Big Data in Cyber Security

Latifur Khan, PhD

Professor
Department of Computer Science, School of Engineering
The University of Texas at Dallas, USA
www.utdallas.eu/~lkhan

Agenda

- Detecting Cheats of Computer Game
  - Funded by NSF, AFOSR
- Website Fingerprinting
  - Funded by NSF, AFOSR
- Insider Threat Detection
  - Funded by NSF, AFOSR
- Secure Data Analytics
  - Funded by NSF, AFOSR
- Real Time Anomaly Detection
  - Funded by Sandia via DOE

Highlights: Publication

- Detecting Cheats of Computer Game
  - Published in IEEE Transactions Journal, TDSC, IEEE Big Data Conference
- Website Fingerprinting
  - Published in ACSAC conference
- Insider Threat Detection
  - Best Paper Award from ICTAI conference, Patent
- Secure Data Analytics
  - Published in ACM CCS, ASIACCS, ESORIC conference
- Real Time Anomaly Detection
  - Published in IEEE BigData conference

THE UNIVERSITY OF TEXAS AT DALLAS

GCI: A GPU Based Transfer Learning Approach for Detecting Cheats of Computer Game

Md Shihabul Islam, Bo Dong, Swarup Chandra,

Faculty: Latifur Khan, PhD

This material is based upon work supported by

Outline

- ➔ Intro to Cheating in Video Games
- ➔ Motivations & Challenges
- ➔ Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework Details
- ➔ Empirical Evaluation
- ➔ Future Works

Video Game Industry

- One of the largest Entertainment Industries
- Global revenue is expected to reach nearly $180 billion in 2020 [1]
- Most revenue comes from
  - In-game purchases
  - Advertisements
  - Consoles and controllers

- A serious impediment damaging this multi-billion dollar industry: **Cheating**

[1] https://www.marketwatch.com/story/videogames-are-a-bigger-industry-than-sports-and-movies-combined-thanks-to-the-pandemic-11608654990

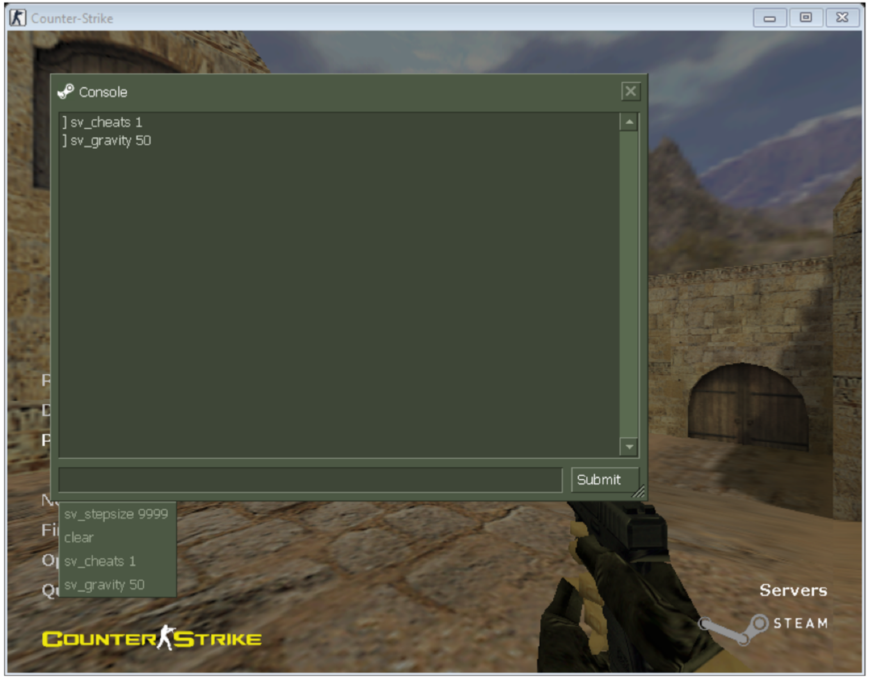
What is Cheating in Video Games?

- Any behavior performed by a game player to change normal execution of game-play and obtain unfair advantages while playing video games

- The game player who cheats is called a <u>Cheater</u>

How Gamers Cheat: Techniques

- Using Cheat Code
- Modifying Game Code
- Modifying System Software
- Modifying Game Traffic
- Using Game Bots



Example of using cheat codes in Counter-Strike game

How Gamers Cheat: Resources

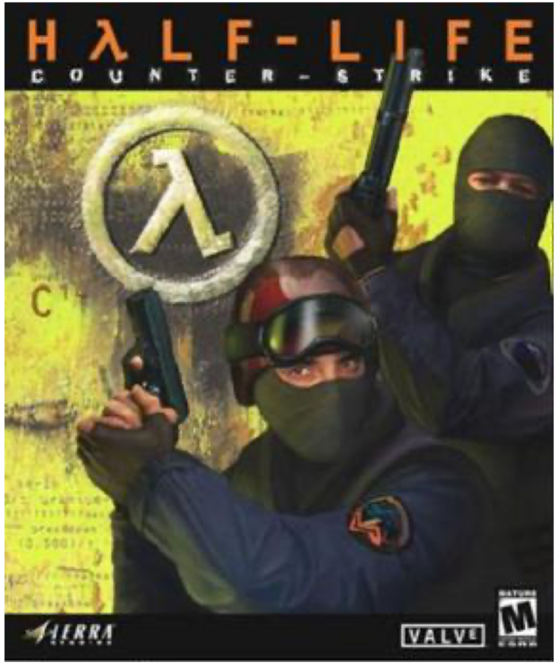
- Gaming community
- Social media [2]
    - Discord
    - Instagram

[2] https://egmnow.com/the-human-side-of-those-who-cheat-at-and-hack-games/

Online Video Game Example

- Counter-Strike 1.6 ४
  - Multiplayer first-person shooting game
  - One of the most popular online games
  - Many cheats available

- Some cheats
  - Wall-hack
  - Speed-hack
  - Aim-bot
  - Trigger-bot
  - Artificial-lag



४https://www.valvesoftware.com/en/

Why Gamers Cheat?

Profit

Competitiveness

Entertainment

Damages of Cheating

- Adversely affects game's popularity and reputation
  - 77% of players are likely to stop playing online multiplayer games if they suspect other players are cheating [3]
  - 60% have had negative gaming experiences because of cheating [3]

- Hurts revenue
  - 48% players would be reluctant to purchase any in-game content if other players are cheating. [3]

source: vecteezy.com

[3] https://resources.irdeto.com/irdeto-global-gaming-survey/infographic-cheating-game-over

Outline

- ➔ Intro to Cheating in Video Games
- ➔ Motivations, Challenges, and Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework
- ➔ Empirical Evaluation
- ➔ Future Works

Motivation

- Resist cheating trend in online games
- Limited client-side information
    - Detecting cheats is challenging mainly due to the limited client-side information.
- Complexity
    - The cheating techniques are unknown and complex.

Challenges

- Game dependent:
  - Most cheat detection methods analyze decrypted game-dependent data.
- Covariate shift:
  - The assumption of training set and test set having similar distribution may not hold.
  - This may be due to sampling bias caused by label scarcity, inaccessibility, and the cost of label procurement.
- Limited labeled data:
  - Supervised learning models such as SVM, kNN, and neural network typically perform well when training and test datasets have similar distribution.
  - Supervised learning mechanism not suitable for very limited training data.
- Computational efficiency:
  - Current cheat detection methods mainly have delayed detection.
  - A large delay in detection (e.g., using game logs etc.) may not be effective to act upon cheaters at the right time.

Contribution

- Game independence:
  - In this work, we analyze the game traffic, which is encrypted and game independent.
  - It is easier to evaluate over encrypted traffic since most games are not open-source.
- Covariate shift:
  - We utilize relative density ratio to estimate importance weights associated with training data instances.
- Scalability:
  - For server-side cheat detection deployment, we demonstrate the scalability of our proposed approach using Apache Spark and Graphics Processing Unit (GPU).

Outline

➔ Intro to Cheating in Video Games
➔ Motivations, Challenges, and Our Contribution
➔ A Brief Overview of Machine Learning
➔ Proposed Framework
➔ Empirical Evaluation
➔ Future Works

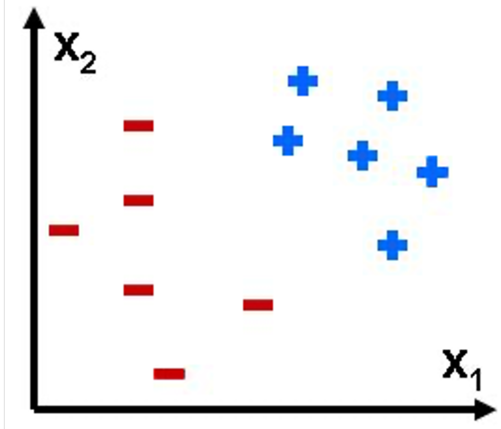
What is Data Classification

- Classification problem can be described as:

  Given a training data TD = $\{(x_1, y_1), (x_2, y_2), \ldots.., (x_N, y_N)\}$.
  Design a function f: $X \rightarrow Y$, that maps any observed data x to a certain class y.

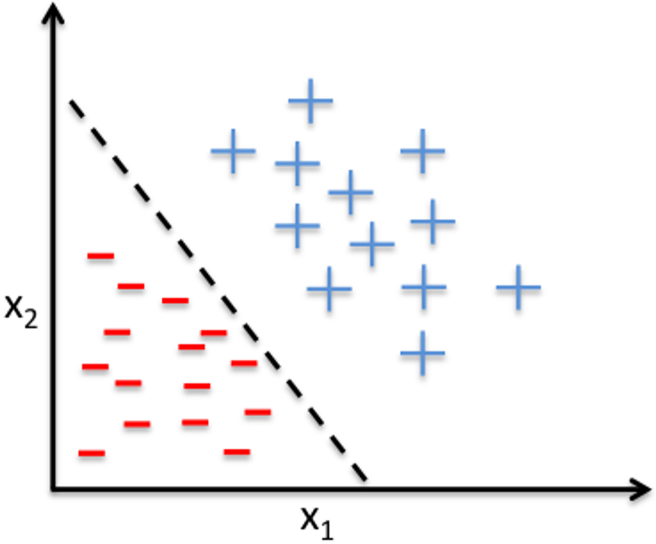
Binary Classification

- Is a classification problem, where we have two classes (we often call one class positive and the other negative)



https://alliance.seas.upenn.edu/~cis520/dynamic/2017/wiki/index.php?n=Lectures.Classification

Binary Classification (linearly separable data)



http://sebastianraschka.com/Articles/2015_singlelayer_neurons.html

Binary Classification (linearly separable data)

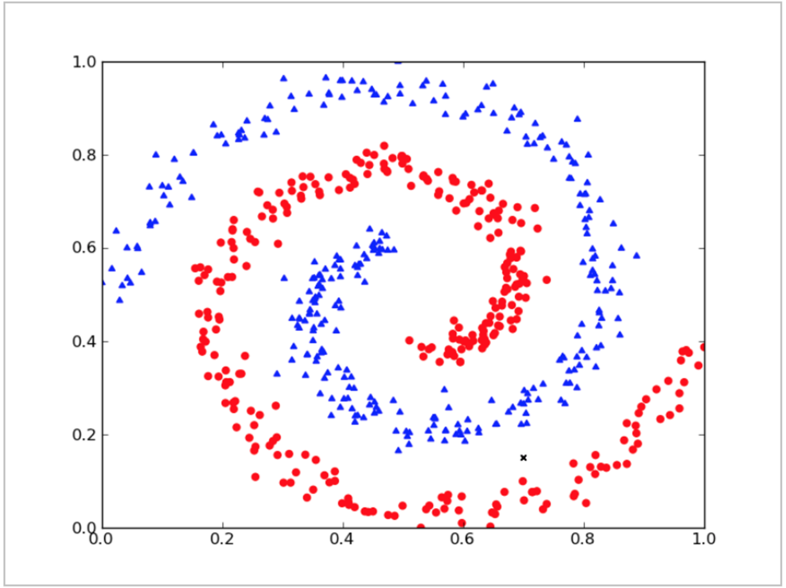
- Our goal is to find a hyperplane such that

  $Y^{i} = sign(w^{T}x^{i} + b)$, for all $(x^{i}, y^{i}) \in$ Training data

- We predict the class y' of data item x' as

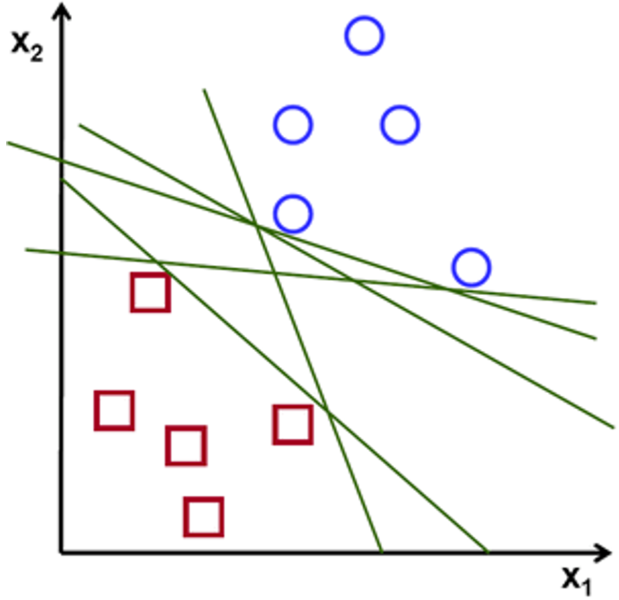
  $Y' = sign(w^{T}x' + b)$

Binary Classification (linearly inseparable data)



https://www.classes.cs.uchicago.edu/archive/2013/winter/12200-1/assignments/pa4/index.html

What is the best linear Separator?



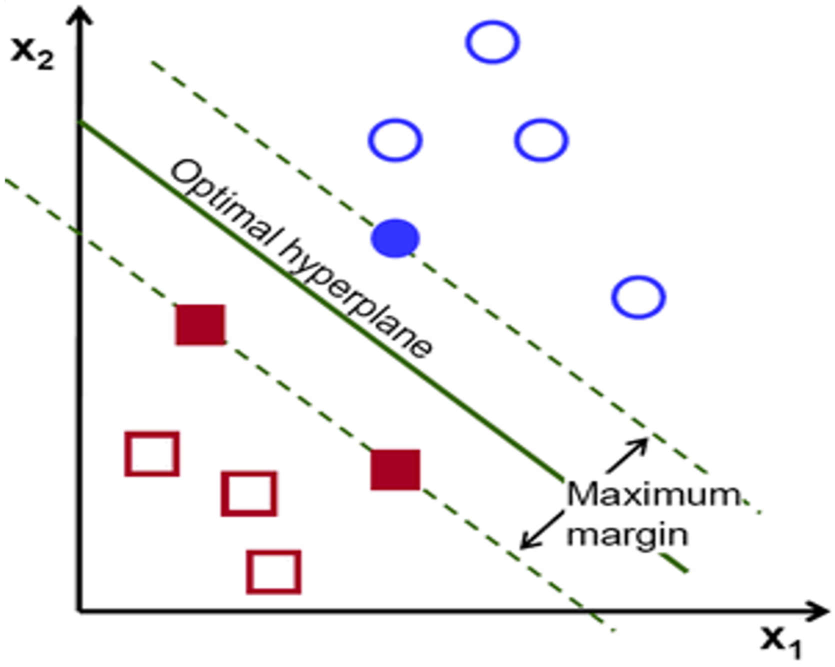
https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html

Support vector machines (SVMs)

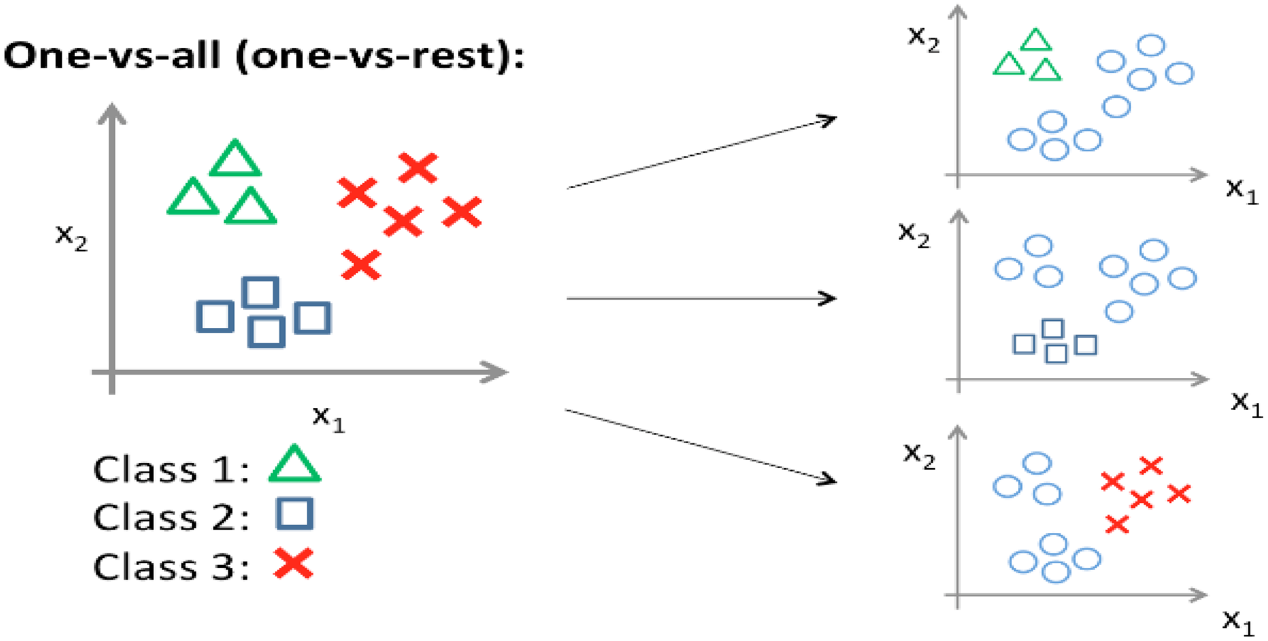
Define the margin to be the twice the distance of the closest data point to the classifier

SVM chooses the classifier (hyperplane) that maximize the margin: Good according to intuition, theory, practice.

SVMs



SVMs- Multi-class classification



https://www.linkedin.com/pulse/multi-class-classification-imbalanced-data-using-random-burak-ozen/

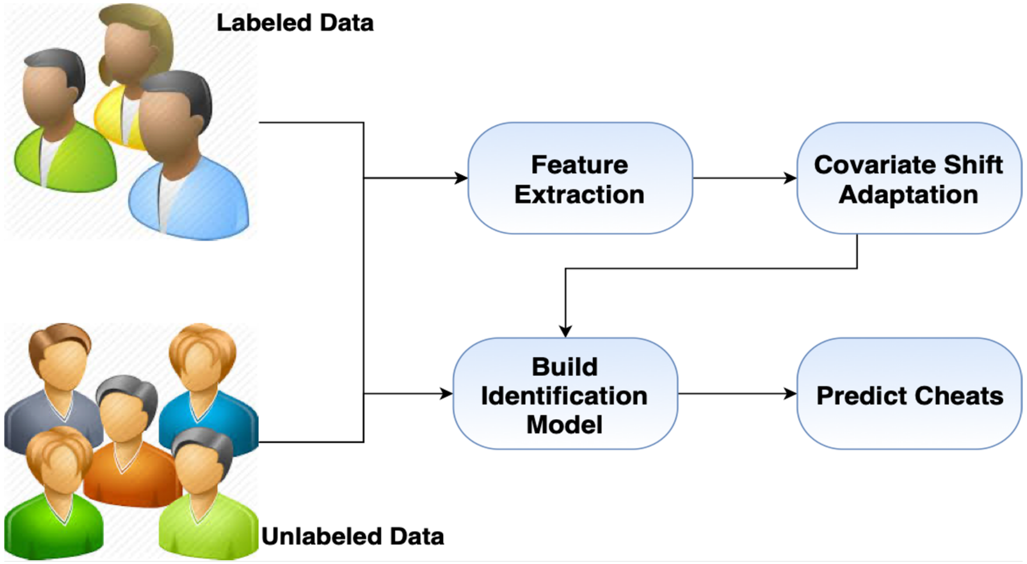
SVMs- one-vs-one

Alternatively we can construct a classifier for all possible pairs of labels.

Given a new data point, we can classify it by majority vote.

Outline

Overview of GCI framework



Feature Extraction

- Packets are encrypted.
- Extract features from packet headers.

- Some general features:
  - Number of incoming packets.
  - Number of outgoing packets.
  - Sum of incoming packet sizes.
  - Sum of outgoing packet sizes.

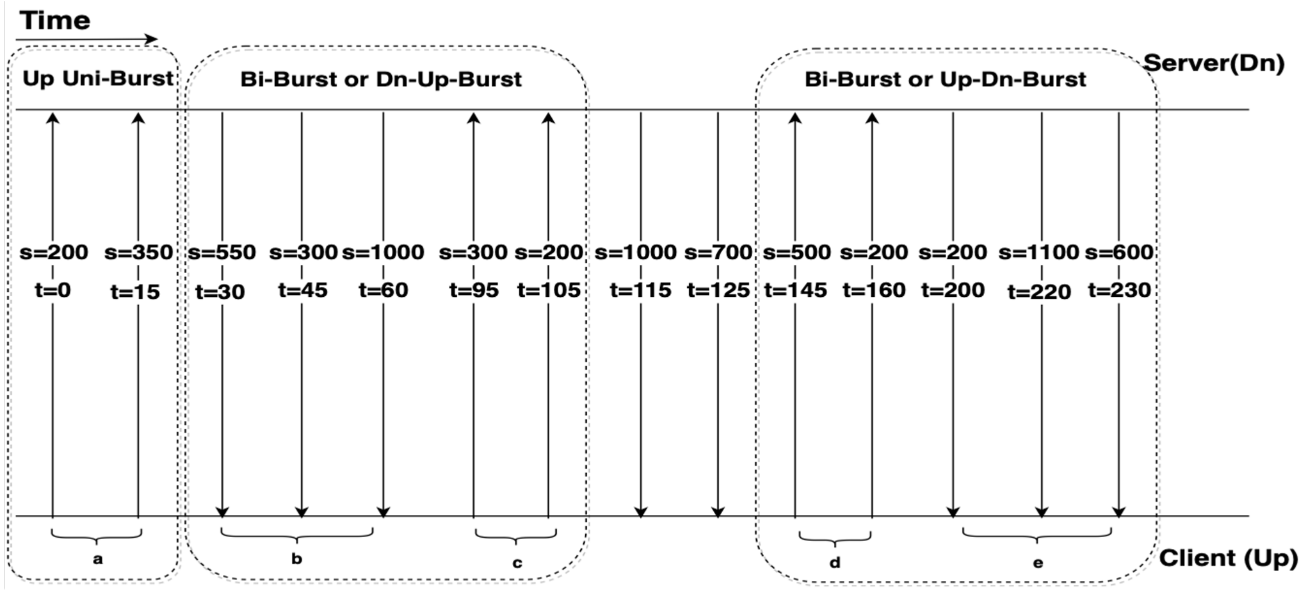
Feature Extraction: BIND

- **BIND** (Fingerprinting with BI-directioNal Dependence)[4] [5]:
  - Works with <u>Bursts</u>
  - A burst is a sequence of consecutive packets transmitted along the same direction
  - Uni-Burst:
    - Size
    - Time
    - Direction
    - Number of packets in the burst
  - Bi-Burst:
    - Size
    - Time
    - Number of packets in the burst

[4] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham, "Adaptive encrypted traffic fingerprinting with bi-directional dependence," in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC '16. Los Angeles, California, USA, 2016, pp. 177–188.

[5] Al-Naami, K., El Ghamry, A., Islam, M.S., Khan, L., Thuraisingham, B.M., Hamlen, K.W., Alrahmawy, M. and Rashad, M., 2019. Bimorphing: A bi-directional bursting defense against website fingerprinting attacks. *IEEE Transactions on Dependable and Secure Computing*.

Feature Extraction: BIND



DnUp-1850-500

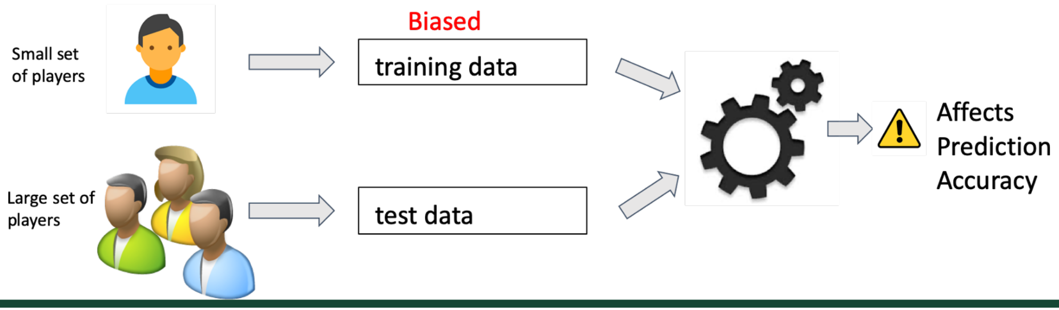
UpDn-700-1900

An example of feature extraction procedure following BIND

Covariate Shift Problem

- What if we do not find a good training set?
- Different sets of players may cause biased training data with respect to test data.

Example Scenario



- **Solution:**
  - We utilize relative density ratio to estimate importance weights associated with training data instances.
  - We propose a expectation-maximization technique to automatically learn model parameters for relative density ratio estimation from available data.

Covariate Shift Adaptation: Spark Implementation

- Scalability:
  - As our proposed work contains a great deal of large-scale matrix multiplications, we utilize Spark to accelerate the process.
  - We separate the large matrix computation into small blocks and distribute the small tasks parallel on Spark clusters.

- Computation efficiency:
  - Applying Spark reduces the execution time and improves performance when we have large data set.

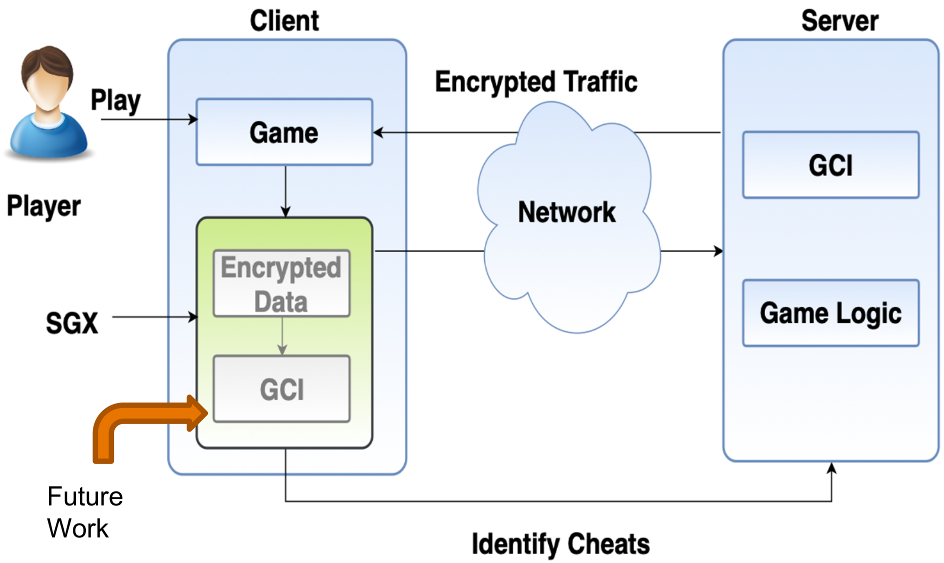
Covariate Shift Adaptation: GPU Implementation

- Graphics Processing Units (GPU)
  - Powerful parallel processing capability with abundant computing cores
  - High memory bandwidth
  - Reduces processing burden from the CPU

- We use GPU to accelerate major time-consuming operations
  - Learning parameters for relative density ratio
  - Hyper-parameters searching for the estimator.

Deployment

- ❑ Deploy GCI framework in game server-side
- ❑ Since our mechanism is not game-specific, we can deploy cheat detection on the client-side as well. **(Future Work)**
- ❑ We plan to deploy our GCI framework in SGX [6] in game client-side for future work.



[6] V. Costan and S. Devadas, “Intel sgx explained.” IACR Cryptology ePrint Archive, vol. 2016, no. 086, pp. 1–118, 2016.

Outline

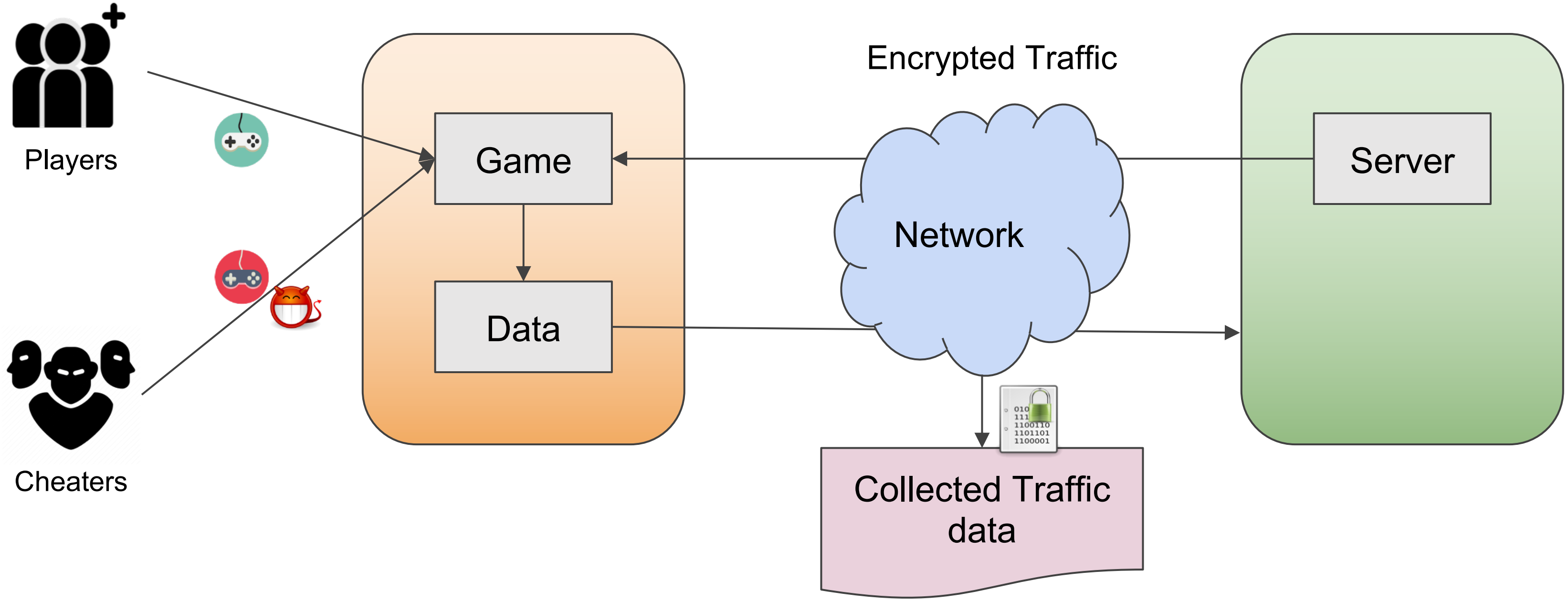
- ➔ Intro to Cheating in Video Games
- ➔ Motivations, Challenges, and Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework
- ➔ Empirical Evaluation
- ➔ Future Works

Empirical Evaluation: Data Collection

- We collect game traffic with help of students from class CS 6301: Cyber Security Essentials of University of Texas at Dallas and Big Data Analytics and Management Lab.
- In total 20 students participate to collect data.
- Students install in their personal machines the game Counter-Strike 1.6 and the three selected cheat types downloaded from a diverse community of popular cheating sources.[1,2]
- They connect to the server and play the game in both normal game mode as well as using the cheats applied to the game.

[1] https://www.gamespot.com/counter-strike/cheats/
[2] https://www.unknowncheats.me/forum/index.php

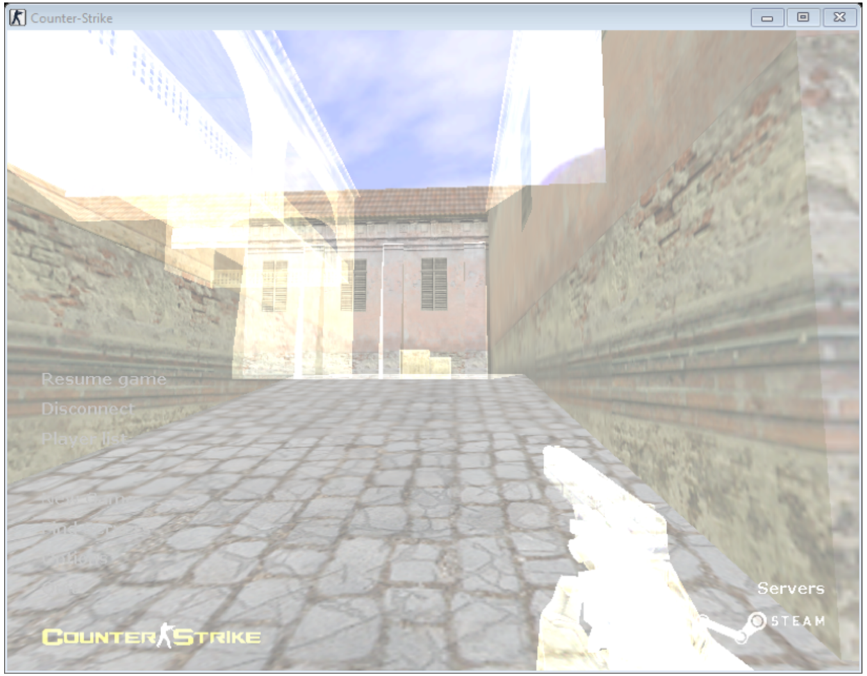
Empirical Evaluation: Data Collection

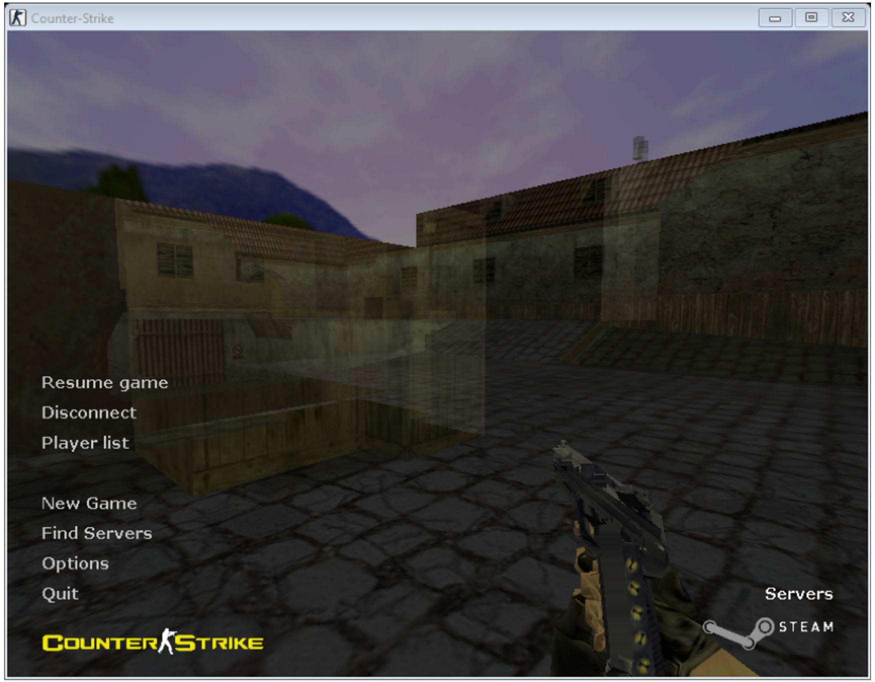


Empirical Evaluation: Counter-strike Cheats

- Aim-bot
  - Enables automatic targeting the opponent while shooting.
  - This targeting works even if the opponent is too far away or behind walls.
- Speed-hack
  - Enables speed increase in player’s movement while playing the game.
  - A player can apply different variations of speeds and play the game.
- Wall-hack
  - Makes the walls transparent for the player so that he or she can see the enemy through the walls.

Wall-hack Example





Empirical Evaluation: Experiment Settings

- Feature extraction:
  - We first extract features following [4][5]

- Generate training and test data:
  - We generate data in different 10 groups by selecting different fixed sized training set and run experiment by cross-validation.

[4] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham, “Adaptive encrypted traffic fingerprinting with bi-directional dependence,” in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC ’16. Los Angeles, California, USA, 2016, pp. 177–188.

Empirical Evaluation: Experiment Settings

- Multi class labels:
  - Aim-bot
  - Speed-hack
  - Wall-hack
  - Normal (without cheats)

- Binary class labels:
  - Cheats (aim-bot, speed-hack, wall-hack)
  - Normal (without cheats)

Empirical Evaluation: Baseline Methods

Baseline Methods	Description
KMSVM	Equip KMM[7] with base classifier weighted SVM to build classification models.
KLISVM	Equip KLIEP[8] with base classifier weighted SVM to build classification models.
SVM	Multi class Support Vector Machine.
Proposed Method	**Description**
GCI	Equip revised RULSIF with base classifier weighted SVM to build classification models

[7] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Scholkopf, "Correcting sample selection bias by unlabeled data," in Proceedings of the 19th International Conference on Neural Information Processing Systems, ser. NIPS'06. Cambridge, MA, USA: MIT Press, 2006, pp. 601–608.
[8] Y. Kawahara and M. Sugiyama, "Sequential change-point detection based on direct density-ratio estimation," Stat. Anal. Data Min., vol. 5, no. 2, pp. 114–127, Apr. 2012

Empirical Evaluation: Performance

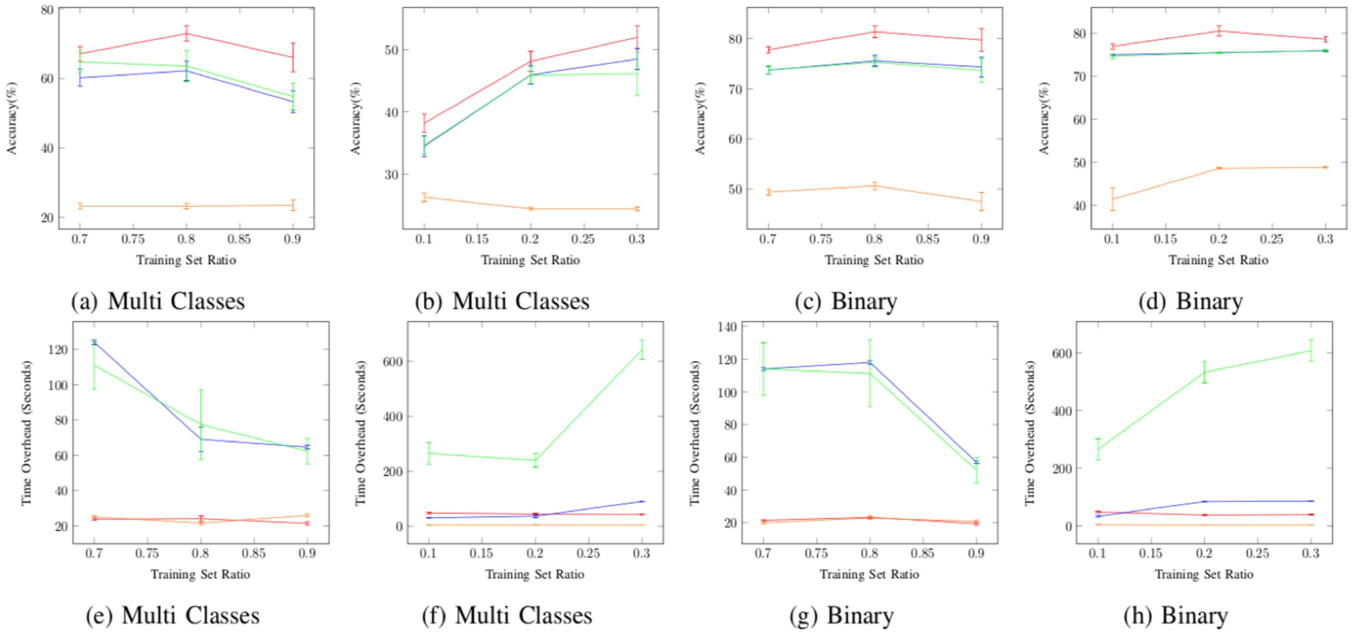


Figure 5: Performance of classification for all approaches. (GCI; KMSVM; KLISVM; SVM).

Empirical Evaluation:Performance

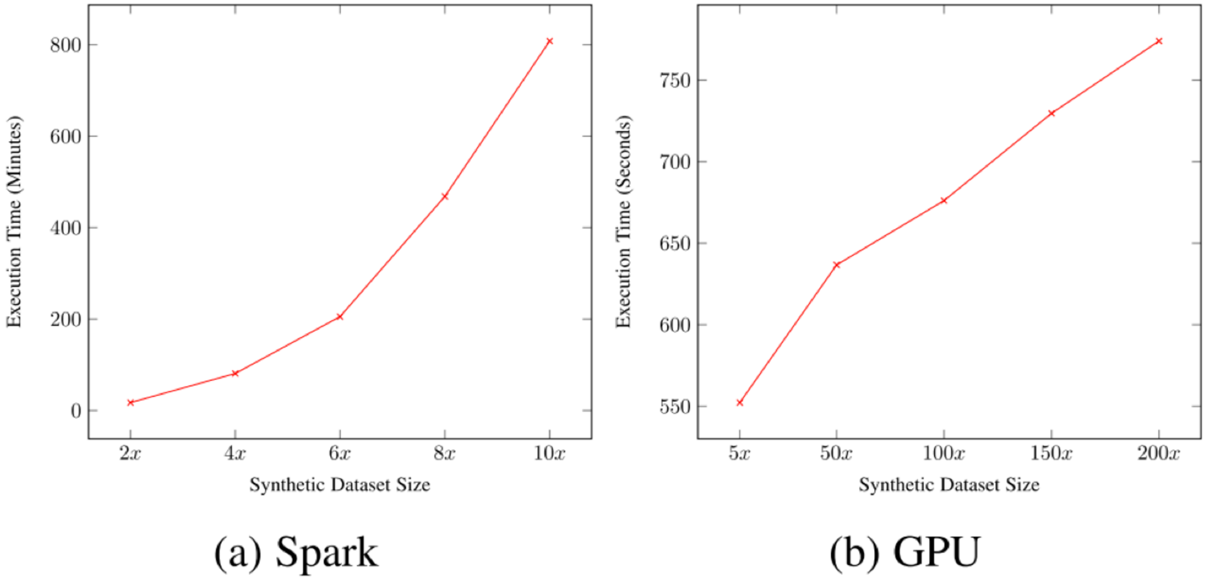


Fig. 6. Performance of Spark and GPU for large datasets.

Outline

Future Direction

- We plan to investigate the performance of GCI when more cheating techniques are introduced.
- We will consider other games and examine how GCI performs.
- We plan to perform secure execution of cheat detection at the client-side with Trusted Execution Environments such as Intel SGX platform.
- We will explore similar detection methods for distributed massive online games, i.e., those which do not have a server-client architecture.

Agenda

- Detecting Cheats of Computer Game
  - Funded by NSF, AFOSR
- Website Fingerprinting ←=================
  - Funded by NSF, AFOSR
- Insider Threat Detection
  - Funded by NSF, AFOSR
- Secure Data Analytics
  - Funded by NSF, AFOSR
- Real Time Anomaly Detection
  - Funded by Sandia via DOE

Adaptive Encrypted Traffic Fingerprinting With Bidirectional Dependence*

K. Al-Naami, G. Ayoade, A. Siddiqui, N. Ruozzi, L. Khan and B. Thuraisingham, "P2V: Effective Website Fingerprinting Using Vector Space Representations," Computational Intelligence, 2015 IEEE Symposium Series on, Cape Town, 2015, pp. 59-66.

K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham. 2016. Adaptive encrypted traffic fingerprinting with bi-directional dependence. In Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC '16), Los Angeles, CA.

* This work is funded by NSF, AFOSR, and NSA.

Website Fingerprinting (WFP)

Website Fingerprinting (WFP) is a Traffic Analysis (TA) attack that threatens web navigation privacy.

WFP allows attackers to learn information about a website accessed by the user, by recognizing patterns in traffic.

Victims and Attackers:

- Individuals, businesses and governments.

Website Fingerprinting

The Goal is to identify the websites

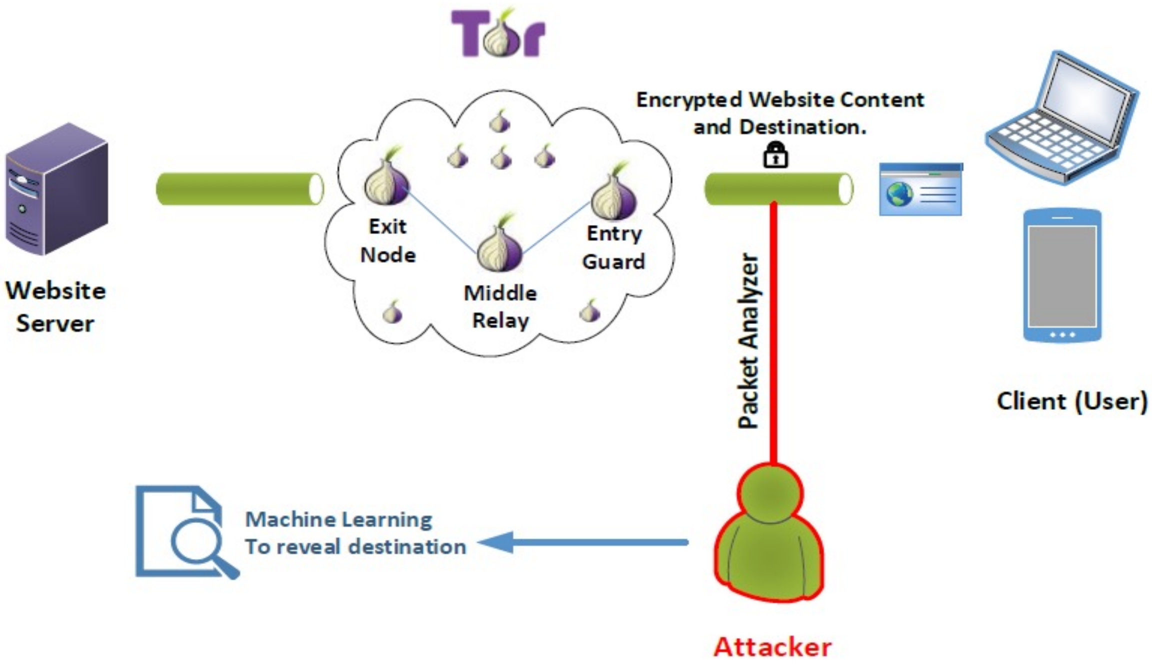
Can harm certain individuals

- Journalists
- Activists
- Bloggers

Can also help identify threats

- Bad people

WFP Diagram – Tor



Challenge

Data Not encrypted

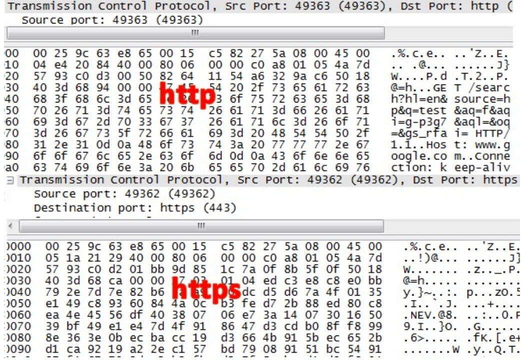
- Solution is easy

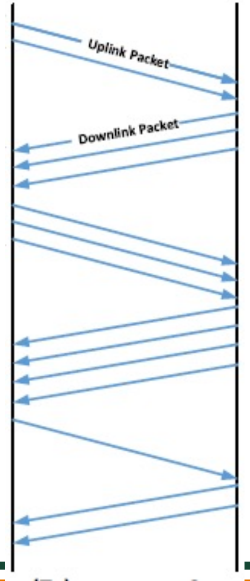
However

- Data is encrypted

All we can see is just:

- packet size in bytes
- packet time





Contributions

A novel multi-domain coarse-feature extraction approach (*BIND*) (fingerprinting with BI-directioNal Dependence) over encrypted data

- considers the relationship among sequences of packets in opposite directions

Across multiple domains

- HTTPS
- Tor
- Apps

Website Fingerprinting

App Fingerprinting

Closed-world and open-world settings

The approach is more immune and resilient to known defenses

- Adaptive Nature

Attackers and Defenders – Arm Race

The competition between WFP attackers and defenders is continually evolving

Attackers collect the packets and apply ML.

Defenders morph packets (website A to look like website B)

The coarser the features, the more resistant

BIND: coarse-feature approach

Summary of previous and proposed approaches

Data Analysis Method	Setting Type	Features	Classifier
VNG++ [17]	Closed	Uni-Burst Size & Count Total Trace Time Uplink/Downlink Bytes	Naïve Bayes
P [28]	Closed	Uni-Burst Size & Count Packet Size Packet Ordering	SVM
OSAD [37]	Closed	Cell Traces	Optimized SVM
BINDSVM *	Closed	BIND features: Bi-Burst Size & Time Uni-Burst Size, Time, & Count Packet Size	SVM
WKNN [36]	Open	Same features as P	Weighted k-NN
BINDWKNN *	Open	BIND features: Same features as BINDSVM	Weighted k-NN
BINDRF *	Open	BIND features: Same features as BINDSVM	Random Forest

*new approaches introduced in this paper

Adaptive Learning

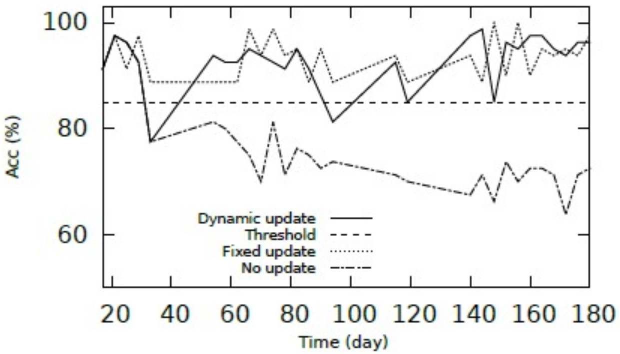


Figure 9: Adaptive Learning.

Agenda

- ❑ Detecting Cheats of Computer Game
  - ❑ Funded by NSF, AFOSR
- ❑ Website Fingerprinting
  - ❑ Funded by NSF, AFOSR
- ❑ Insider Threat Detection 🡐=================
  - ❑ Funded by NSF, AFOSR
- ❑ Secure Data Analytics
  - ❑ Funded by NSF, AFOSR
- ❑ Real Time Anomaly Detection
  - ❑ Funded by Sandia via DOE

Evolving Insider Threat Detection using Stream Analytics and Big Data

Funded by:

What is the Problem?
Definition of an Insider

An **Insider** is someone who exploits, or has the intention to exploit, his/her **legitimate access** to assets for unauthorised purposes.

For example, over time, legitimate users may enter commands that read or write private data, or install malicious software

Challenges/Issues

Reduce false alarm rate without sacrificing threat detection rate

Threat detection is challenging since insiders mask and adapt their behavior to resemble legitimate system.

Different Data Types:

- Sequence Data
- Non Sequence Data

Sequence Data

Movement Pattern
(student center)(office)(ml)
(maqs ave)(ml)(tang)(ml)(sloan)(ml)
(100 memorial)(ml)(tang)(black sheep restaurant)(ml)(sloan)(ml)
(off phm)(ml)(starbucks)(ml)
(hamshire &broadway)(off phm)(ml)(starbucks)(ml)
(ml)(100 memorial)(ml)(tang)(black sheep restaurant)(ml)(sloan)(ml)

Movement Pattern
(student center)(office)
(student center)(office)(ml)
(student center)(pbe)(ml)(office)(ml)
(black sheep restaurant)(ml)(sloan)(ml)(maqs ave)(ml)
(student center)(ml)(office)(ml)
(ml)(black sheep restaurant)(ml)(sloan)(ml)(maqs ave)(ml)

Nathan Eagle, Alex (Sandy) Pentland, "Reality mining: sensing complex social systems," Journal Personal and Ubiquitous Computing Volume 10 Issue 4, March 2006 Pages 255 - 268

Graph Based Representation

header,150,2, execve(2),,Fri Jul 31 07:46:33 1998, +

652468777 msec

path,/usr/lib/fs/ufs/quota

attribute,104555,root,bin,8388614,187986,0

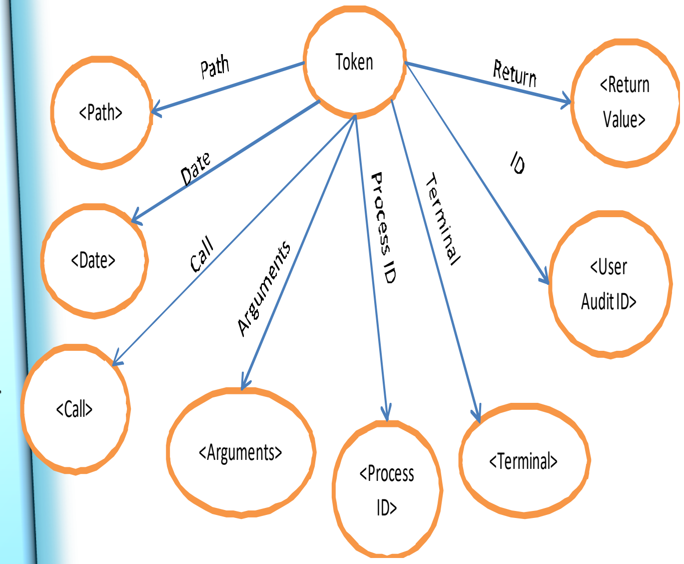
exec_args,1,

/usr/sbin/quota

subject,2110,root,rjm,2110,rjm,280,272,0-0-172.16.112.50

return,success,0

trailer,150



Contribution

Anomaly

Sequence Data

(student center)(office)(ml)
(maqs ave)(ml)(tang)(ml)(sloan)(ml)
(100 memorial)(ml)(tang)(black sheep restaurant)(ml)(sloan)(ml)
(off phm)(ml)(starbucks)(ml)
(hamshire &broadway)(off phm)(ml)(starbucks)(ml)
(ml)(100 memorial)(ml)(tang)(black sheep restaurant)(ml)(sloan)(ml)

Unsupervised Learning (Quantized Dictionary)

MAP Reduce + Hadoop

Stream Analytic & Ensemble

Anomal
y

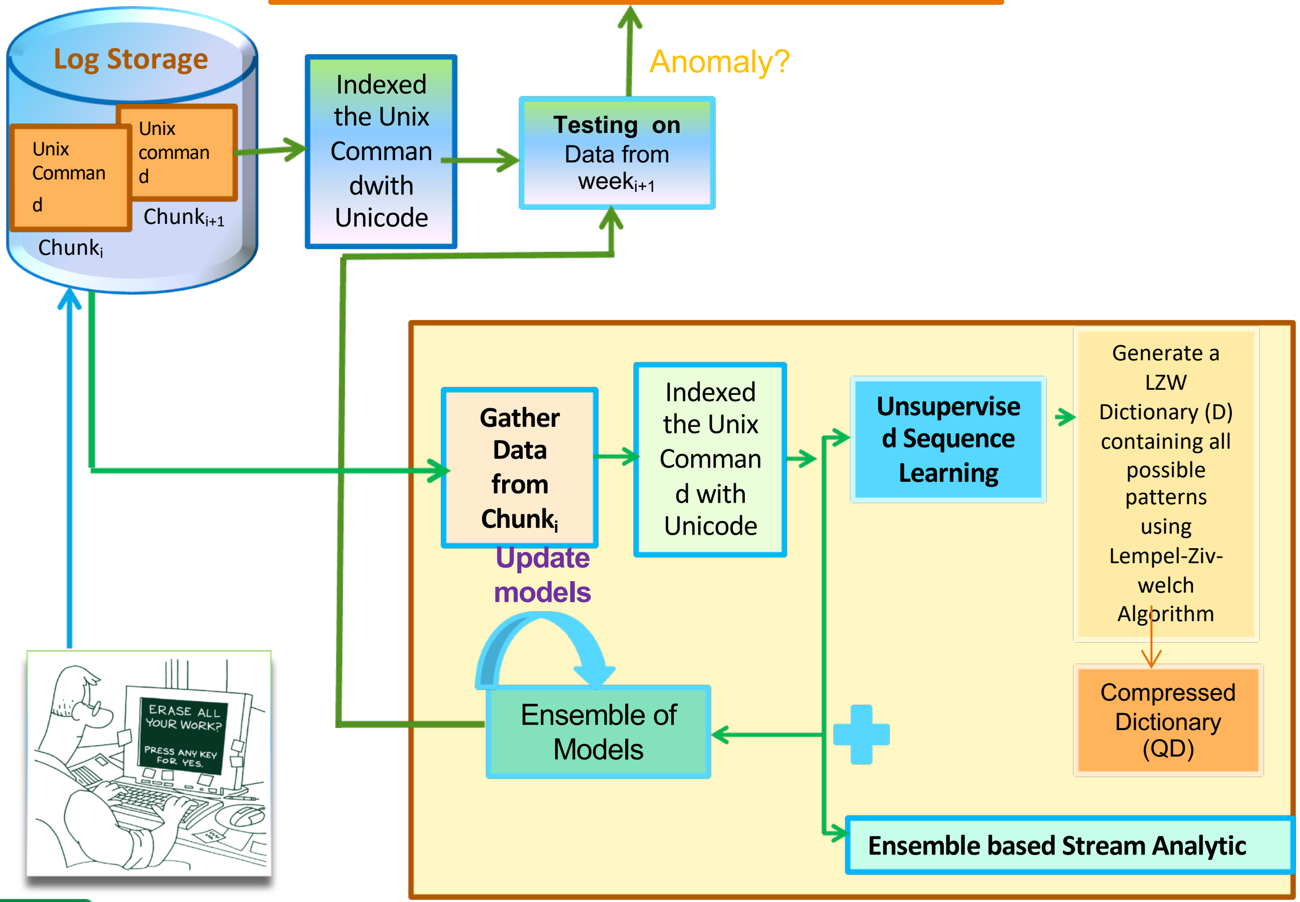
Non Sequence Data
[10,20,5,50]

Unsupervised Learning (Graph-Based)

Stream Analytic & Ensemble

UTD

Architecture



Proposed Method

Unsupervised Sequence Learning

+

Ensemble based Stream Analytics

Unsupervised Sequence Learning

- **Normal users** have a **repetitive sequence of commands**, system calls etc..
- A sudden deviation from normal behavior, raises an alarm indicating an insider threat

- **To find an insider threat**

We need to collect these repeated sequences of commands in an unsupervised fashion

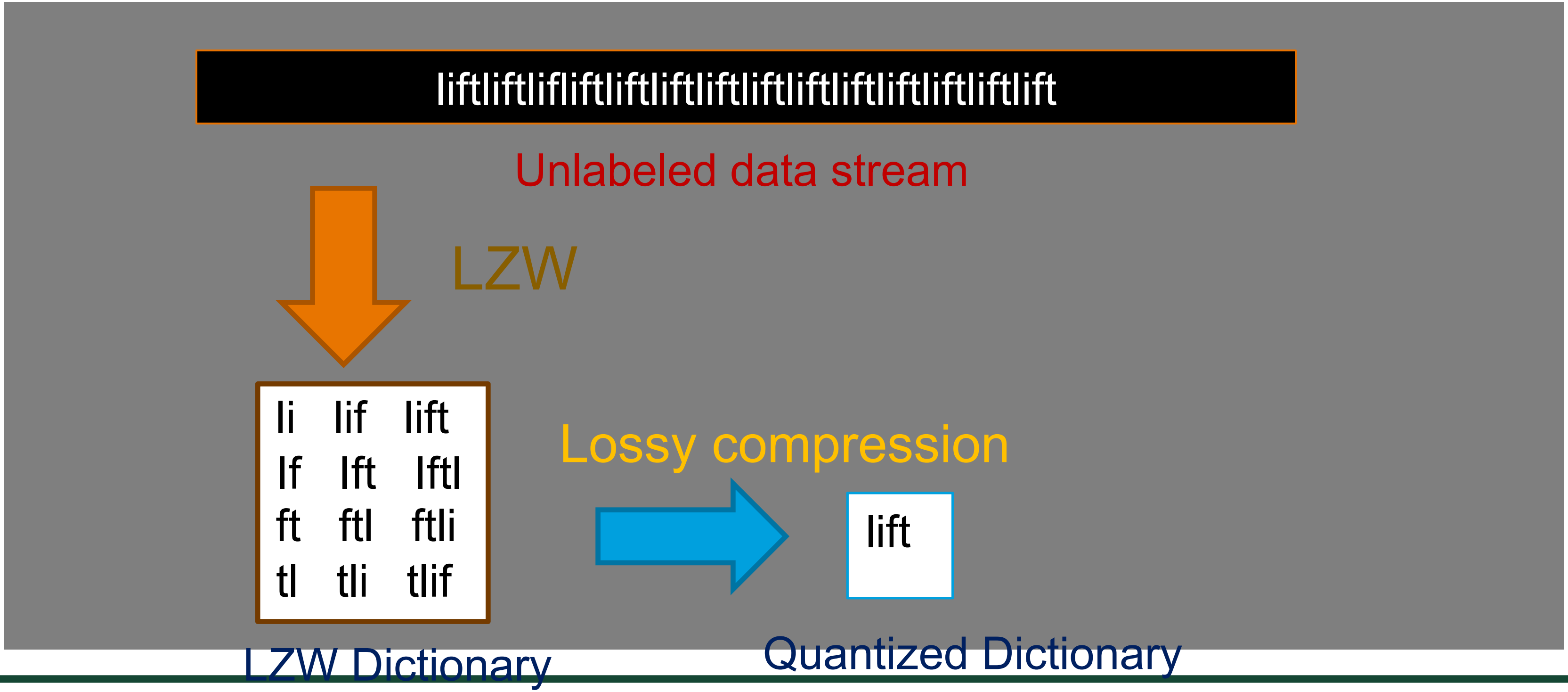
- **First challenge**: variability in sequence length

  Overcome: Generating a LZW dictionary with combinations of possible potential patterns in the gathered data using

  Lempel- Ziv- Welch algorithm (LZW)
- **Second Challenge**: Huge size of the Dictionary

Overcome: Compress the Dictionary

Example of LZW & Quantized Dictionary



Construct a Quantized Dictionary

- LZW Dictionary:

Contains set of patterns p_{ij} and their corresponding weights according to following Eq.

$$w_{ij} = f_{ij}$$

Here, f_{ij} is the frequency of the pattern *i* in chunk *j.*

- Quantized Dictionary:

$$\text{Max } \{(w_{ij} \text{ *Length } (p_{ij})\}, \text{ where } p_{ij} \subseteq P$$

Here, P is a set of possible combination of a particular pattern

Construct Quantized Dictionary using Compression Technique

Keep only the longest, frequent unique patterns according to their associated weight

Discarding other subsumed patterns.

Levenshtein Edit Distance is used to find longest pattern

$$\min \begin{cases} dist_{i-1,j-1} & + \begin{cases} 0 & \text{if} \quad p[i] = q[j] \\ 1 & \text{otherwise} \end{cases} \\ dist_{i-1,j} & + \quad 1 \\ dist_{i,j-1} & + \quad 1 \end{cases}$$

Agenda

- ❑ Detecting Cheats of Computer Game
  - ❑ Funded by NSF, AFOSR
- ❑ Website Fingerprinting
  - ❑ Funded by NSF, AFOSR
- ❑ Insider Threat Detection
  - ❑ Funded by NSF, AFOSR
- ❑ Secure Data Analytics 🡐=================
  - ❑ Funded by NSF, AFOSR
- ❑ Real Time Anomaly Detection
  - ❑ Funded by Sandia via DOE

Securing Data Analytics on SGX with Randomization

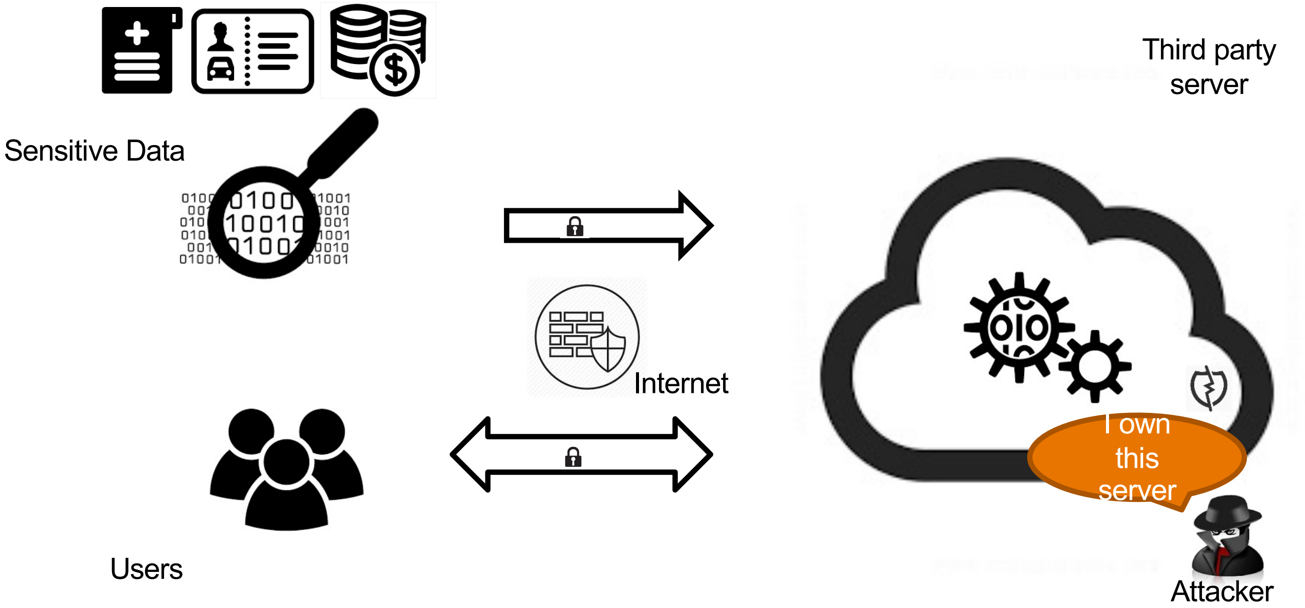
Swarup Chandra, Vishal Karande, Zhiqiang Lin, Latifur Khan, Murat Kantarcioglu

Department of Computer Science , University of Texas at Dallas.

This work is supported by NSF

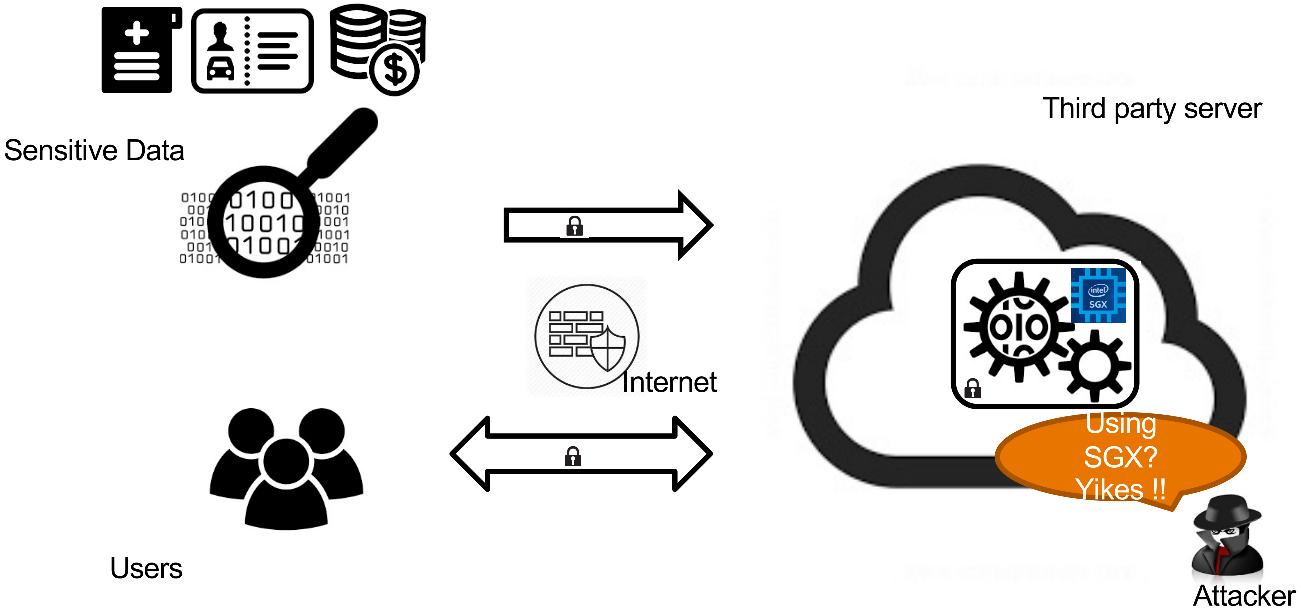
Swarup Chandra, Vishal Karande, Zhiqiang Lin, Latifur Khan, Murat Kantarcioglu and Bhavani Thuraisingham. 2017. Securing Data Analytics on SGX with Randomization. (Accepted in ESORICS - 2017).

Motivation

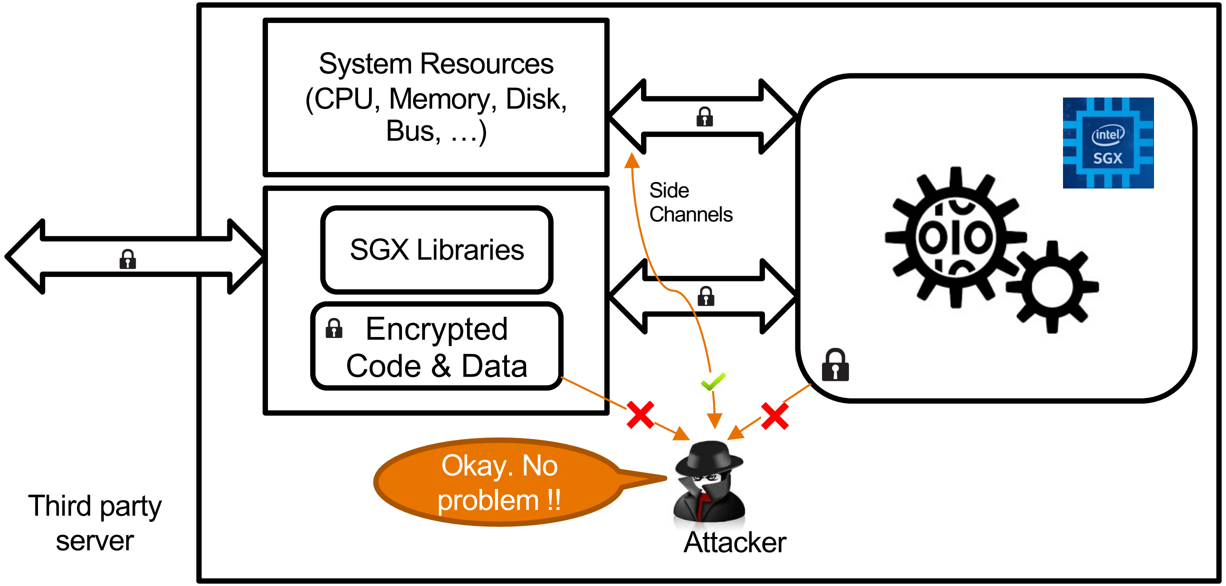


Sensitive Data
Users
Internet
Third party server
I own this server
Attacker

Motivation


Sensitive Data
Third party server
Internet
Users
Using SGX? Yikes !!
Attacker

Motivation



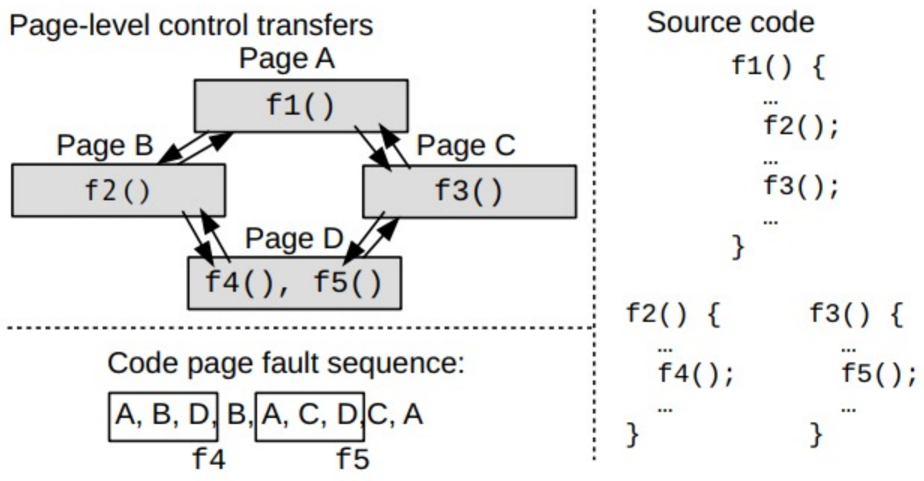
System Resources (CPU, Memory, Disk, Bus, …)
SGX Libraries
Encrypted Code & Data
Side Channels
intel
SGX
Okay. No problem !!
Attacker
Third party server

Motivation

Side Channels

- Page Faults can be controlled by adversary

Page access sequence can be observed by the adversary, revealing the sensitive function call.



Xu, Y., Cui, W., & Peinado, M. (2015, May). Controlled-channel attacks: Deterministic side channels for untrusted operating systems. In Security and Privacy (SP), 2015 IEEE Symposium on (pp. 640-656). IEEE. Chicago

Motivation

Defense against side-channel attacks

- Data Oblivious Solution
  - Eliminate data dependence memory access

```
int max(int x, int y) {
    if(x > y) {
        return x;
    } else {
        return y;
    }
}
```

a) Non-oblivious max

```
int max(int x, int y) {
    int d;
    if(x > y) {
        d = 1;
    } else {
        d = 0;
    }
    return (x*d + y*(1-d));
}
```

b) Oblivious max

Value of variables x and y are encrypted, and are placed in different registers.

Non-oblivious: return value depends on condition x > y.

Oblivious: variable d is accessed regardless of condition.

Motivation

Defense against side-channel attacks

- Data Oblivious Solution
  - Eliminate data dependence memory access
    - Access all array variables
    - Conditional statements: Access all variables independent of condition.
  - Data Analytics:
    - Protect confidential parameters
    - Example: Decision tree

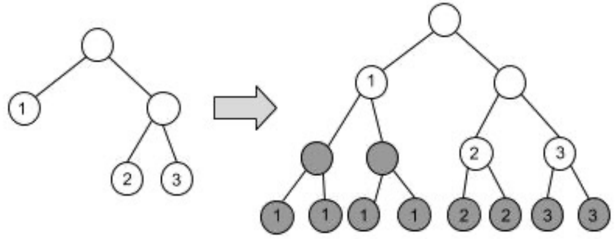
Public Parameters:
- Number of variables
- Number of class labels

Confidential Parameters:
- Tree structure
- Labels of data instances

Motivation

Securing Data Analytics

- Continuing Example: Decision Tree.
  - Tree structure secured by balancing the tree.



  - For every test data, all paths from root to leaf are accessed (i.e. every leaf is accessed)
  - Ignore label at leaf that are not part of the correct tree path according to test data.

Ohrimenko, O., Schuster, F., Fournet, C., Mehta, A., Nowozin, S., Vaswani, K., & Costa, M. (2016, August). Oblivious Multi-Party Machine Learning on Trusted Processors. In *USENIX Security Symposium* (pp. 619-636).

Objectives

What if large number of parameters are present?

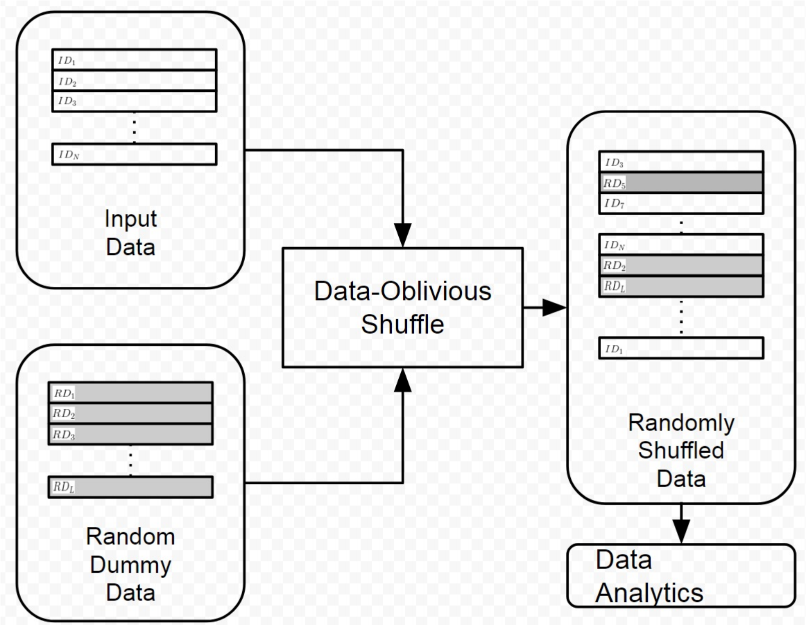
- Large decision tree
- Large number of variables
- Large domain size

Challenges

- Prohibitive execution time
- Complete data privacy may not be necessary

Secure Data Analytics

Data Oblivious Shuffling



Data Oblivious sorting using Batcher's Odd-even sort with oblivious comparison.

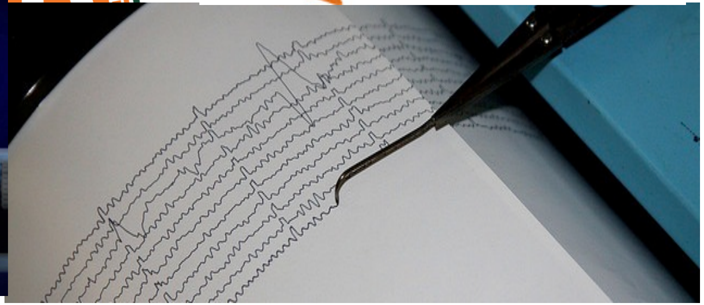
Traces captured by the adversary for input and dummy data are indistinguishable.

Batcher, K. E. (1968, April). Sorting networks and their applications. In *Proceedings of the April 30--May 2, 1968, spring joint computer conference* (pp. 307-314). ACM.

Secure Data Analytics

Decision Tree

- Create balanced decision tree.
- Given a test dataset of size n, generate L dummy data instances, $L \geq n$.
- Randomly shuffle with user-given test dataset.
- Evaluate class label for each instance sequentially.
  - Dummy data instance tracked obliviously.
  - Obliviously ignore results of dummy data.

Statistical Technique for Online Anomaly Detection Using Spark Over Heterogeneous Data from Multi-source VMware Performance Data

Mohiuddin Solaimani, Mohammed Iftekhar, Latifur Khan

The University of Texas at Dallas

M. Solaimani, M. Iftekhar, L. Khan, and, B. Thuraisingham, "Statistical technique for online anomaly detection using spark over heterogeneous data from multi-source VMware performance data." *Big Data (Big Data), 2014 IEEE International Conference on*. IEEE, 2014.

Anomaly Detection

- Real-time Anomaly Detection
  - ✓ Detecting anomaly on continuous stream data.

- Challenges
  - ✓ Data comes continuously from multi-sources.
  - ✓ Large volume of data.
  - ✓ High velocity of data.
  - ✓ Variation of data over time.

- Goal
  - ✓ Real-time anomaly detection.

Background

- VMware Performance Stream Data
  - ✓ We have used vSphere Guest SDK for collecting VMware statistics periodically.
  - ✓ Guest SDK gives several performance statistics of CPU, memory, like CPU elapse time, share CPU, used memory, reserve memory, etc.
  - ✓ We have also used unix tool ***mpstat*** (CPU statistics) and ***vmstat*** (memory statistics) and integrated with Guest SDK.
  - ✓ We have used *Kafka API* to send data to our distributed framework.

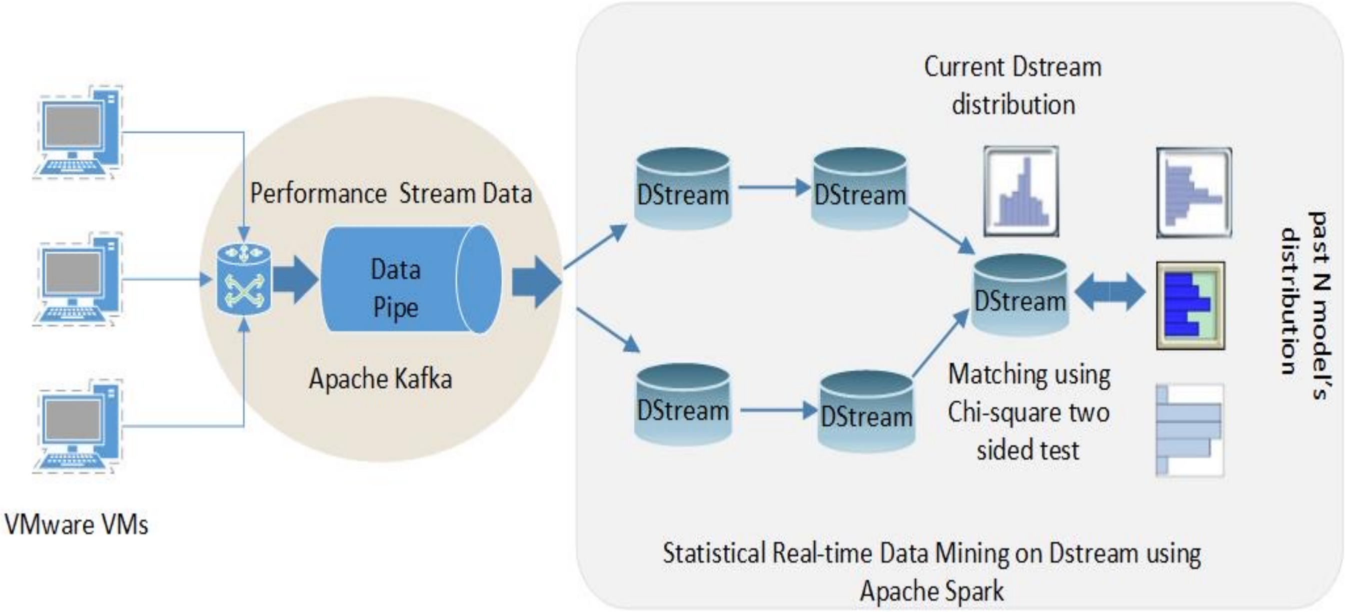
A sample data row (34 columns) looks like following.

Time stamp	VM name	IP	cpuReservationMHz	cpuLimitMHz	cpuShares	cpuUsedMs	…	Memory usage %
Thu Aug 21 15:28:41 2014	dmlhdpc8	10.176.148.58	0	0	0	**4294967295**	…	10.09

For our experiment, we have filtered the data and used CPU usage % and memory usage %.

Spark-based Statistical Anomaly Detection Framework

- Statistical Stream Data Mining Module



Open Source Software by Dr. Khan's group

Provider name	Tool name	Max 2 sentence description of tool's capabilities	URL where tool is available
Dr. Khan & Zhuoyi Wang	CPE	Tools for the Robust High Dimensional Stream Classification with Novel Class Detection	https://github.com/Vitvicky/Convolutional-Net-Prototype-Ensemble
	FASR	Tools for the adversarial representation learning framework under few labeled samples for stream mining	https://github.com/Vitvicky/FSAR
Dr. Khan & Yang Gao	StyleTransfer	A PyTorch Deep Style Transfer Library for Images	https://github.com/AlenUbuntu/StyleTransfer
	OAHU	Tools for Self-Adaptive Online Metric Learning	https://github.com/AlenUbuntu/OAHU
Dr. Khan & Shihab Islam	GCI	Transfer Learning Approach for Detecting Computer Game Cheats	https://github.com/shibz-islam/GCI
	BiMorphing	Tool for Defense Against Website Fingerprinting Attacks	https://github.com/shibz-islam/BiMorphing
Dr. Khan & Yifan LI	STGCN	Spatiol-temporal graph neural network based framework for time-seres forecasting	https://github.com/evanli05/ViEWS_Competition
Dr. Khan & Sayeed	RePAIR	Recommendation of political actors in real time using news articles - In this project, we extend the knowledgebase of actors required by automated event coders. We first recommend political actors found the news articles to the human annotators. They provide feedback and based on that we include the recommended actors to an online dictionary. This dictionary is later used by the automated event coders to generate political events.	https://github.com/openeventdata/political-actor-recommendation
	SPEC	Spark Based Political Event Coding - We created a framework to run automated event coder in distributed manner to encode large number of unprocessed news articles and generate political events.	https://github.com/openeventdata/SPEC_CORENLP
	Web-Scraper	The news article collector is running on a single node, collecting ~ 10K-13K news articles daily from ~400 news sources. This is the input to the SPEC and RePAIR projects described above.	https://github.com/Sayeedsalam/web-scraper-and-crawler
	Event Data API	We created this project to facilitate the access to the Event Dataset being developed here at UTD. It has REST API and running at http://eventdata.utdallas.edu	https://github.com/Sayeedsalam/spec-event-data-server