An SVM-based Masquerade Detection Method with Online Update Using Co-occurrence Matrix

Liangwen Chen, Masayoshi Aritsugi
Gunma University, Japan

Outline

- Background
- Conventional results
- Our proposal
- Experiments
- Conclusion

Background

- A computer can provide multiple services to multiple users
- Users can login to a computer through network

Security mng. costs increase

- Hard to protect computers from malicious access completely

Masquerade detection

Conventional results

Researchers	Approaches	False Positive Rate	Hit Rate
Schonlau et al.	Uniqueness	1.4%	39.4%
	Bayes one-step Markov	6.7%	69.3%
	Hybrid multistep Markov	3.2%	49.3%
	Compression	5.0%	34.2%.
	Sequence Matching	3.7%	36.8%
	IPAM	2.7%	41.1%
Maxion and Townsend	Naïve Bayes (updating)	1.3%	61.5%
	Naïve Bayes (no updating)	4.6%	66.2%
Kim and Cha	SVM-based approach with voting	9.7%	80.1%
Oka et al.	ECM	2.5%	72.3%

Problems

- Conventional researches have attempted to improve the accuracy rate
- Users' behaviors would change with time

↓

Need to adapt to changes

	ECM
False Positive	2.5%
Hit Rate	72.3%
ROC Score	0.918
Training cost	1046.37 min.
Detection cost	22.13 sec.
CPU	Xeon 3.2GHz
Memory Size	4GB

Our strategy

- To borrow the same data
  - To compare results with conventional work
- To borrow ECM
  - Low false positive rate
  - High hit rate
  - High ROC score
- To exploit SVM
  - Low training cost
  - Adapt to changes of users' behaviors

Correlation of commands

time →

User1 :	*cd ls less ls less cd ls cd cd ls*
User2 :	*emacs gcc gdb emacs ls gcc gdb ls ls emacs*
User3 :	*mkdir cp cd ls cp ls cp cp cp cp*

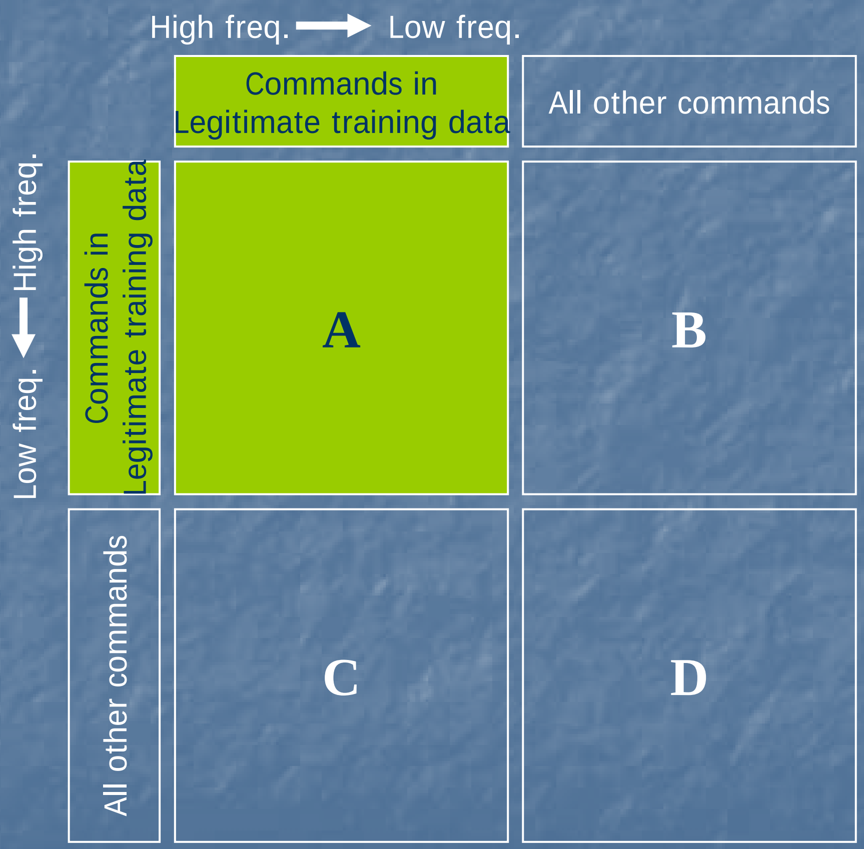
Strength of correlation of ls and *less* ： 2+1=3

Co-occurrence matrix

User1 :	*cd*	*ls*	*less*	*ls*	*less*	*cd*	*ls*	*cd*	*cd*	*ls*
User2 :	*emacs*	*gcc*	*gdb*	*emacs*	*ls*	*gcc*	*gdb*	*ls*	*ls*	*emacs*
User3 :	*mkdir*	*cp*	*cd*	*ls*	*cp*	*ls*	*cp*	*cp*	*cp*	*cp*

	cd	*ls*	*less*	*emacs*	*gcc*	*gdb*	*mkdir*	*cp*
cd	0	0	0	0	0	0	0	0
ls	0	3	0	3	1	1	0	0
less	0	0	0	0	0	0	0	0
emacs	0	4	0	1	3	3	0	0
gcc	0	4	0	2	1	3	0	0
gdb	0	5	0	2	1	1	0	0
mkdir	0	0	0	0	0	0	0	0
cp	0	0	0	0	0	0	0	0

Our co-occurrence matrix

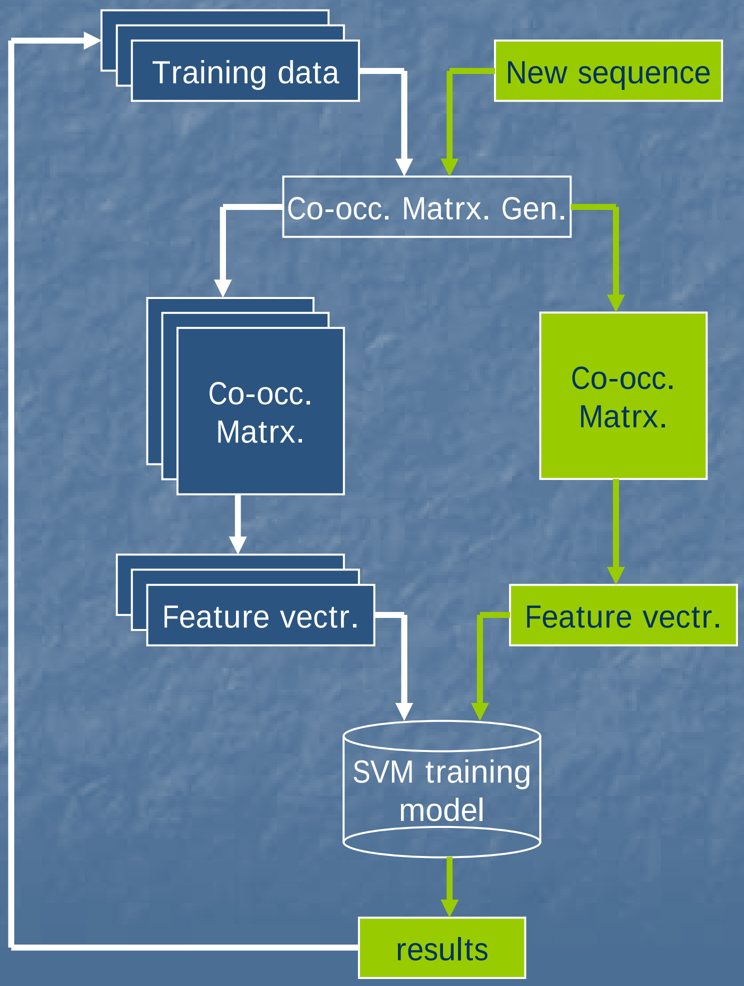


	cd	ls	less	emacs	gcc	gdb	mkdir	cp
cd	0	0	0	0	0	0	0	0
ls	0	3	0	3	1	1	0	0
less	0	0	0	0	0	0	0	0
emacs	0	4	0	1	3	3	0	0
gcc	0	4	0	2	1	3	0	0
gdb	0	5	0	2	1	1	0	0
mkdir	0	0	0	0	0	0	0	0
cp	0	0	0	0	0	0	0	0

	emacs	ls	gcc	gdb	cd	less	mkdir	cp
emacs	2	4	3	3	0	0	0	0
ls	3	3	1	1	0	0	0	0
gcc	2	4	1	3	0	0	0	0
gdb	2	5	1	1	0	0	0	0
cd	0	0	0	0	0	0	0	0
less	0	0	0	0	0	0	0	0
mkdir	0	0	0	0	0	0	0	0
cp	0	0	0	0	0	0	0	0

System overview

- Co-occ. Matrx. generation
- SVM feature vectr. generation
- SVM processing
- Results
- Refinement



Comparison with ECM

	ECM	Our method (based on 2-class SVM)
False Positive	2.5%	3.0%
Hit Rate	72.3%	72.74%
ROC Score	0.918	0.926
CPU	Xeon 3.2GHz	Pentium III 1.4GHz
Memory Size	4GB	512MB
Training cost	1046.37 min.	117.33 sec.
Detection cost	22.13 sec.	0.04 sec.

Comparison with ECM

	ECM	Our method (based on 2-class SVM)
False Positive	2.5%	3.0%
Hit Rate	72.3%	72.74%
ROC Score	0.918	0.926
CPU	Xeon 3.2GHz	Pentium III 1.4GHz
Memory Size	4GB	512MB
Training cost	1046.37 min.	117.33 sec.
Detection cost	22.13 sec.	0.04 sec.

Almost the same

Comparison with ECM

	ECM	Our method (based on 2-class SVM)
False Positive	2.5%	3.0%
Hit Rate	72.3%	72.74%
ROC Score	0.918	0.926
CPU	Xeon 3.2GHz	Pentium III 1.4GHz
Memory Size	4GB	512MB
Training cost	1046.37 min.	117.33 sec.
Detection cost	22.13 sec.	0.04 sec.

With lower power machine

Comparison with ECM

	ECM	Our method (based on 2-class SVM)
False Positive	2.5%	3.0%
Hit Rate	72.3%	72.74%
ROC Score	0.918	0.926
CPU	Xeon 3.2GHz	Pentium III 1.4GHz
Memory Size	4GB	512MB
Training cost	1046.37 min.	117.33 sec.
Detection cost	22.13 sec.	0.04 sec.

Smaller

Comparison with ECM

	ECM	Our method
False P		
Hit R		
ROC S		
CP		Hz
Memory Size	4GB	512MB
Training cost	1046.37 min.	117.33 sec.
Detection cost	22.13 sec.	0.04 sec.

With lower power machine

Training cost:535times smaller
Detection cost:553times smaller
Achieved almost the same good charac.

Online update

- **To run the system a.s.a.p. even if we don't have enough amount of data for training**
- **To adapt changes of users' behaviors**

•Our proposal is with low comput. cost
•Online update of training model
•By modifying application of the data

2-class and 1-class based methods

- 2-class vs. 1-class
  - Data: 2-class > 1-class
  - Cost: 2-class > 1-class
  - Accuracy: 2-class > 1-class
- We look them concretely by experiments

Update Under 2-class SVM

# trained commands	20 blks. (10000)	30 blks. (15000)	40 blks. (20000)	50 blks. (25000)
False Positive	8%	6%	5%	3%
Hit Rate	68%	69%	68%	72.74%
ROC Score	0.89	0.90	0.91	0.93
Update costs	43.86 s	59.53 s	89.65 s	107.30 s
SVM training costs	3.36 s	7.04 s	6.90 s	10.03 s
Detection cost	0.04 s			

Update Under 2-class SVM

# trained commands	20 blks. (10000)	30 blks. (15000)	40 blks. (20000)	50 blks. (25000)
False Positive	8%	6%	5%	3%
Hit Rate	68%	69%	68%	72.74%
ROC Score	0.89	0.90	0.91	0.93
Update costs	43.86 s	59.53 s	89.65 s	107.30 s
SVM training costs	3.36 s	7.04 s	6.90 s	10.03 s
Detection cost	0.04 s			

improved

Update Under 2-class SVM

# trained commands	20 blks. (10000)	30 blks. (15000)	40 blks. (20000)	50 blks. (25000)
False Positive	8%	6%	5%	3%
Hit Rate	68%	69%	68%	72.74%
ROC Score	0.89	0.90	0.91	0.93
Update costs	43.86 s	59.53 s	89.65 s	107.30 s
SVM training costs	3.36 s	7.04 s	6.90 s	10.03 s
Detection cost	0.04 s			

improved

improved

Update Under 2-class SVM

# trained commands	20 blks. (10000)	30 blks. (15000)	40 blks. (20000)	50 blks. (25000)
False Positive	8%	6%	5%	3%
Hit Rate	68%	69%	68%	72.74%
ROC Score	0.89	0.90	0.91	0.93
Update costs	43.86 s	59.53 s	89.65 s	107.30 s
SVM training costs	3.36 s	7.04 s	6.90 s	10.03 s
Detection cost	0.04 s			

improved

improved

improved

Update Under 1-class SVM

# trained commands	20 blks. (2000)	30 blks. (3000)	40 blks. (4000)	50 blks. (5000)
False Positive	12%	8%	7%	6%
Hit Rate	68%	64%	61%	62.77%
ROC Score	0.85	0.86	0.87	0.88
Update costs	0.88 s	1.53 s	1.79 s	2.15 s
SVM training costs	0.17 s	0.18 s	0.22 s	0.27 s
Detection cost	0.04 s			

Update Under 1-class SVM

# trained commands	20 blks. (2000)	30 blks. (3000)	40 blks. (4000)	50 blks. (5000)
False Positive	12%	8%	7%	6%
Hit Rate	68%	64%	61%	62.77%
ROC Score	0.85	0.86	0.87	0.88
Update costs	0.88 s	1.53 s	1.79 s	2.15 s
SVM training costs	0.17 s	0.18 s	0.22 s	0.27 s
Detection cost	0.04 s			

improved

Update Under 1-class SVM

# trained commands	20 blks. (2000)	30 blks. (3000)	40 blks. (4000)	50 blks. (5000)
False Positive	12%	8%	7%	6%
Hit Rate	68%	64%	61%	62.77%
ROC Score	0.85	0.86	0.87	0.88
Update costs	0.88 s	1.53 s	1.79 s	2.15 s
SVM **training costs**	0.17 s	0.18 s	0.22 s	0.27 s
Detection cost	0.04 s			

improved

improved

Results: 2-class vs. 1-class

2-class

# trained commands	20 blks. (10000)	30 blks. (15000)	40 blks. (20000)	50 blks. (25000)
False Positive	8%	6%	5%	3%
Hit Rate	68%	69%	68%	72.74%
ROC Score	0.89	0.90	0.91	0.93
Update costs	43.86 s	59.53 s	89.65 s	107.30 s
SVM training costs	3.36 s	7.04 s	6.90 s	10.03 s
Detection cost	0.04 s			

1-class

# trained commands	20 blks. (2000)	30 blks. (3000)	40 blks. (4000)	50 blks. (5000)
False Positive	12%	8%	7%	6%
Hit Rate	68%	64%	61%	62.77 %
ROC Score	0.85	0.86	0.87	0.88
Update costs	0.88 s	1.53 s	1.79 s	2.15 s
SVM training costs	0.17 s	0.18 s	0.22 s	0.27 s
Detection cost	0.04 s			

Conclusion

- Results
  - Extension of ECM with low computing costs
  - Availability with online update
- Future work
  - To do more experiments with other data
  - To improve accuracy by integrating several methods
  - To test and extend our proposal to other applications like databases (SQL injections)

Thank you