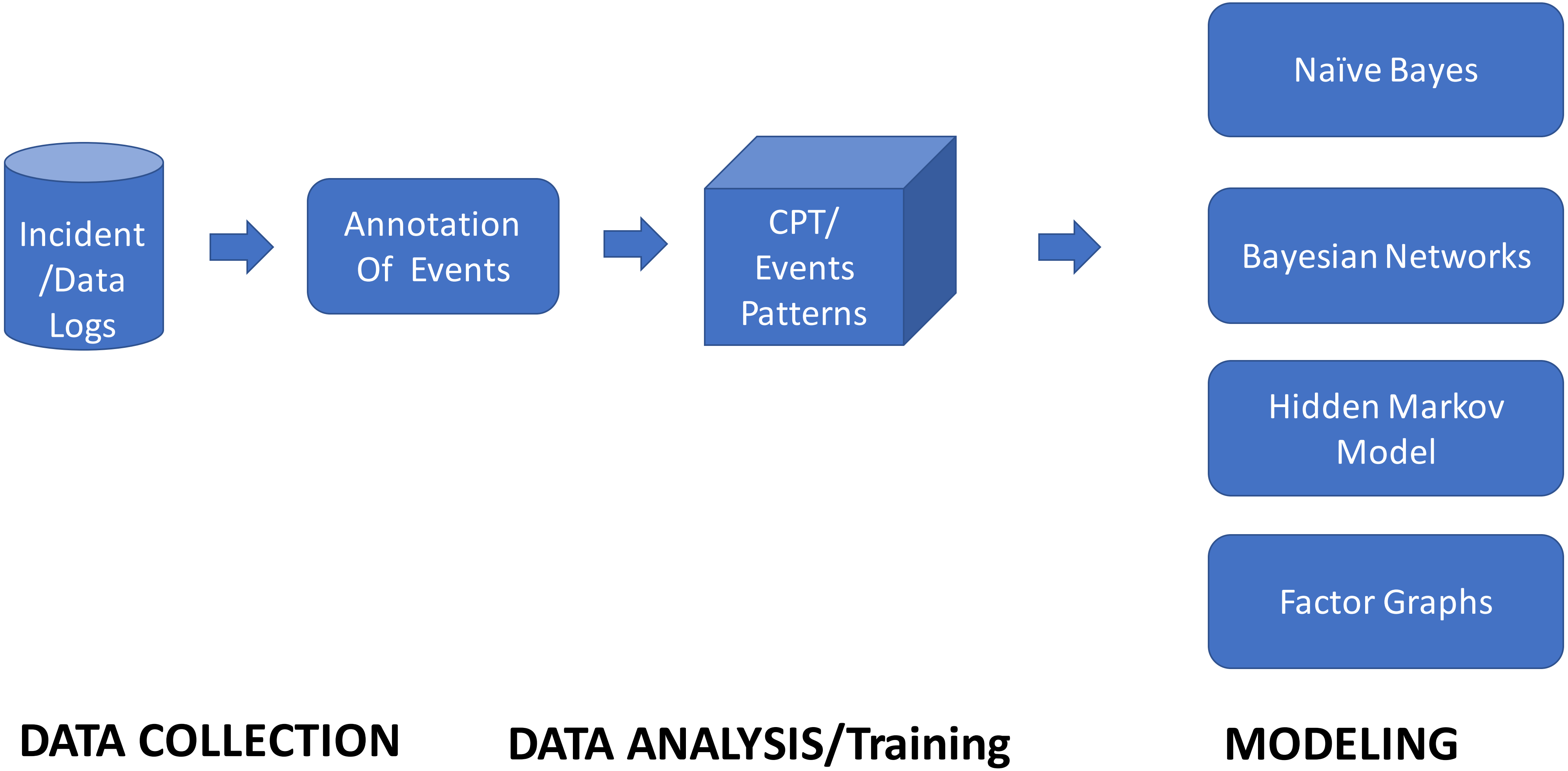
Announcements

- Course Timeline
  - Today 3/23: Conceptual Discussion of PGMs
  - Wed 3/25: Introduction to Hidden Markov Models (HMMs)
  - Mon 3/30: HMMs Continued, **ICA 4**
- MP 2 Timeline
  - Checkpoint 1.5 due on **Wednesday March 25 @ 11:59 PM**
    - Submit via https://forms.gle/88Wk6QtxvaWsFChX6
  - Final Checkpoint due on **Monday March 30 @ 11:59 PM on Compass2G**
- Final Project
  - Make sure to review feedback from proposals
  - Progress report 1 due **Friday March 27 @ 11:59 PM** on Compass2G
    - We are expecting reasonable progress from the time of the project proposals…

Outline

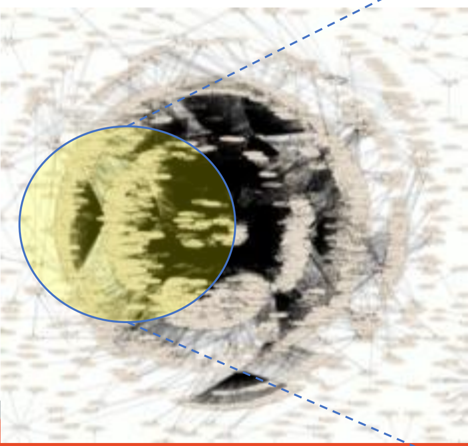
- Conceptually understanding of Probabilistic Graph Models (PGMs) -- from Bayesian, Hidden Markov Models and Factor Graphs
- A case study
  - A credential-stealing attack
- Probabilistic modeling of attacks using Naïve Bayes, Bayesian Networks, and Factor Graphs

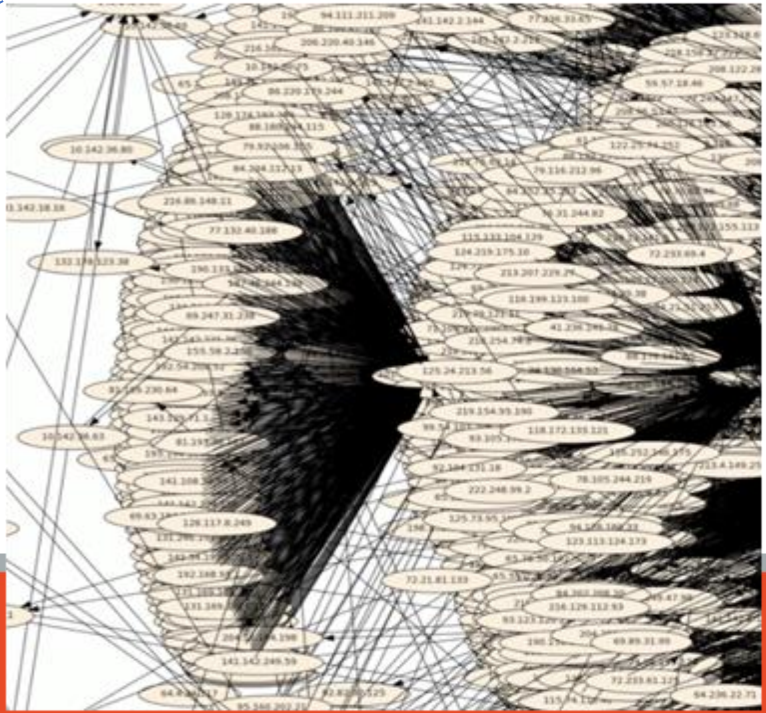
Overview of PGM Data Analytics/Modeling Process



Measurements from NCSA@Illinois: Five minute Snap Shot

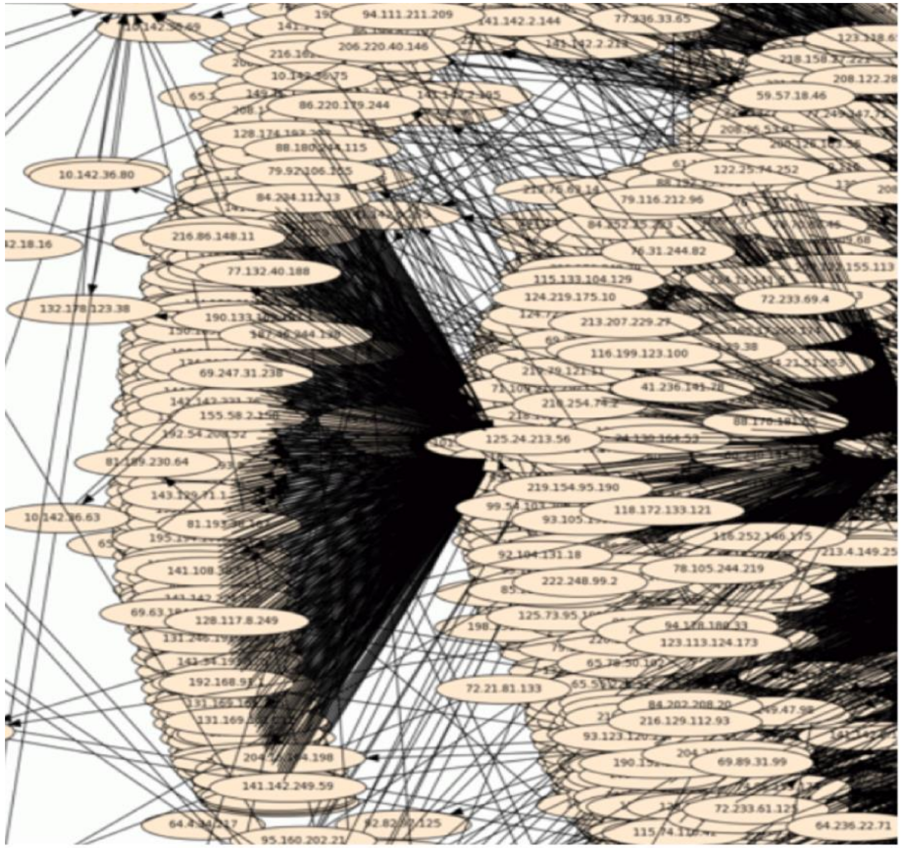
- Goals:
  - Provide a system-level characterization of incidents and evaluate the intricacies of real-time diagnosis
  - Design protection strategies to reduce missed incidents and false positives
  - Experimentally Demonstrate new techniques in a sandbox
- Challenges



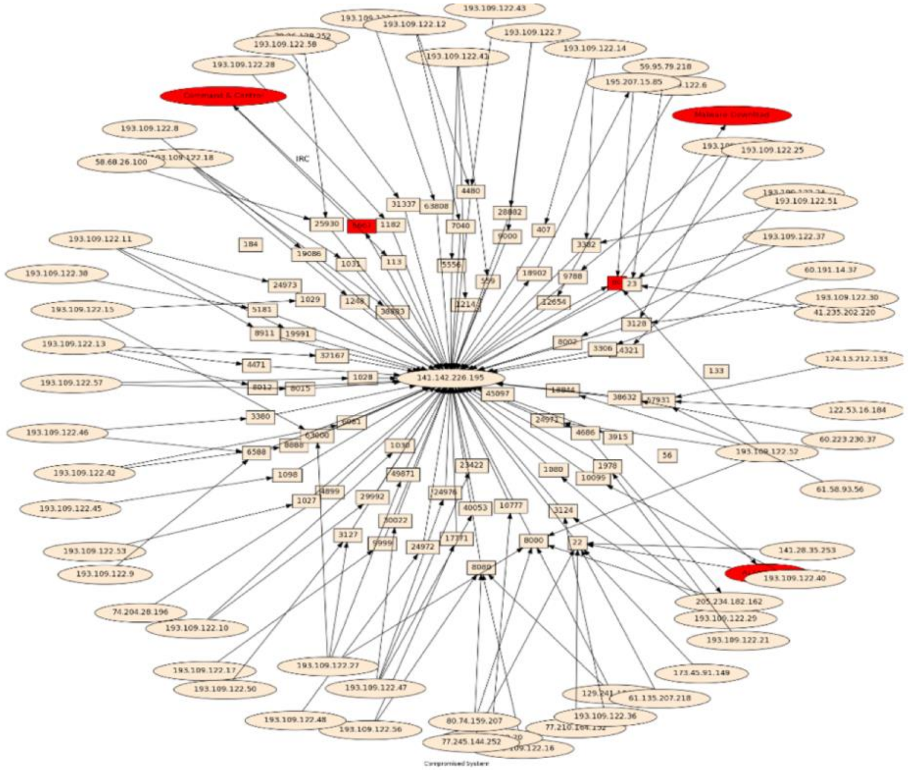


Five-Minute Snapshot of In-and-Out Traffic at NCSA

Five-Minute Snapshot of In-and-Out Traffic within NCSA@Illinois

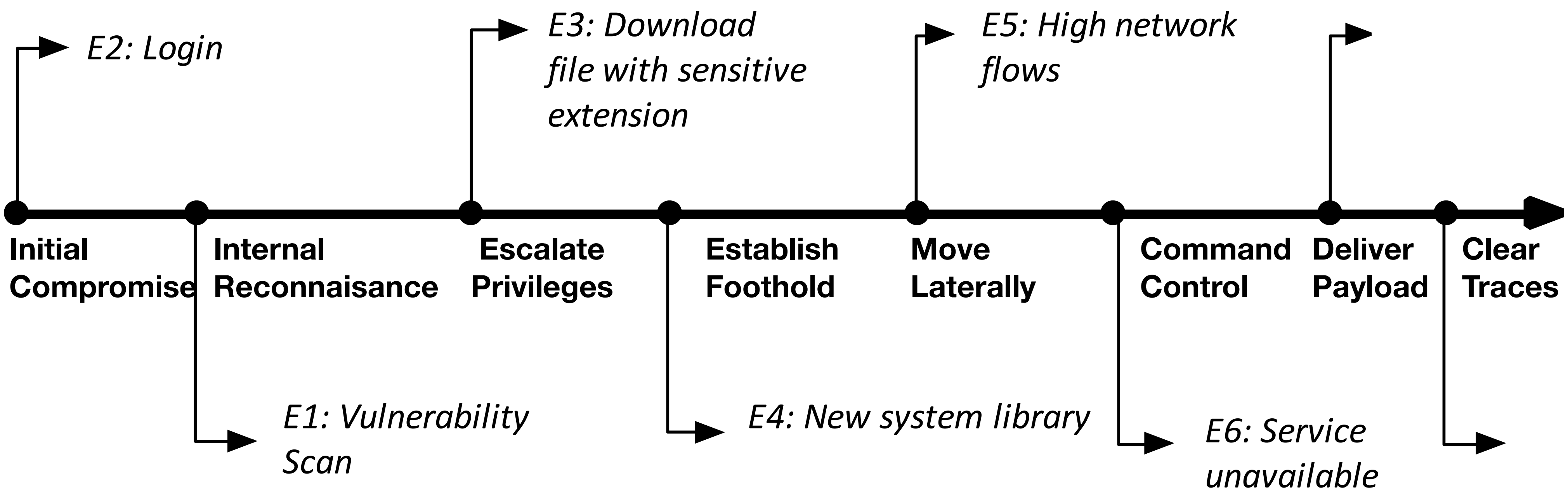


(a)



(b)

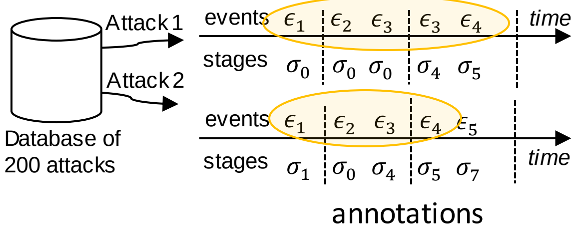
An Application in Security Data Analytics
Individual components of an attack as attack progresses



Attack stages for the credential stealing attack

Annotation and extracting patterns in past attacks

a) Annotated events and attack stages in a pair of attacks



b) Event-stage annotation table for the attack pair (Attack 1 and Attack 2)

Event	**Attack stage**
$\{\epsilon_1\}$	$\{\sigma_0\|\sigma_1\}$
$\{\epsilon_2\}$	$\{\sigma_0\}$
$\{\epsilon_3\}$	$\{\sigma_4\}$
$\{\epsilon_4\}$	$\{\sigma_5\}$
$\{\epsilon_5\}$	$\{\sigma_7\}$

c) Example patterns, stages, probabilities, and significance learned from the attack pair

Pattern	Attack stages	Probability in past attacks	Significance (p-value)
$[\epsilon_1, \epsilon_3, \epsilon_4]$	$[\sigma_1, \sigma_4, \sigma_5]$	q_a	p_a
$[\epsilon_1]$	$[\sigma_0\|\sigma_1]$	q_b	p_b

…

Naïve Bayes

Bayesian Network

Dynamic Bayesian Network

Hidden Markov Model

Factor Graphs

OFFLINE ANNOTATION ON PAST ATTACKS

OFFLINE LEARNING OF PATTERNS

PROBABILISTIC GRAPHICAL MODELS

Observed Security events

Unknown attack stages

Factor function

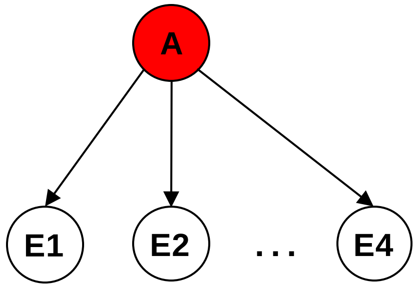
* Attack detected and stopped before the system misuse

ϵ_1 vulnerability scan
ϵ_2 login
ϵ_3 sensitive_uri
ϵ_4 new_library

σ_0 benign
σ_1 discovery
σ_4 privilege escalation
σ_5 persistence

Note: ϵ_i *is the corresponding value of an event* E_t

Modeling the credential stealing attack using Naïve Bayes vs. Bayesian Network

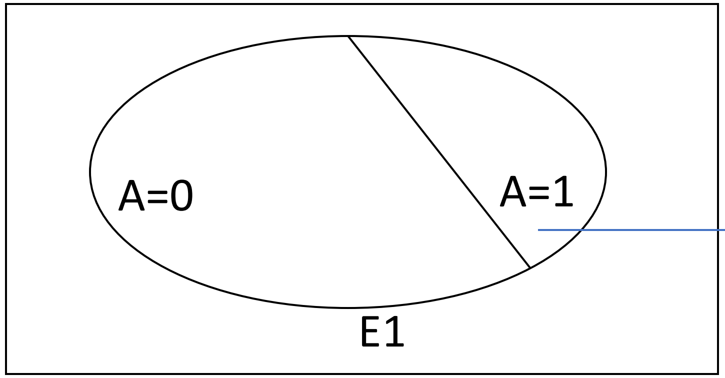


Naïve Bayes

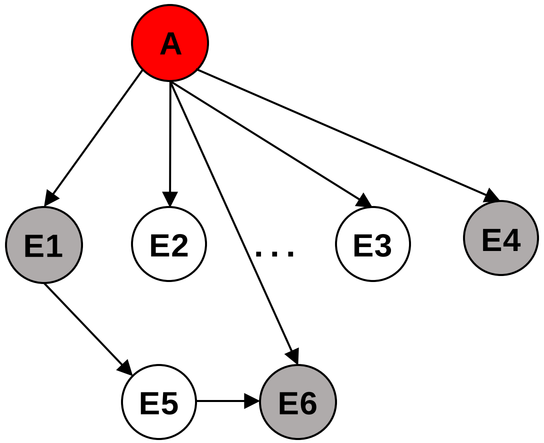
$$P(A, E_1, E_2, \ldots, E_4) = P(A) \prod_i P(E_i|A)$$

Is (E1, E2, ..., E4) represents Benign activity?

$$[P(\mathrm{E}_1|\mathrm{A} = \mathrm{Benign}) \ldots . P(E_4|\mathrm{A} = \mathrm{Benign})]P(\mathrm{A} = \mathrm{Benign}) > [P(E_1|A = Attack) \ldots P(E_4|\mathrm{A} = \mathrm{Attack})]P(\mathrm{A} = \mathrm{Attack})$$



$P(E_1|A = 1)$



Bayesian Network

ID	Description
A	Attack
E1	Vulnerability scan
E2	Login
E3	Download file with sensitive extension
E4	New system library
E5	High network flows
E6	Service unavailable

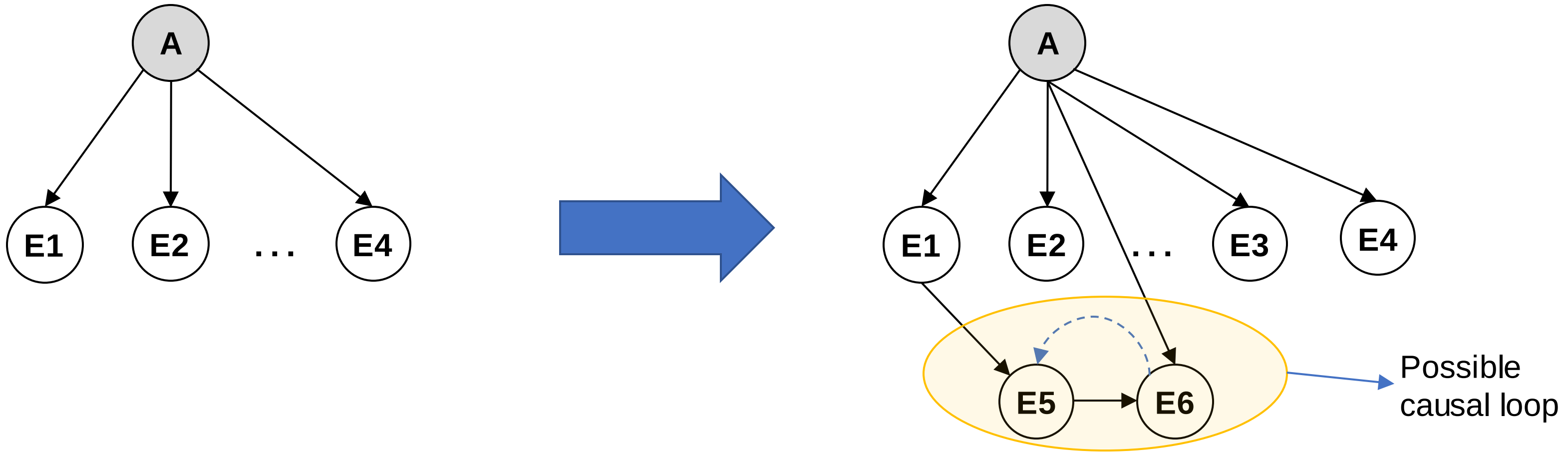
Joint Distribution: $P(E_1, E_2, \ldots, E_n, A) = P(A) \prod_{i=1}^{n} P(E_i|parents(E_i))$

Hypothesis:

$$P(A = attack|E_1, E_4\ E_6) = ?$$

$$P(A = benign|E_1, E_4\ E_6) = ?$$

Modeling the credential stealing attack using Naïve Bayes vs. Bayesian Network



Naïve Bayes **Bayesian Network**

ID	Description
A	Attack
E1	Vulnerability scan
E2	Login
E3	Download file with sensitive extension
E4	New system library
E5	High network flows
E6	Service unavailable

Model assumptions

1. All events share the same parent variable
2. All events are conditionally independent

Advantage:

Simplify calculation of posterior probability on A

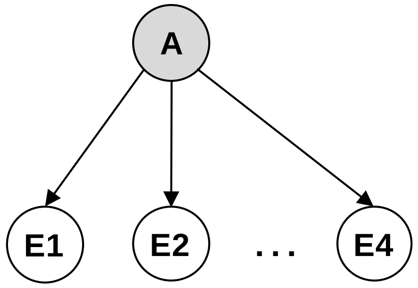
Model assumptions

1. An event can be preceded (causal) by another event
2. There is no cycle in the network

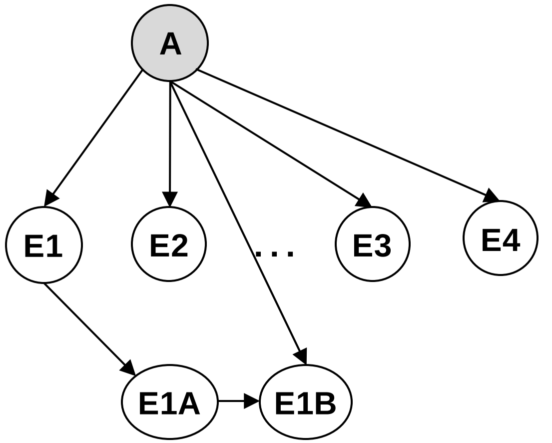
Disadvantage

Explicitly assume causal relationships
(Causality may not be clear from the data)
For complicated attacks, causal loops may form and render the BN invalid

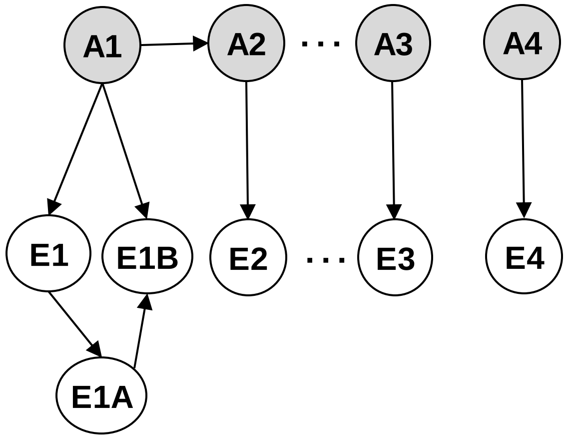
Modeling the credential stealing attack using Naïve Bayes vs. Bayesian Network



Naïve Bayes

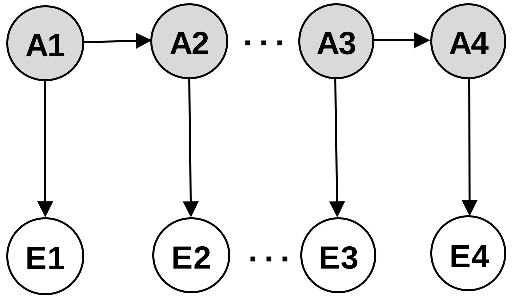


Bayesian Network



Dynamic Bayesian Network

- When we consider the time evolution of the BN each variable in each timestep together, e.g., *t* and *t+1,* we have a Dynamic Bayesian Network that captures the first-order dependency --> referred to as the Markov Property
- This concept can be extended to higher order dependencies e.g on , t-2, t-3, … and is called a higher-order Markov property, e.g., 2nd or 3rd Markov property.

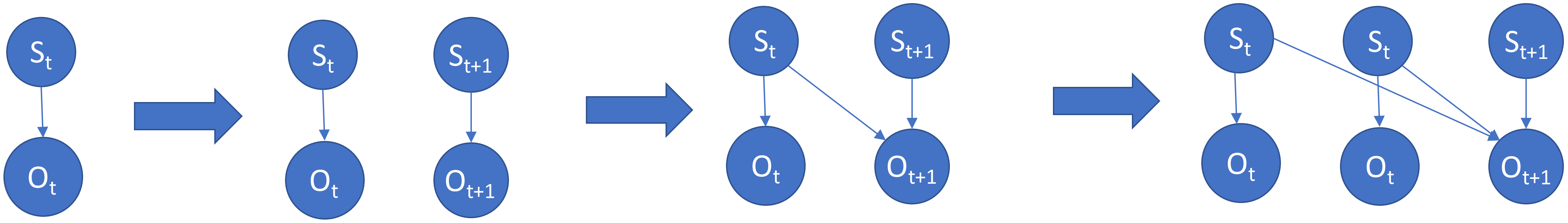


$$P(A_1, E_1, \ldots, A_n, E_n) = P(A_1)P(E_1|A_1) \ldots P(E_{t+1}|A_{t+1})P(A_{t+1}|A_t)$$

Hidden Markov Model

Dynamic Bayesian Networks

- We have considered BNs with a static set of random variables, e.g., two variables: only one measurement variable and one state variable of the system.
- In reality, data is often time series in which each time step *t* has one measurement variable O_t and one state variable S_t . Thus, the number of random variables is proportional with the number of timesteps.
- Without correlating the random variables in each timestep, we have T disconnected BNs
- When we correlate each variable in each timestep together, e.g., *t* and *t+1,* we have a Dynamic Bayesian Network that captures the first-order Markov property.
- This concept can be extended for t, t+1, t+2, … and is called a higher-order Markov property, e.g., 2^{nd} or 3^{rd}



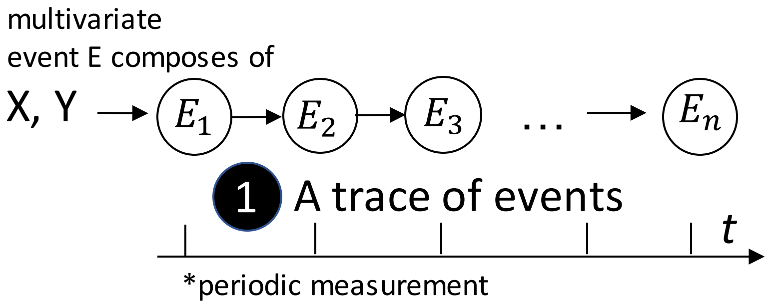
$P(S_t, O_t) = P(S_t)P(O_t|S_t)$

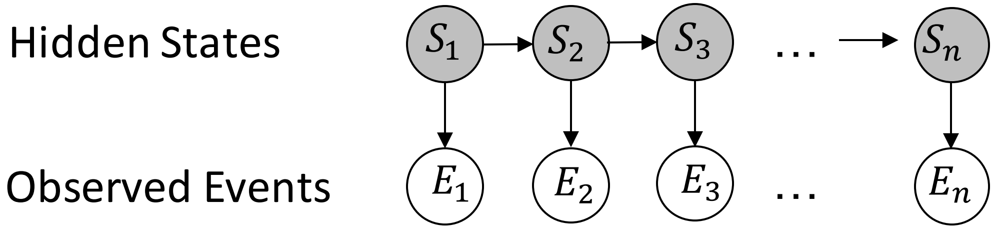
$P(S_t, O_t) = P(S_t)P(O_t|S_t)$

$P(S_{t+1}, O_{t+1}) = P(S_{t+1})P(O_{t+1}|S_{t+1})$

$P(S_t, S_{t+1}, O_t, O_{t+1}) = P(S_t)P(O_t|S_t)P(O_{t+1}|S_t, S_{t+1})P(S_{t+1})$

From a trace of events to a Hidden Markov Model





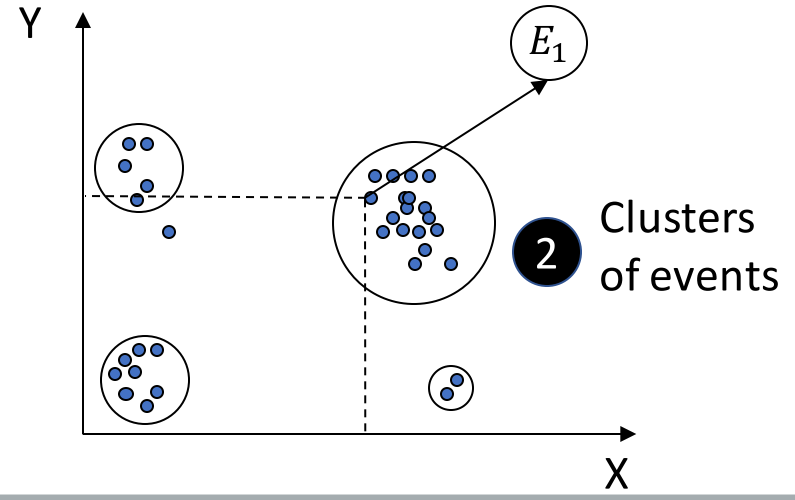
4

$$P(S_1, \dots, S_n, E_1, \dots, E_n) = p(S_1)P(E_1|S_1)\prod p(S_i|S_{i-1})P(E_i|S_i)$$

A Hidden Markov Model

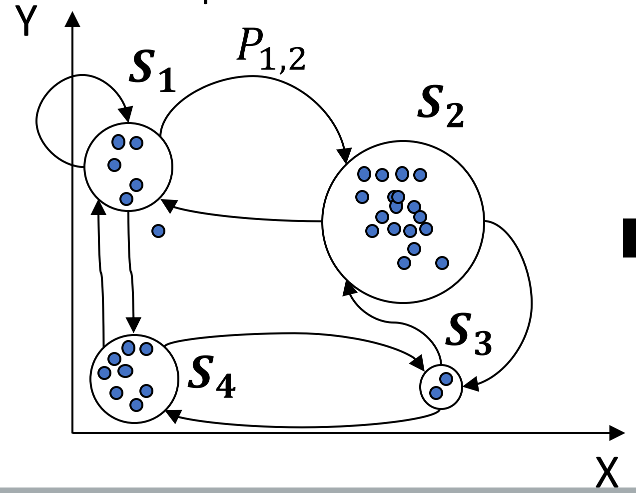
Observation

Underlying states



Two properties:

- A current state depends on the immediate past state
- Time spent in each state is an exponential distribution



$$Q = \begin{bmatrix} P_{1,1} & \cdots & P_{1,4} \\ \vdots & \ddots & \vdots \\ P_{n,1} & \cdots & P_{n,n} \end{bmatrix}$$

3 Transition matrix

Hidden Markov Models

Model assumptions

An observation depends on its hidden state

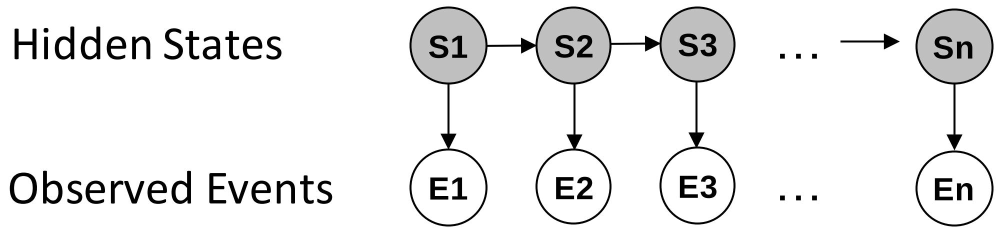
A state variable only depends on the immediate previous state (Markov assumption)

The future observations and the past observations are conditionally independent given the current hidden state

Advantages:

HMM can model sequential nature of input data (future depends on the past)

HMM has a linear-chain structure that clearly separates system state and observed events.



$$P(S_1, \dots, S_n, E_1, \dots, E_n) = p(S_1)P(E_1|S_1)\prod p(S_i|S_{i-1})P(E_i|S_i)$$

A Hidden Markov model on observed events and system states

Markov Model

- Consider a system which can occupy one of *N* discrete *states* or *categories*

$$x_t \in \{1, 2, \ldots, N\} \longrightarrow \text{state at time } t$$

- We are interested in *stochastic* systems, in which state evolution is random
- Any *joint* distribution can be factored into a series of *conditional* distributions:

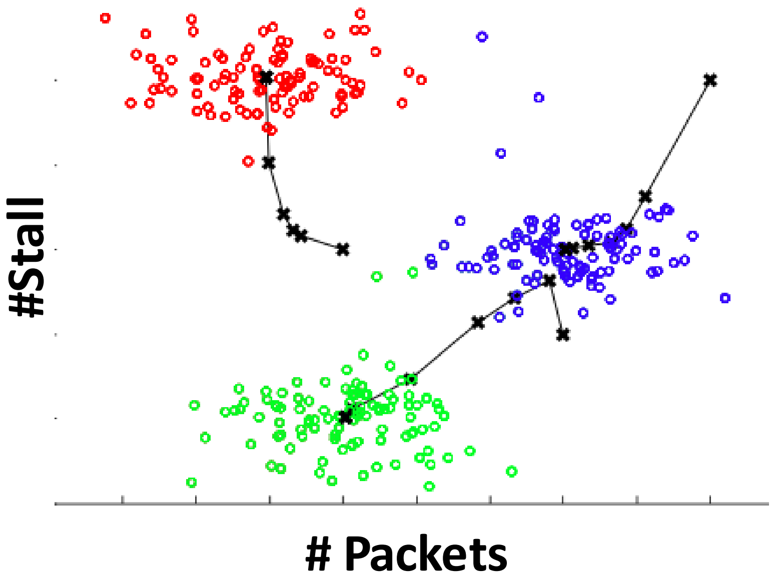
$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_0, \ldots, x_{t-1})$$

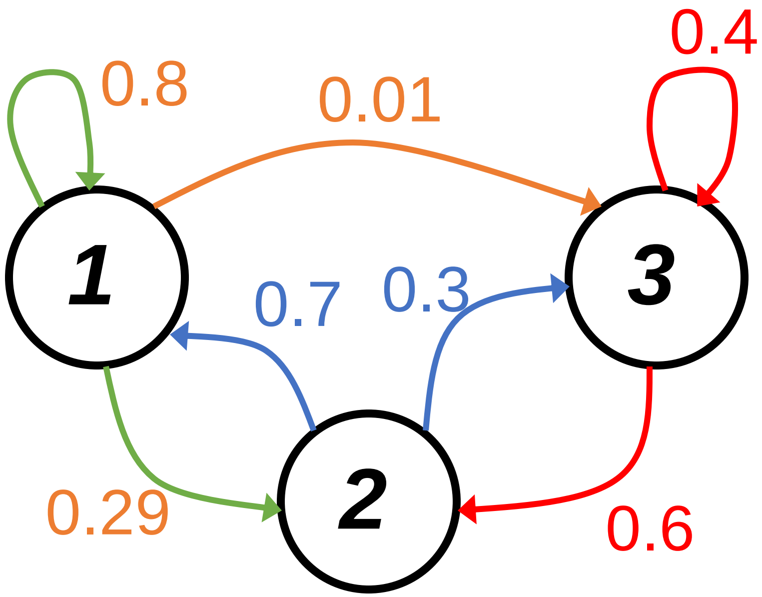
- For a *Markov* process, the next state depends only on the current state:

$$p(x_{t+1} \mid x_0, \ldots, x_t) = p(x_{t+1} \mid x_t)$$

State Transition Diagrams

Stall experienced by each packet



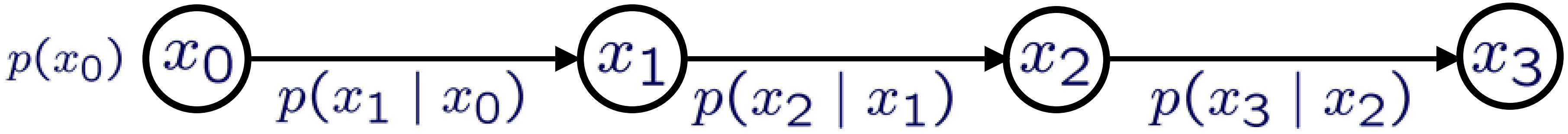


$$q_{ij} \triangleq p(x_{t+1} = i \mid x_t = j)$$

- Think of a particle randomly following an arrow at each discrete time step
- Most useful when *N* small, and *Q* *sparse*

Markov Chains: Graphical Models

$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1})$$



$$Q = \begin{bmatrix} 0.80 & 0.7 & 0.0 \\ 0.29 & 0.0 & 0.6 \\ 0.01 & 0.3 & 0.4 \end{bmatrix}$$

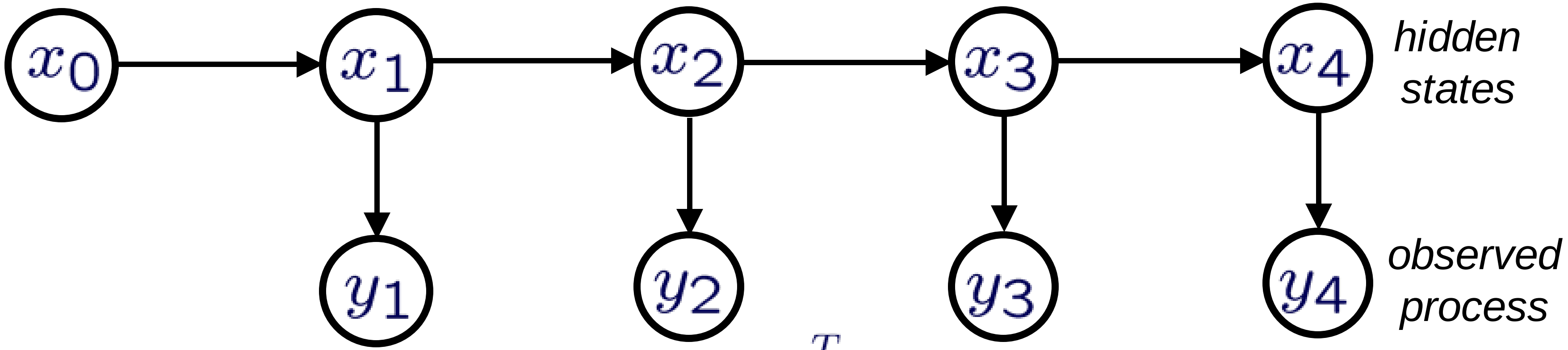
Constraints on valid transition matrices:

$$q_{ij} \geq 0 \,, \quad \sum_{i=1}^{N} q_{ij} = 1 \quad \text{for all } j$$

$$q_{ij} \triangleq p(x_{t+1} = i \mid x_t = j)$$

Hidden Markov Models

- Stall exists due to congestion
- Not directly measurable at runtime (hidden)
- Motivates *hidden Markov models (HMM):*



$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1}) p(y_t \mid x_t)$$

Given x_t, previous observations impact future observations

$$p(y_t, y_{t+1}, \ldots \mid x_t, y_{t-1}, y_{t-2}, \ldots) = p(y_t, y_{t+1}, \ldots \mid x_t)$$

State Transition Matrices

- A *stationary* Markov chain with *N* states is described by an *NxN* *transition matrix:*

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix}$$

$$q_{ij} \triangleq p(x_{t+1} = i \mid x_t = j)$$

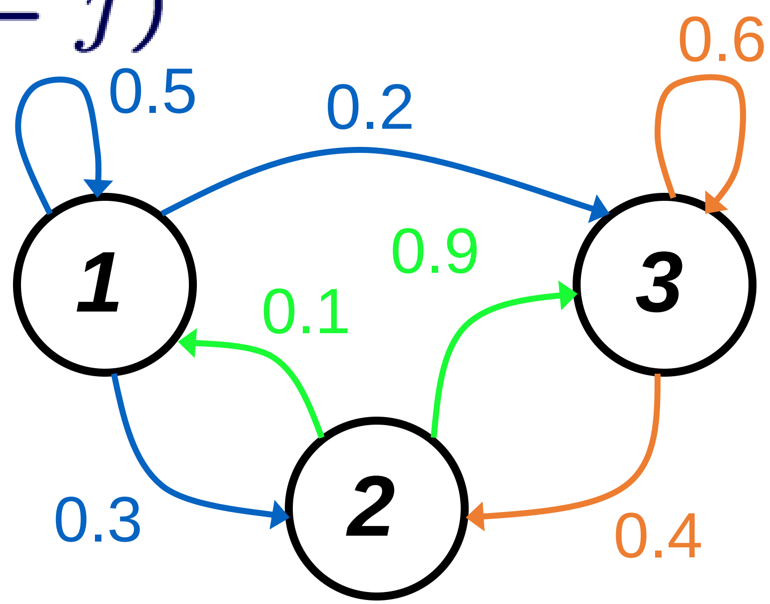
- Constraints on valid transition matrices:

$$q_{ij} \geq 0 \qquad \sum_{i=1}^{N} q_{ij} = 1 \quad \text{for all } j$$

State Transition Diagrams(Another Example)

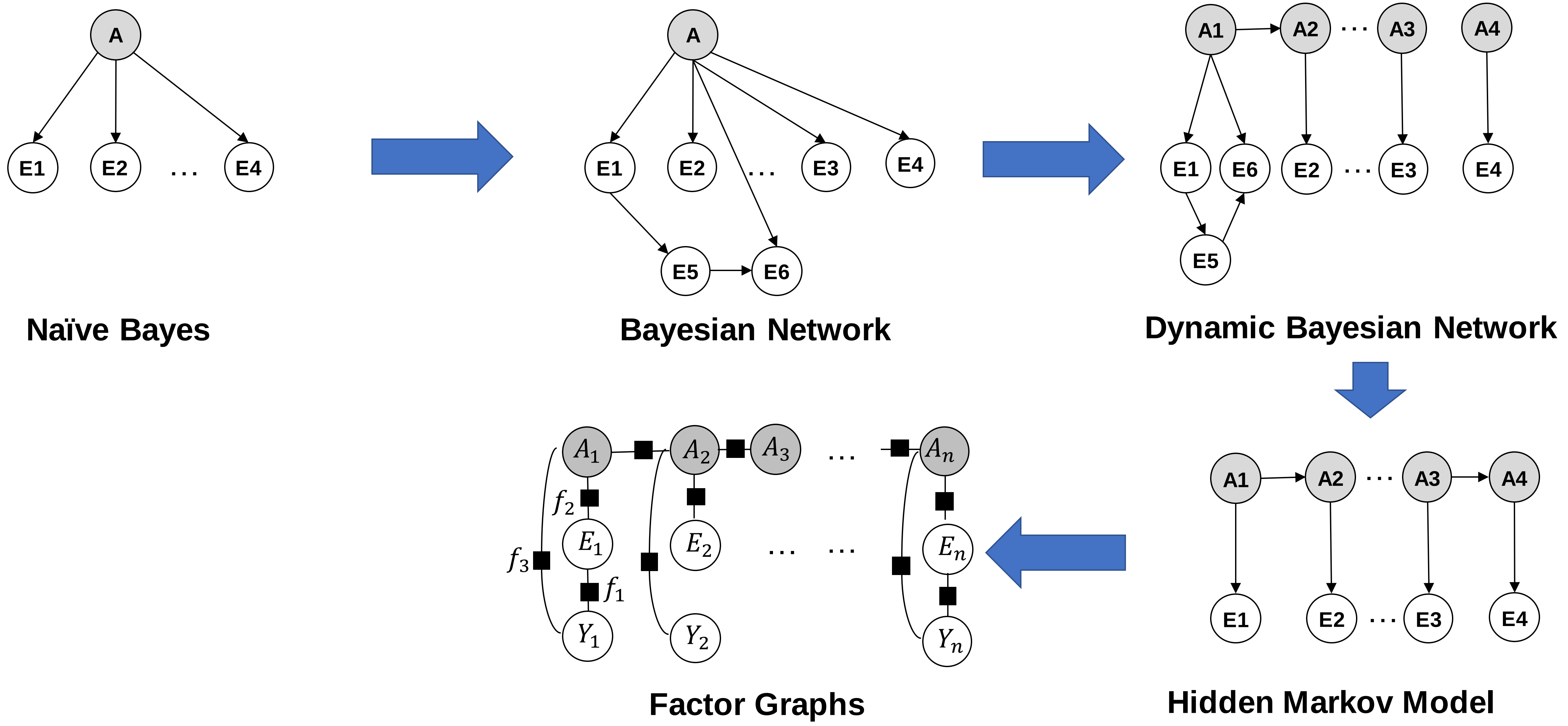
$$q_{ij} \triangleq p(x_{t+1} = i \mid x_t = j)$$

$$Q = \begin{bmatrix} 0.5 & 0.1 & 0.0 \\ 0.3 & 0.0 & 0.4 \\ 0.2 & 0.9 & 0.6 \end{bmatrix}$$



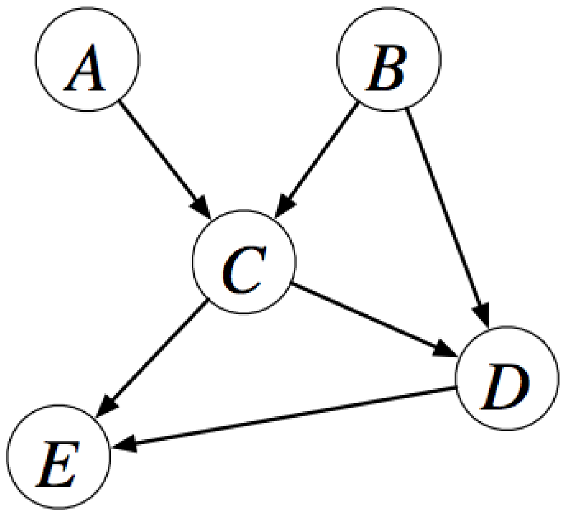
- Think of a particle randomly following an arrow at each discrete time step
- Most interesting when *Q* *sparse*

Modeling the credential stealing attack using Naïve Bayes vs. Bayesian Network



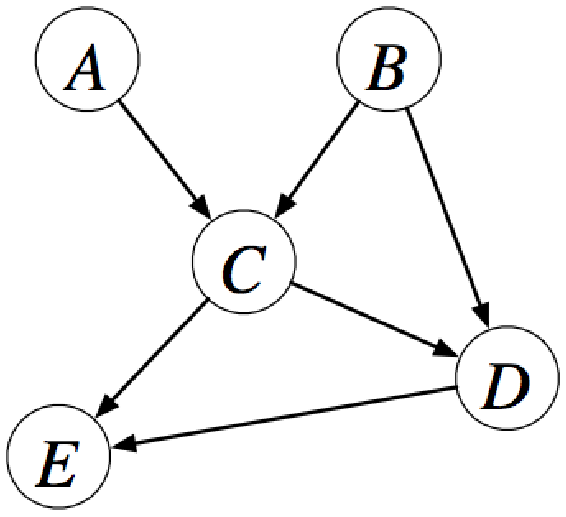
Representing knowledge through graphical models

- A PGM encodes structural aspects of a joint probability distribution
  - G = {V,E}
- A node corresponds to a random variable
- An edge represent a dependencies between the variables



Why do we need graphical models?

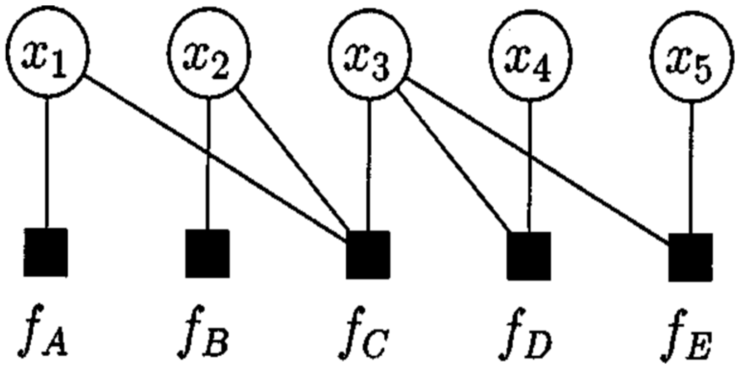
- Graphs are an intuitive way of visualizing relationship among variables
- A graph shows the conditional independence between variables via edges
- Effective inference algorithms can be run on graphs such as belief propagation to infer marginal probabilities of variables



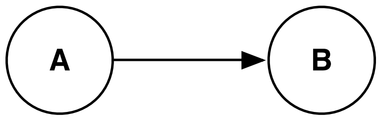
Definition of a Factor Graph

A factor graph is a **bipartite, undirected graph of random variables and factor functions. [Frey et. al. 01]**

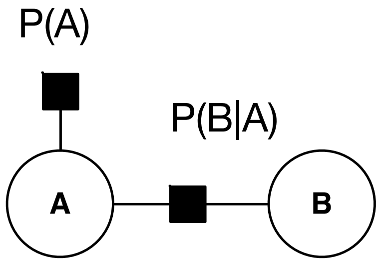
A factor function is a mathematical definition of ***prior beliefs*** or expert knowledge. *FG can represent both* ***causal and non-causal relations***.



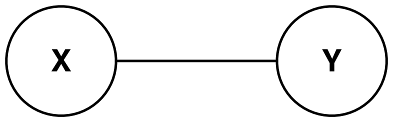
A factor graph for the product $f_A(x_1)f_B(x_2)f_C(x_1, x_2, x_3) \cdot f_D(x_3, x_4)f_E(x_3, x_5)$.



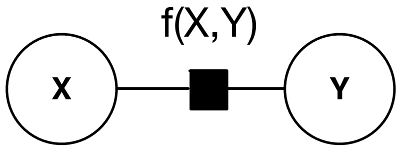
Bayesian Network (BN)



Factor Graph equivalent of BN



Markov Random Fields (MRF)



Factor Graph equivalent of MRF

Applications of Probabilistic Graphs in Security Domain

Problem statement. Given a set of security events, infer whether an attack is in progress?

Modeling Approach.

Each security event is a known variable **e**, each takes value from a discrete set of events **E**.

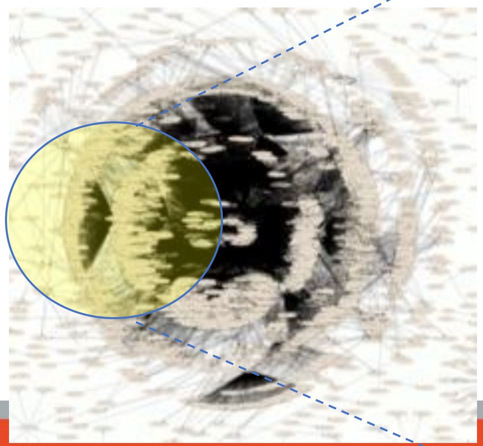
An attack happens in a chain of exploits, thus we have a sequence of events in time dimension.

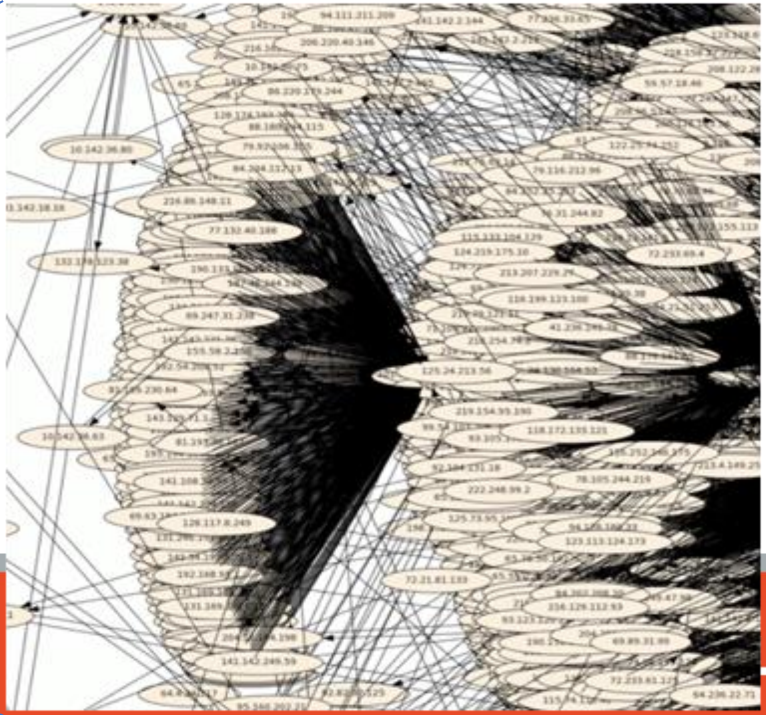
Each event is associated with a corresponding attack state **s,** which is unknown. The simplest approach is to classify **s** as a binary {0,1}. However, when we can infer **s** it is often too late (the attacker is already in the system)

Thus, we want to discretize **s** to smaller attack stages and provide update on such stages as soon as an event is observed.

Measurements from NCSA@Illinois: Five-minute snapshot

- Goals:
  - Provide a system-level characterization of incidents and evaluate the intricacies of attack maturation in real-time.
  - Design protection strategies to reduce missed incidents and false positives
  - Experimentally demonstrate new techniques in a sandbox, embedded in production network
- Challenges:
  - **Big data**

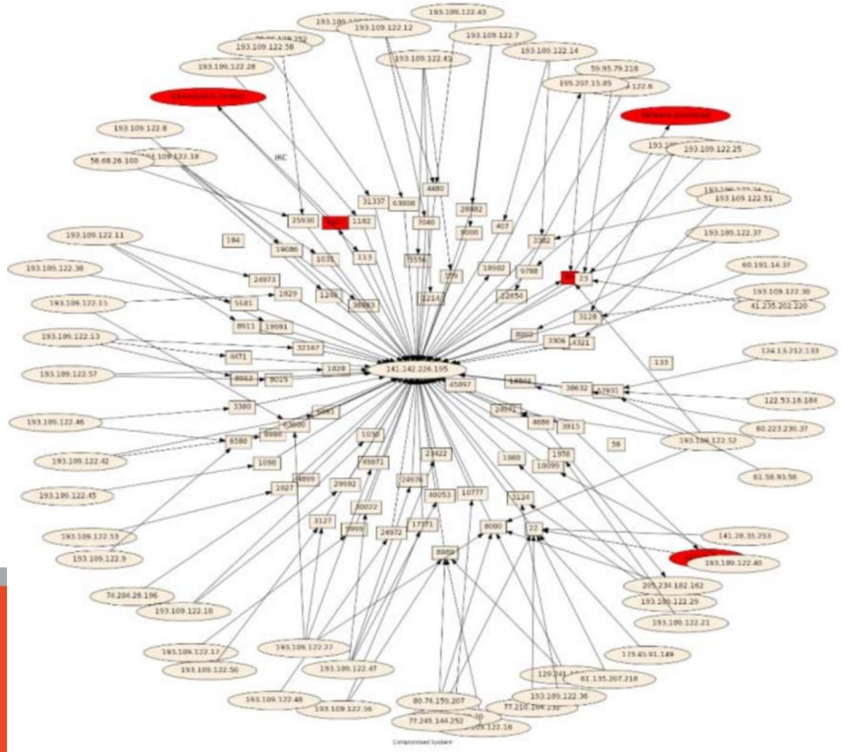




Five-Minute Snapshot of In-and-Out Traffic at NCSA

Measurements from NCSA@Illinois: Five-minute snapshot

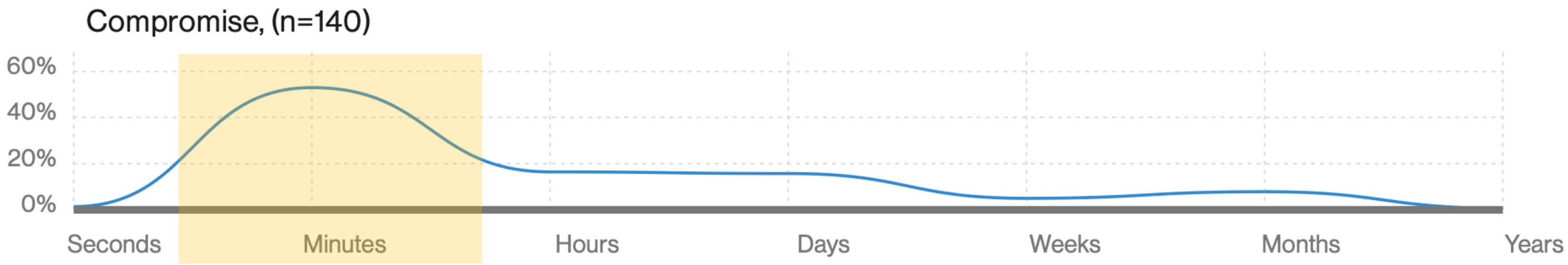
- Goals:
  - Provide a system-level characterization of incidents and evaluate the intricacies of attack maturation in real-time.
  - Design protection strategies to reduce missed incidents and false positives
  - Experimentally demonstrate new techniques in a sandbox, embedded in production network
- Challenges:
  - Big data
  - **Partial view of attacks**

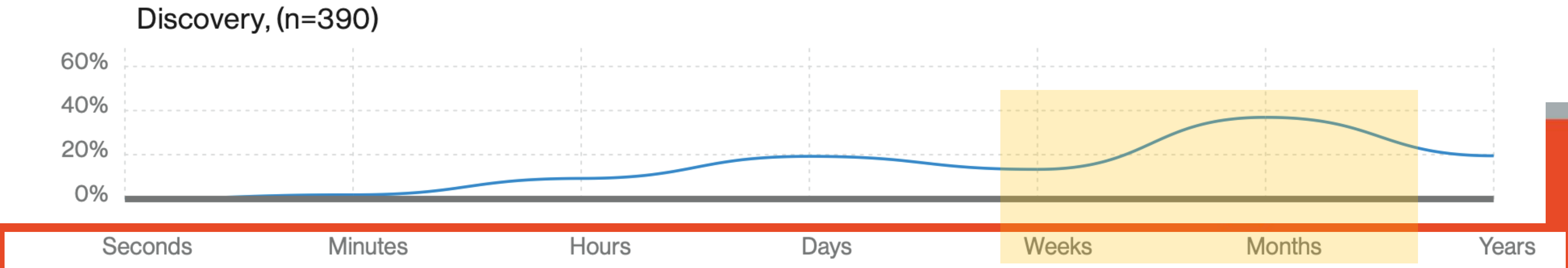


62% (23/37) OF HIGH-SEVERITY INCIDENTS WERE CAUGHT IN THE BREACH-PHASE, HAVING ALREADY RESULTED IN SIGNIFICANT DAMAGE

Measurements from NCSA@Illinois: Five-minute snapshot

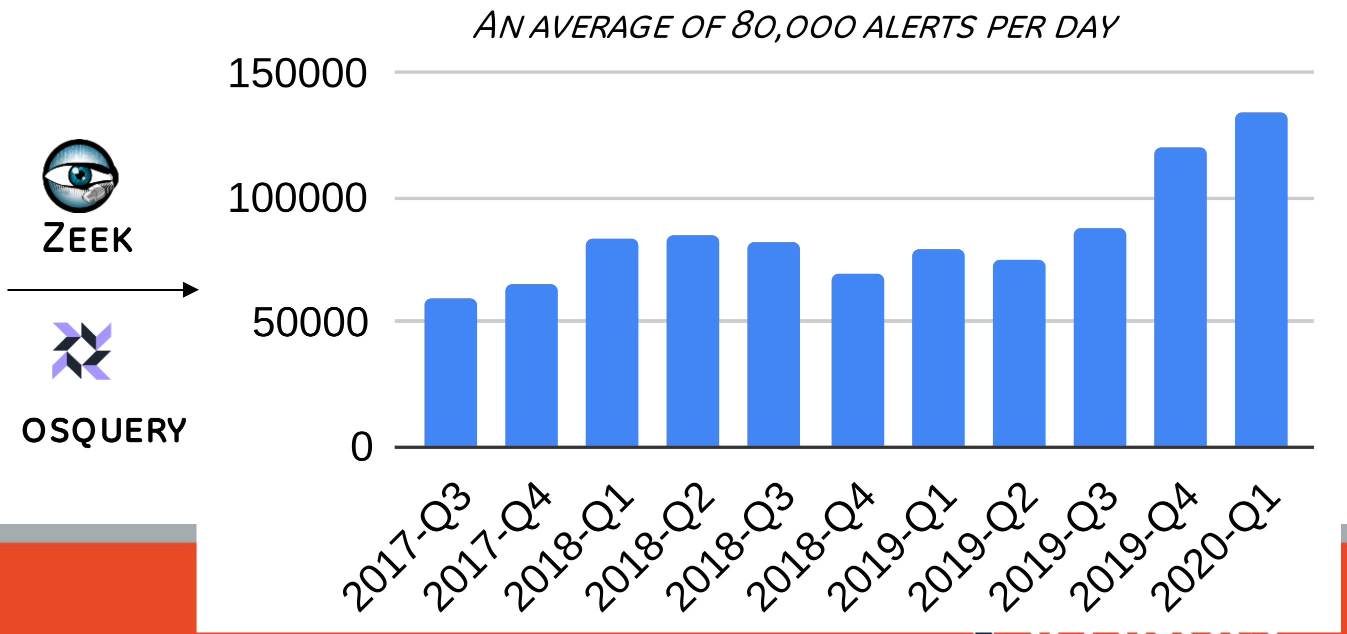
- Goals:
  - Provide a system-level characterization of incidents and evaluate the intricacies of attack maturation in real-time.
  - Design protection strategies to reduce missed incidents and false positives
  - Experimentally demonstrate new techniques in a sandbox, embedded in production network
- Challenges:
  - Big data
  - **Fast attacks**





Measurements from NCSA@Illinois: Five-minute snapshot

- Goals:
    - Provide a system-level characterization of incidents and evaluate the intricacies of attack maturation in real-time.
    - Design protection strategies to reduce missed incidents and false positives
    - Experimentally demonstrate new techniques in a sandbox, embedded in production network
- Challenges:
    - Big data
    - Fast attacks
    - **Many alerts**



Measurements from NCSA@Illinois: Five-minute snapshot

- Goals:
  - Provide a system-level characterization of incidents and evaluate the intricacies of attack maturation in real-time.
  - Design protection strategies to reduce missed incidents and false positives
  - Experimentally demonstrate new techniques in a sandbox, embedded in production network
- Challenges:
  - Big data
  - Fast attacks
  - Many alerts
  - Partial view of attacks
  - **Impractical evaluation**

Synthetic attacks

Lab Environment ⟷ Production Network

Real attacks
- Data Exfiltration — EQUIFAX
- Denial of Service — Apache Tomcat
- System Misuse — SSH

INPUT DATA FOR EACH INCIDENT

1. PAST INCIDENTS → 2. PATTERNS → 3. PROBABILITIES → 4. MODEL → 5. EVALUATION

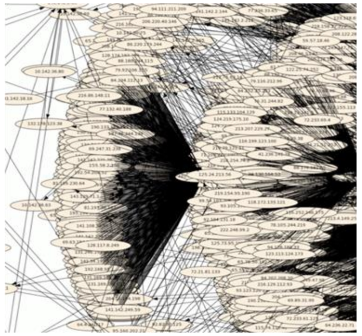
INCIDENT 1

INCIDENT 2

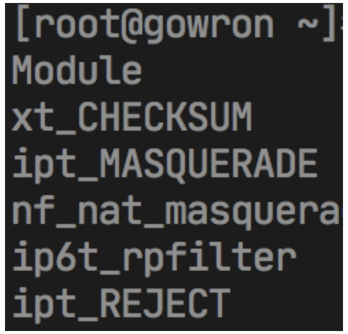
....

INCIDENT 120

1. RAW LOGS CONTAINING ATTACKS AND ALERTS



NETWORK FLOWS



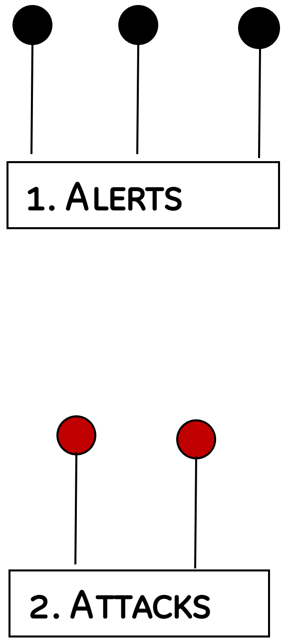
HOST LOGS

2. INCIDENT REPORT CONTAINING COMPROMISED USER ID

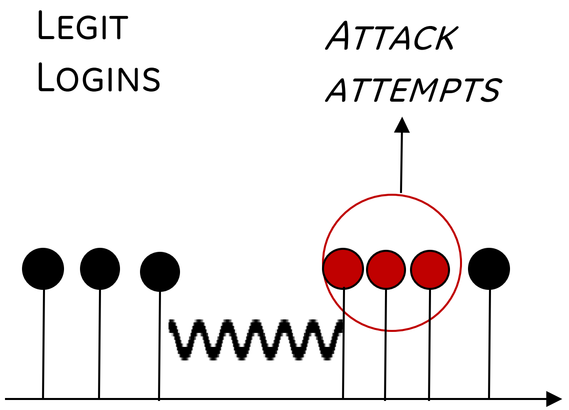
USER X PASSWORD HAS BEEN COMPROMISED

Combining alerts and attacks of an incident into a single stream of security events

1. Past Incidents → 2. Patterns → 3. Probabilities → 4. Model → 5. Evaluation



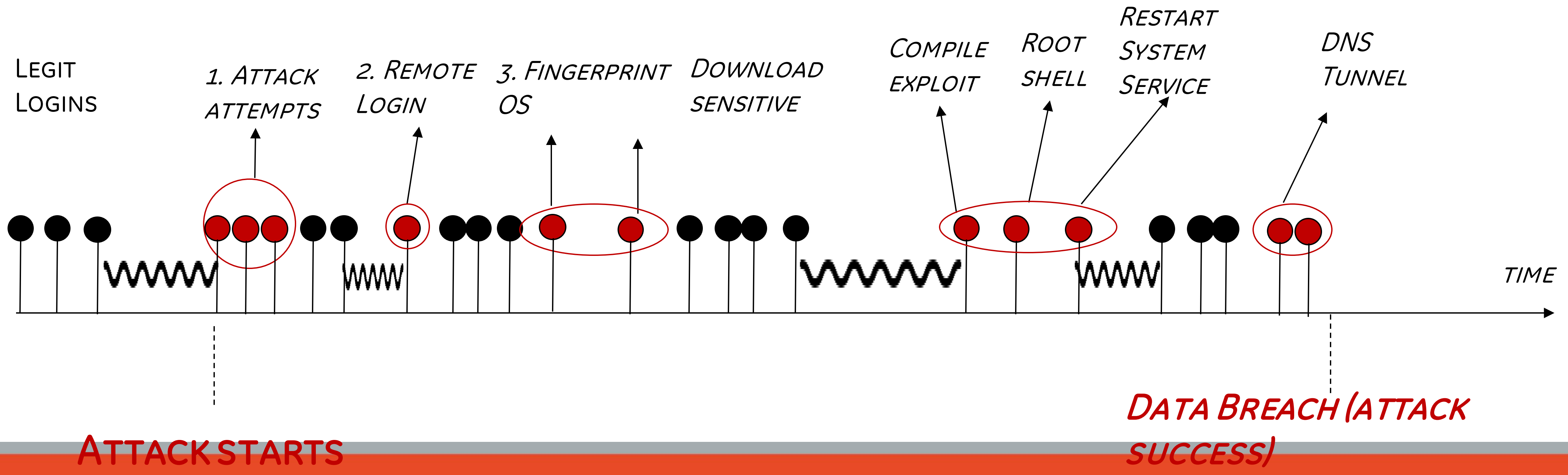
→ Stream of security events



An example stream of security events

The evolution of a credential stealing attack

- Attack starts with repeated login attempts, and ends with data exfiltration through a network tunnel
- The attack is enabled by stolen credentials, followed by attempts to install exploits and keylogger to collect data.
- Several alerts occur in burst (close in time) and thus could be grouped into clusters.



Each attack cluster has a certain probability

A1. Remote login

A2. OS fingerprinting

A3. Download sensitive files

$P(\text{Attack}|A1) = 0.04$

$P(\text{Attack}|A2) = 0.03$

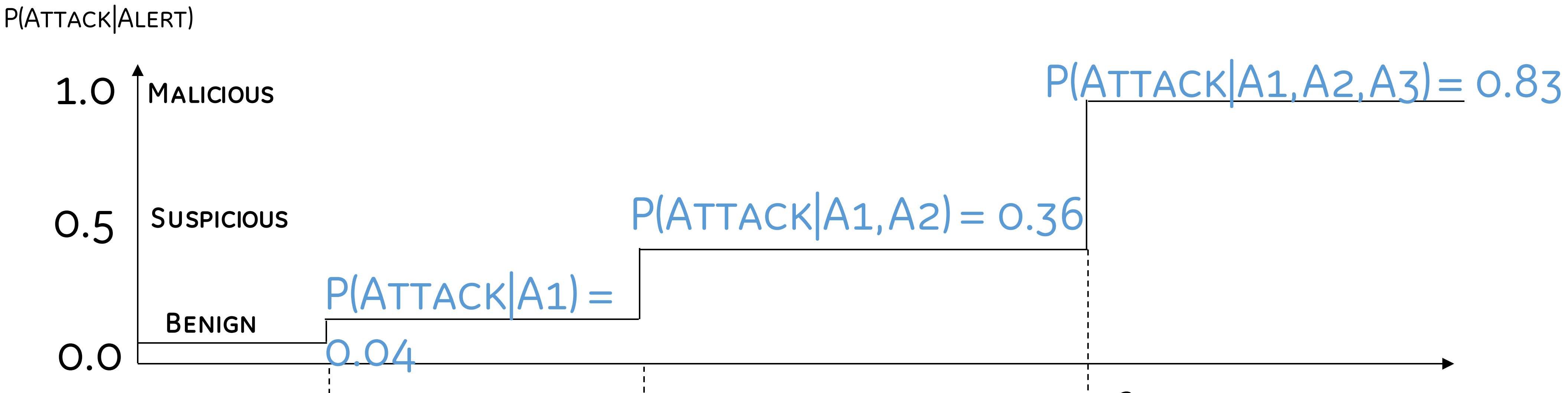
$P(\text{Attack}|A3) = 0.18$

⇩

Individual probabilities of each alert is very low and are inconclusive.

Can we fuse these alerts together to pre-empt the attack?

The suspicion level, P(Attack|Alert), increases as alerts are observed.



At network level, remote logins are common as people travel off-campus.

Cannot conclude an attack at this point.

At host level, OS fingerprinting correlates with maintenance activities, which typically performed on-site instead of off-campus.

Performing both OS fingerprinting off-site is suspicious.

Combining host and network with the act of download a source code file (.c) raises the suspicion level, exceeding threshold of 0.5.

The attack is concluded before the attacker misuses the system.

A1. Remote login

A2. OS fingerprinting

A3. Download sensitive files

Applications in the Security Domain (cont.)

Problem statement. Given a set of security events, infer whether an attack is in progress?

Formally, the problem becomes

1. Define a joint probability distribution function (joint pdf)

$P(e_1,e_2,..,e_n,s_1,s_2,...,s_n)$

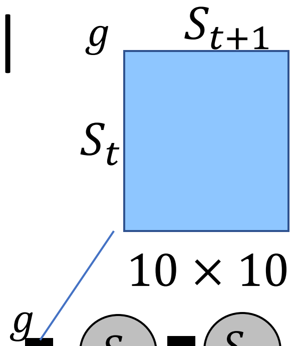
2. Derive a conditional probability

$P(e_1,e_2,..,e_n|s_1,s_2,...,s_n)$

However, the search space is exponentially large (by the order of the number of observed stages and events) and the joint pdf is sophisticated.

We want to break the joint pdf into smaller components that are easier to compute, i.e., factorize the joint pdf.

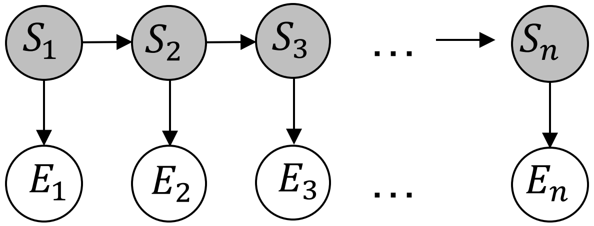
Underlying representation of a Hidden Markov Model and conversion to a Factor Graph



Hidden Markov Model

Hidden States

Observed Events



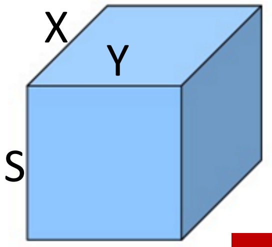
X, Y

Example

$|\boldsymbol{S}| = \mathbf{10}$

$|\boldsymbol{X}| = \mathbf{10}$

$|\boldsymbol{Y}| = \mathbf{10}$



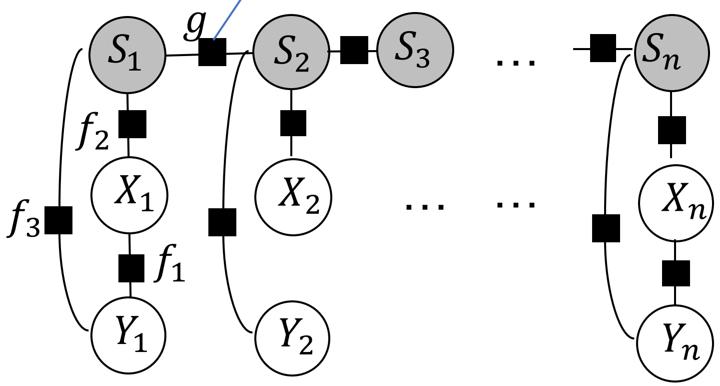
3d-tensor

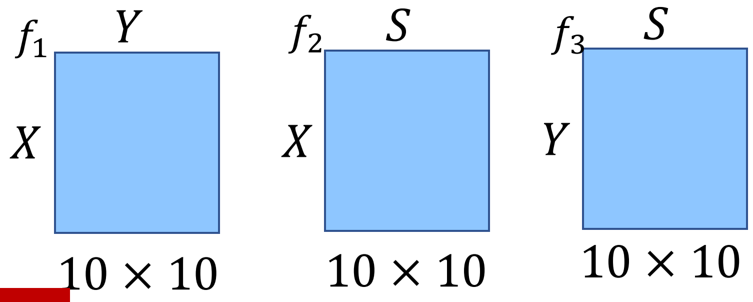
$$\textit{size of tensor}$$
$$10 \times 10 \times 10 = \mathbf{1000}$$

Factor Graph of the HMM

Hidden States

Observed Events





Domain knowledge: *"variables are pair-wise related"* **reduces dimensionality**

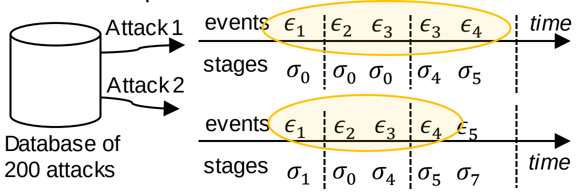
$$\textit{size of three matrices} + \textit{one transition}$$
$$10 \times 10 + 10 \times 10 + 10 \times 10 + 10x10 \;= \mathbf{400}$$

$$\mathbf{1000^{\boldsymbol{n}}} \gg \mathbf{400} \;\; \times \; \boldsymbol{n}$$

Modeling the credential stealing attack using Factor Graphs

OFFLINE ANNOTATION ON PAST ATTACKS

a) Annotated events and attack stages in a pair of attacks



b) Event-stage annotation table for the attack pair (Attack 1 and Attack 2)

Event	Attack stage
$\{\epsilon_1\}$	$\{\sigma_0 \mid \sigma_1\}$
$\{\epsilon_2\}$	$\{\sigma_0\}$
$\{\epsilon_3\}$	$\{\sigma_4\}$
$\{\epsilon_4\}$	$\{\sigma_5\}$
$\{\epsilon_5\}$	$\{\sigma_7\}$

OFFLINE LEARNING OF PATTERNS

c) Example patterns, stages, probabilities, and significance learned from the attack pair

Pattern	Attack stages	Probability in past attacks	Significance (p-value)
$[\epsilon_1, \epsilon_3, \epsilon_4]$	$[\sigma_1, \sigma_4, \sigma_5]$	q_a	p_a
$[\epsilon_1]$	$[\sigma_0 \mid \sigma_1]$	q_b	p_b

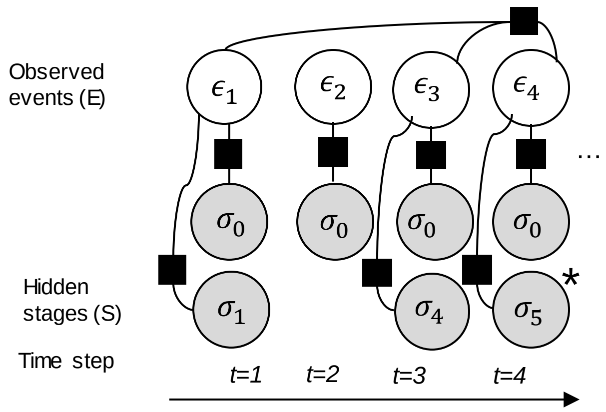
…

$$f(E) = \exp\{q_E(1 - p_E)\}$$

A factor function defined on the learned pattern, stages, and its significance

RUNTIME DETECTION OF UNSEEN ATTACKS

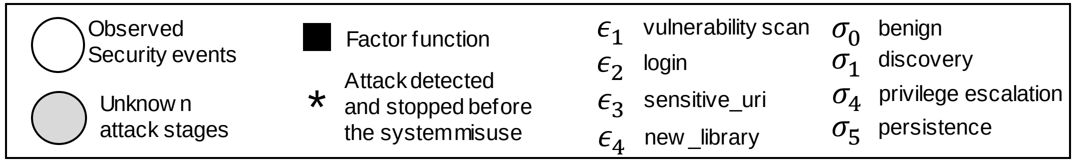
d) An evolution of the Factor Graph for the port knocking attack at run-time



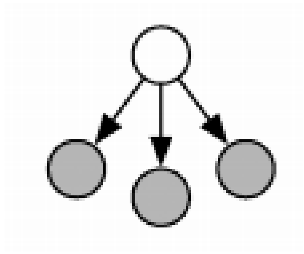
Model assumptions

1. There are multivariate relationships among the events
2. Such relationships are represented by factor functions
3. There is no restriction on order of the relationships like causal in Bayesian Network

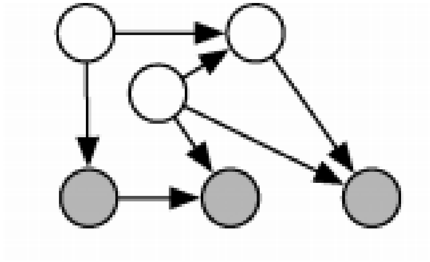
More suitable for modeling highly complex attacks, where the causal relations among the events are not immediately clear.



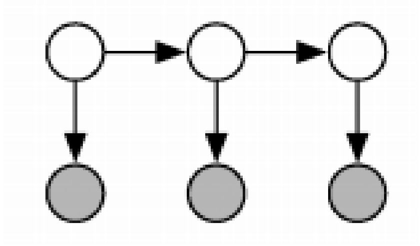
Taxonomy of graphical models



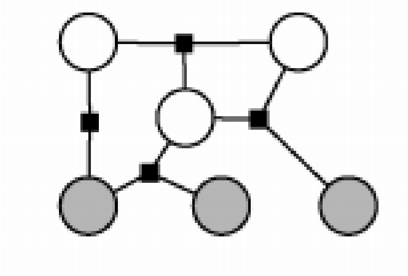
Naïve Bayes



Bayesian Network



Hidden Markov Model



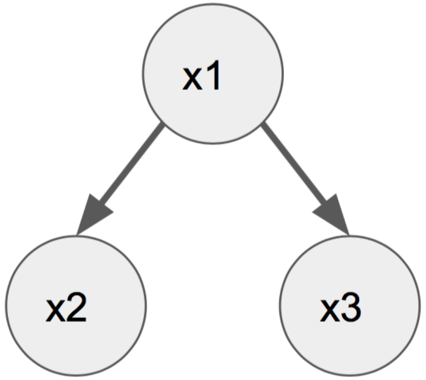
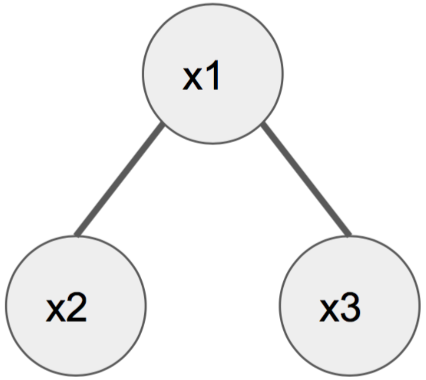
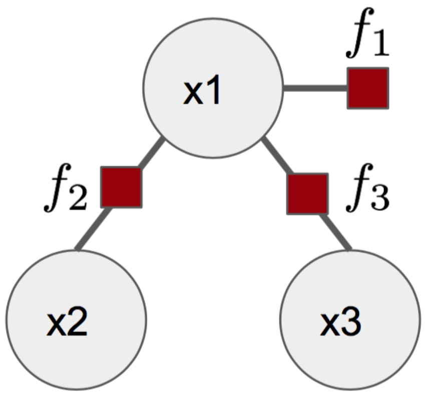
Factor Graph

Conditional probabilities and statistical dependencies can be represented by a general type of graph: Factor Graph

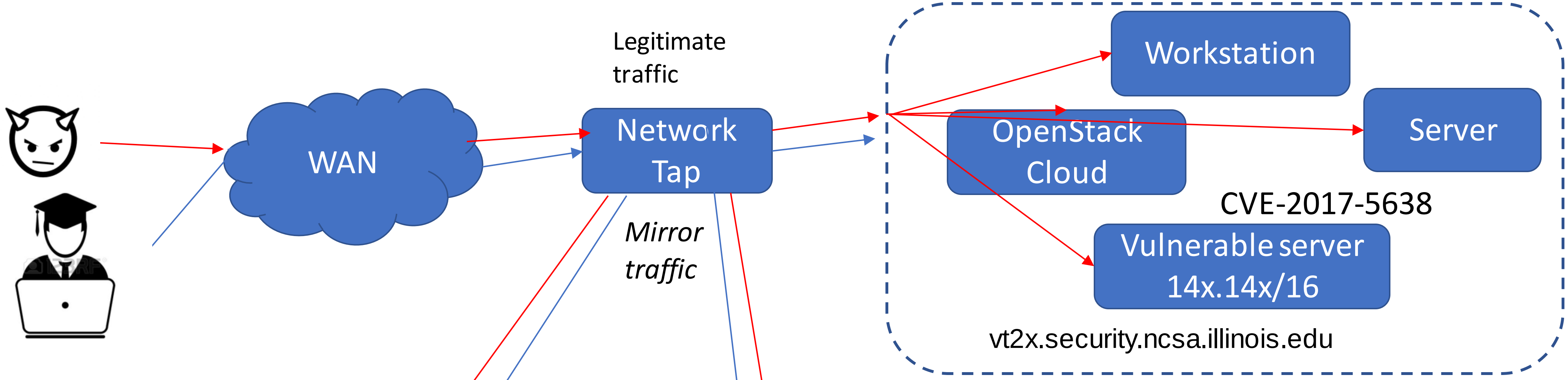
Model structure and inference in PGMs

		Naïve Bayes	Bayesian Network	Hidden Markov Model	Factor Graphs
MODEL STRUCTURE	*Graph type*	Directed	Directed	Directed	**Undirected**
	Graph structure	Parent-child	Hierarchical parent-child	Sequential	**Arbitrary structure**
	Variable of interest	Attack (0 or 1)	Attack (0 or 1)	Sequence of system states	**Sequence of attack stages**
	Relationship	Conditional independence	Prior Conditional independence	State transitions Emission of event	**Temporal relationships (patterns of events)** **Statistical relationships (severity or repetitiveness of events)**
INFERENCE	*Algorithm*	Multiplication of conditional probabilities	Multiplication of conditional probabilities and priors	Dynamic Programming	**Belief Propagation** **Sampling**

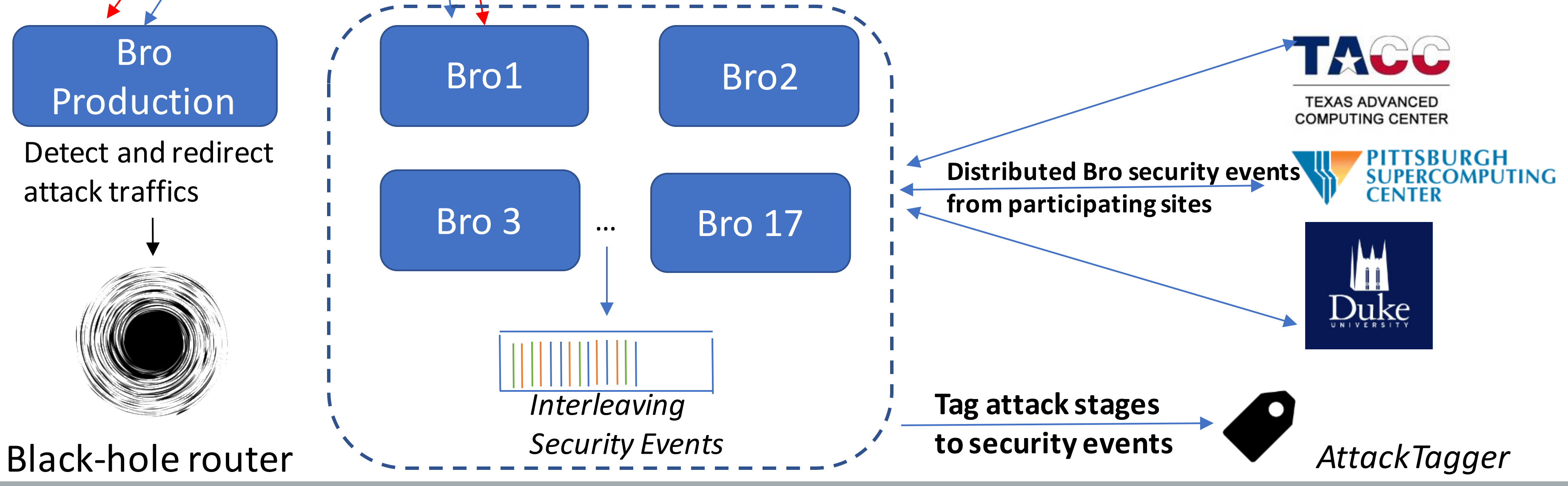
Bayesian Networks vs. Markov Random Fields vs. Factor Graphs

Bayesian networks	Markov random fields	Factor graph
		
$p(x_1)p(x_2\|x_1)p(x_3\|x_1)$	$\frac{1}{Z}\psi_1(x_1,x_2)\psi_2(x_1,x_3)$	$\frac{1}{Z}f_1(x_1)f_2(x_2,x_1)f_3(x_1,x_3)$
Product of conditional probabilities	Product of dependencies among variable cliques	Product of dependencies using univariate, bivariate, or multivariate functions
Causal relationships	Statistical dependencies	Both types of relations (including prior on a variable)

An attack testbed in real production traffic – an experiment at NCSA

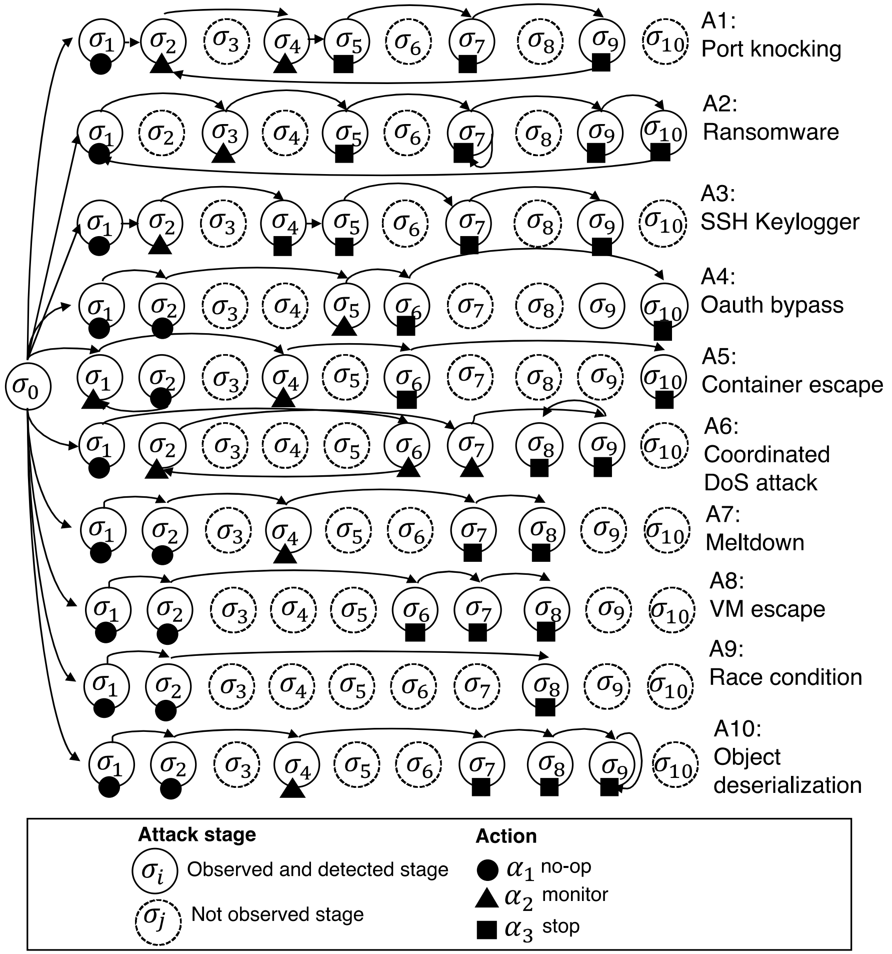


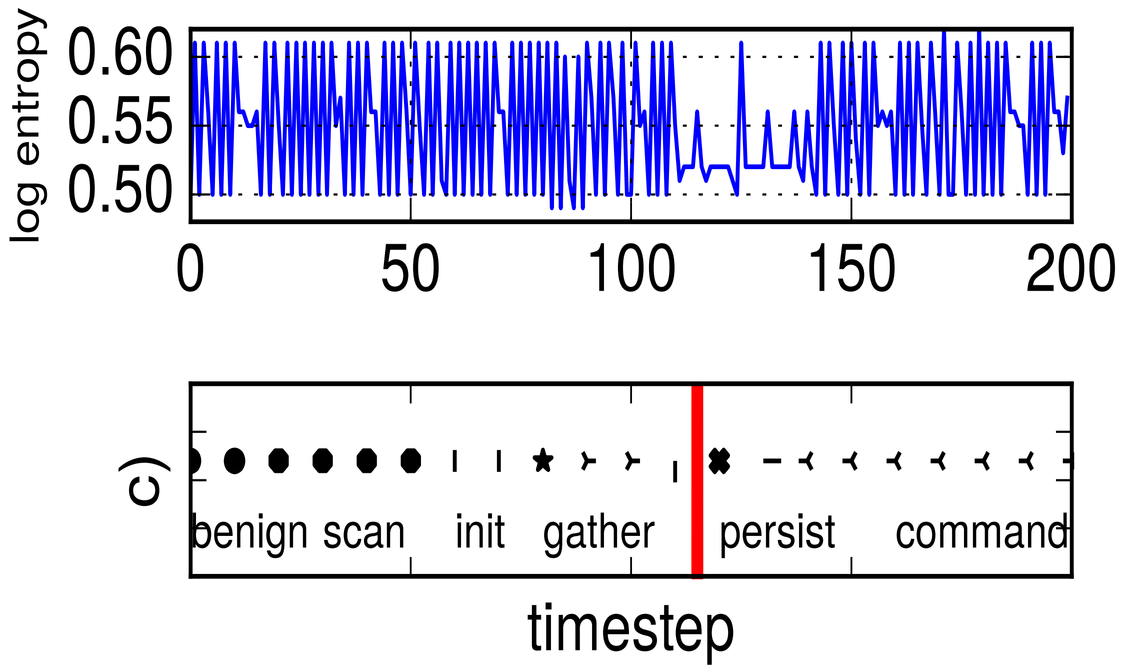
- *Advantages*
  - Attack traces include traces of legitimate users and attackers
- *Cons*
  - Requires tight coordination with security ops team



Evaluation Result

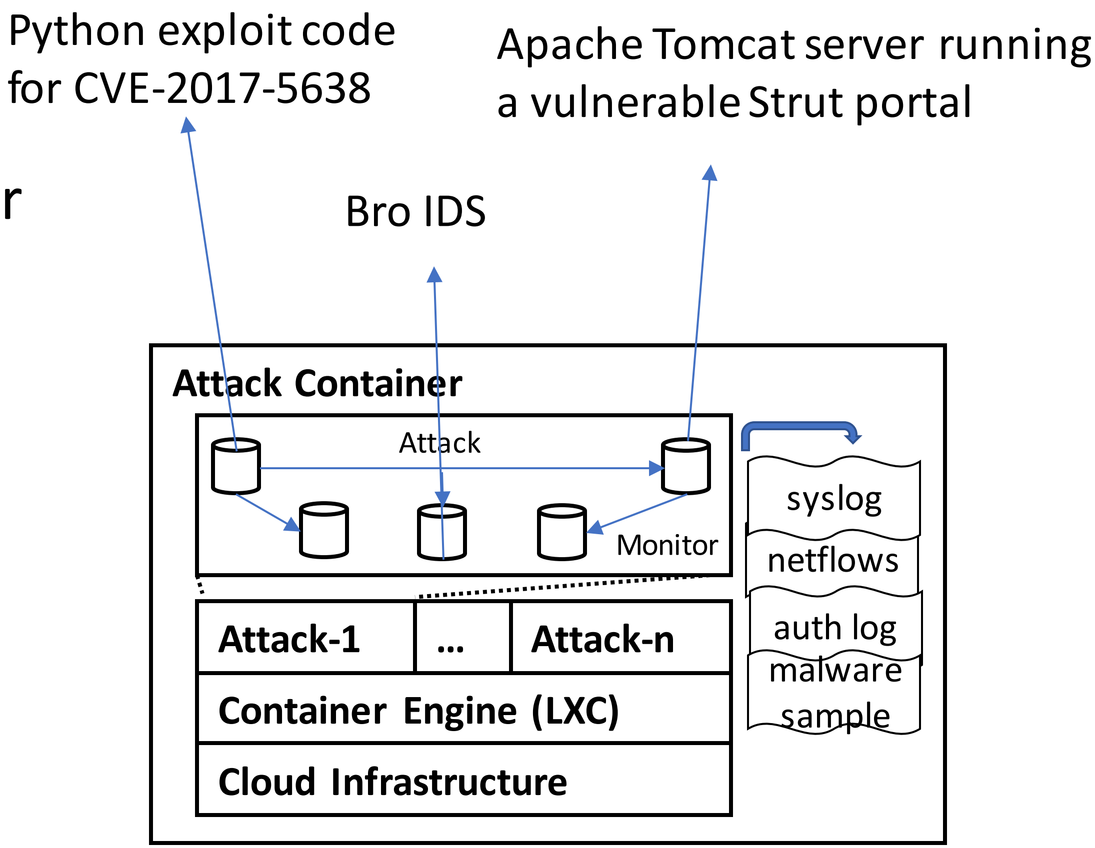
Stage transition of a multi-stage attack that exploits CVE-2017-5638



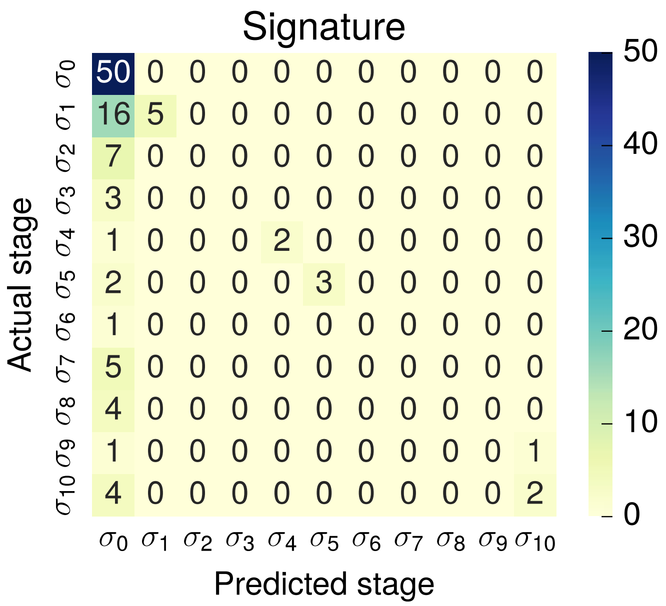


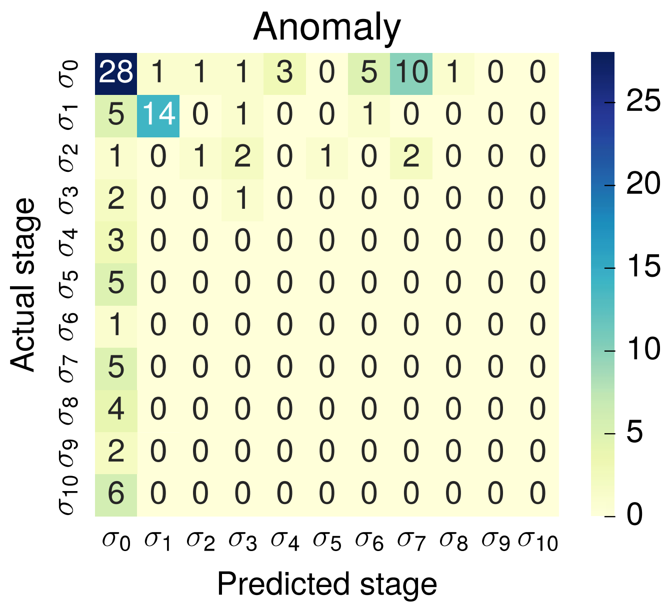
Emulating CVE-2017-5638 in a container-based environment

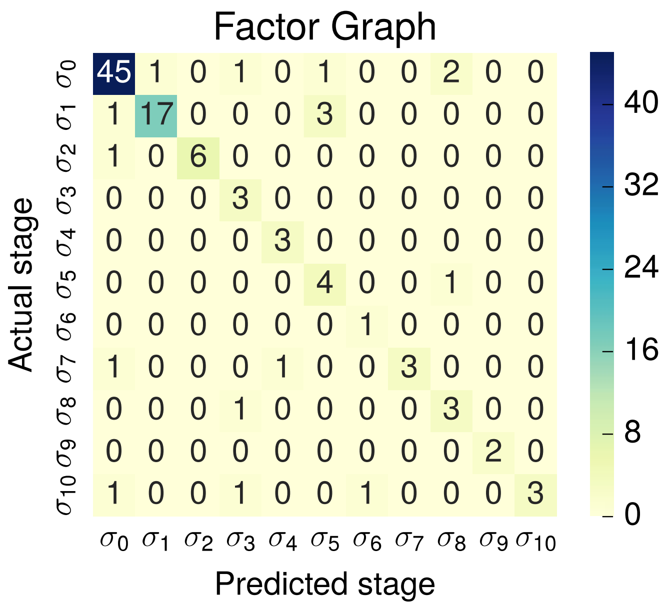
- *Advantages*
  - We were able to create an exact environment for the vulnerable Strut application
  - Monitors are in place to collect attack traces
  - Network policies are implemented to isolate potential outbreak of the attack
- *Limitations*
  - Containers are not exposed to a real network thus are not visible to attackers
  - Traces only include attack activities



Evaluation Results





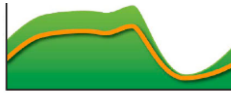


Concluding Remarks

1. Probabilistic Graphical Models appear to be the way to integrate disparate issues on failure and attack pre-emption

2. Continuous and dynamic monitoring and adaptive abstraction offered by the factor graph based learning is critical

3. Going forward: Factor graphs could combine both security logs and error logs for diagnosis

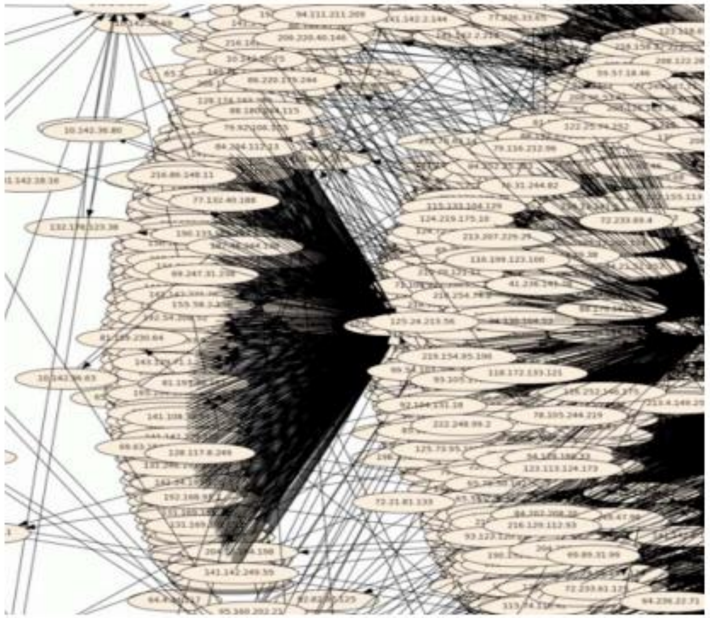


6000+
users

5+ millions
connections

34M+
log events

4.5+ GB
Compressed final log



5-minute snapshot of network traffic in and out of NCSA

Heterogeneous host and network logs

- Syslog
- Netflows
- IDS alerts
- Human-written reports

200+ incidents in the past years (2008-2017)

Brute-force attacks

Credential compromise

Abusing computing infrastructure

- Send spam
- Launch Denial of Service attacks.

NCSA

Why attack injection?

- Vulnerabilities are discovered on a daily basis, however, is a target system immune from such vulnerabilities?

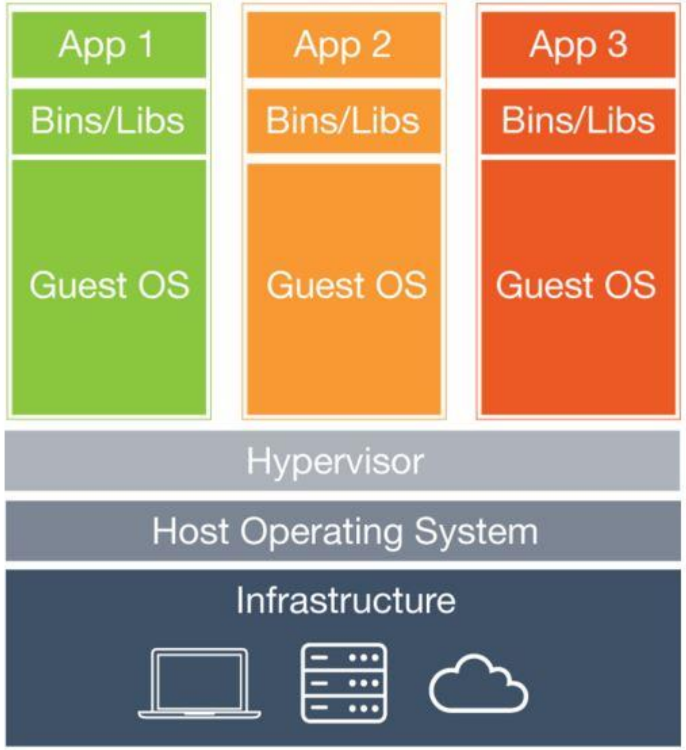
- **Our goals are to:**

  - Evaluate ability of security monitoring systems in capturing attack-related security events

  - Run live, integration tests on applied security patches

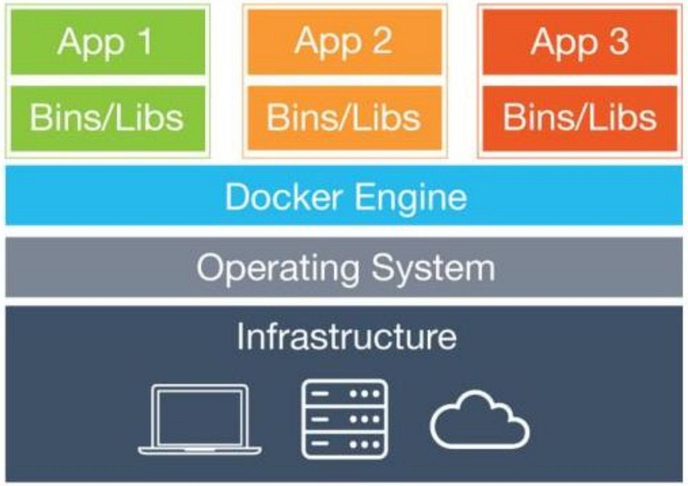
  - Provide a dynamic blueprint of an attack (in terms of attack stages) as the attack unfolds across a production network

What is a Linux Container (LXC)?



Virtual Machine (VM) is an efficient, isolated duplicate of a real computer machine.

Features	Virtual Machine	Linux Container
Emulation	A real machine	**A Linux system**
Guest OS	Almost any OS	**Only Linux system**
Isolation and Resource management	Fully virtualized	**Kernel namespace and control groups**
File system	Separated file system for each VM	**Layered filesystem (AUFS)**
Disk and Memory	GBs	**MBs**
Startup time	Minutes	**Seconds**



Linux Container (LXC) is a virtualization technology for running multiple isolated Linux systems (containers).

How does AttackTagger work?