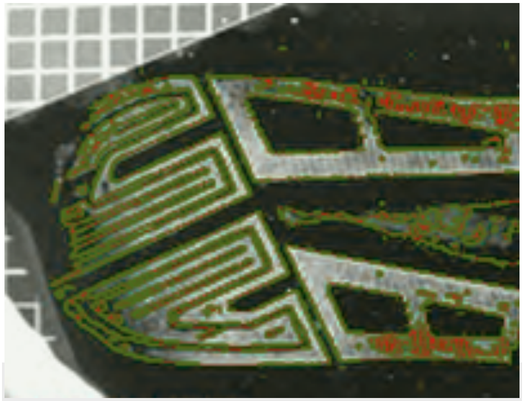
Computational Forensics

Proactive, Ultra-large scale Forensic Investigations

Katrin Franke, Hai Thanh Nguyen
Lars Arne Sand, Jarle Kittelsen, Peter Ekstrand Berg
Norwegian Information Security Laboratory (NISlab),
Department of Computer Science and Media Technology,
Gjøvik University College

www.nislab.no

Forensic Science



Forensic experts study a broad area of objects, substances (blood, body fluids, drugs), chemicals (paint, fibers, explosives, toxins), tissue traces (hair, skin), impression evidence (shoe or finger print, tool or bite marks), electronic data and devices (network traffic, e-mail, images).
Some further objects to be studied are fire debris, vehicles, questioned documents, physiological and behavioral patterns.

Forensic methods consist of multi-disciplinary approaches to perform the following tasks:

- **Investigate** and to **Reconstruct** a crime scene or a scene of an accident,
- **Collect** and **Analyze** trace evidence found,
- **Identify**, **Classify**, **Quantify**, **Individualize** persons, objects, processes,
- **Establish linkages**, associations and **reconstructions**, and
- Use those findings in the prosecution or the defense in a court of law.

So far, mostly dealt with previously committed crime, greater focus is now to prevent future crime.

- Paradigm shift: **Proactive Forensics !**

Høgskolen i Gjøvik — Forensics Lab TESTIMON

Crime in the Modern World

- Digital devices are used everywhere:
  - **Computers,**
  - **Mobile phones, PDAs,**
  - **Cameras,**
  - **Printers, Copy machines,**
  - **Videogame consoles, etc.**

- Used to plan/conducted **physical and digital crimes**
- Digital evidence:
  - Threatening emails or chats messages
  - Documents (e.g., in places they shouldn't be)
  - Suicide notes
  - Bomb-making diagrams
  - Malicious Software: Viruses, Worms, Botnet …

  - Evidence that network connections were made between machines
  - System registry, Event logs, Print spool, Swap files, Recycle bin (trash)
  - Mobile phone SMS messages, Contacts, Connections etc.

HØGSKOLEN I GJØVIK

Forensics Lab TESTIMON

Current Forensic Practice

- Evidence becoming increasingly **data intensive** and **widely distributed**
- Common practice to **seize all data carriers**; amounts to **many terabytes of data**
- Enrich with data available on the **Internet, Social networks,** etc.

- Huge amounts of Paper Documents

Current Situation

- Knowledge and intuition of the human expert plays a central role in daily forensic casework.
- Courtroom **forensic testimony** is often criticized by defense lawyers as **lacking a scientific basis**.
- **Huge amount of data**, tide operational times, and data linkage pose challenges.

Computational Forensics, aka applying Artificial Intelligence Methodologies in Forensic Sciences

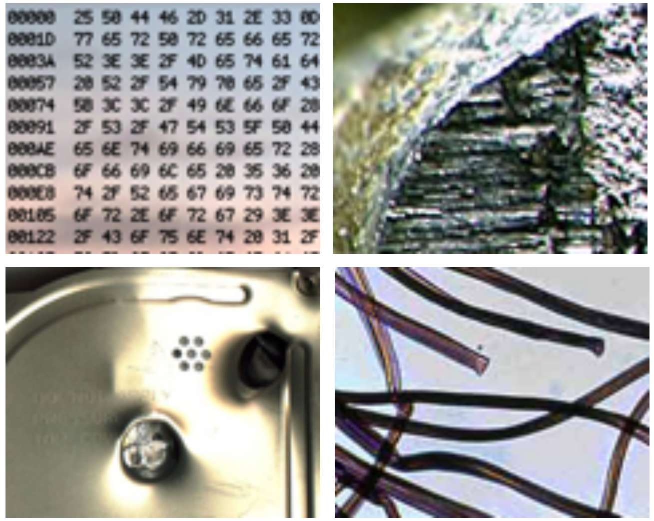
Crime in the Modern World cont.

- Massive amount of data:
  - 247 billion email per day
  - 234 million websites
  - 5 billion mobile-phone users
- ICT Infrastructures:
  - Complex, rapidly growing
  - Dynamically changing
  - Hostile, adversary environment
- Cybercrime:
  - One million victims daily
  - Expected losses 297 billion Euro
  - Crowd sourcing -> Crime sourcing
  - Flash mobs -> Flash robs

- Proactive, Ultra-large scale Forensic Investigations, Computational Forensics:
  - Situation-aware methods
  - Quantified, measurable indicators
  - Adaptive, self-organizing models
  - Distributed, cooperative, autonomous
- Rule-of-Law:
  - Culture, social behaviours
  - Legal & privacy aspects
  - Cross-jurisdiction cooperation
  - European / International cyberlaw
  - Law as framework for ICT
  - Law as contents of ICT, Automation, programming of legal rules

Forensics Lab TESTIMON

Challenges and Demands in Forensic Science



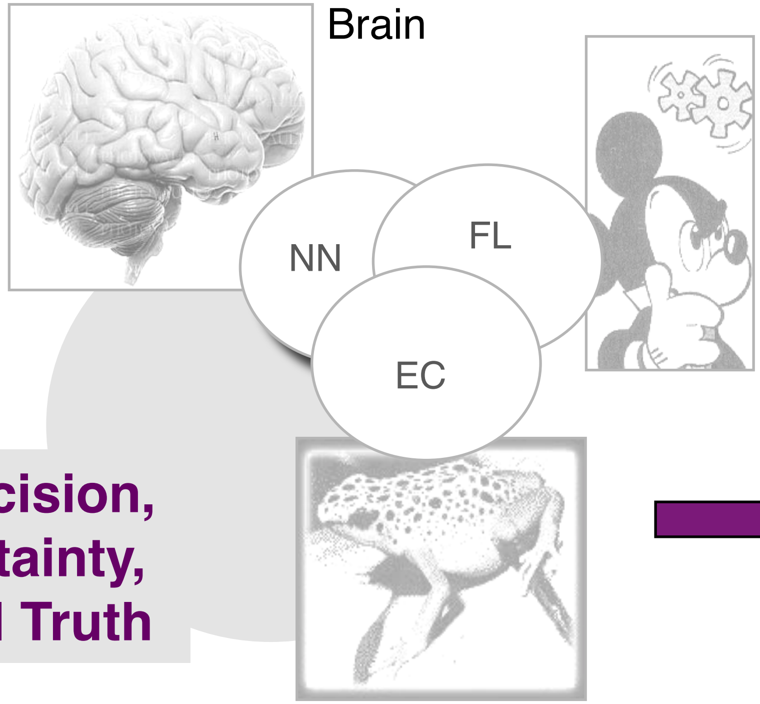
Challenges:

- **Tiny Pieces of Evidence** are hidden in a mostly **Chaotic Environment**,
- Trace Study to reveal **Specific Properties**,
- Traces found will be **Never Identical**,
- Reasoning and Deduction have to be performed on the basis of
  - **Partial Knowledge,**
  - **Approximations,**
  - **Uncertainties** and
  - **Conjectures**.

Demands:

- **Objective** Measurement and Classification,
- **Robustness** and **Reproducibility**,
- **Secure** against Falsifications.

Requirement of Adapted Computer Models & Operators



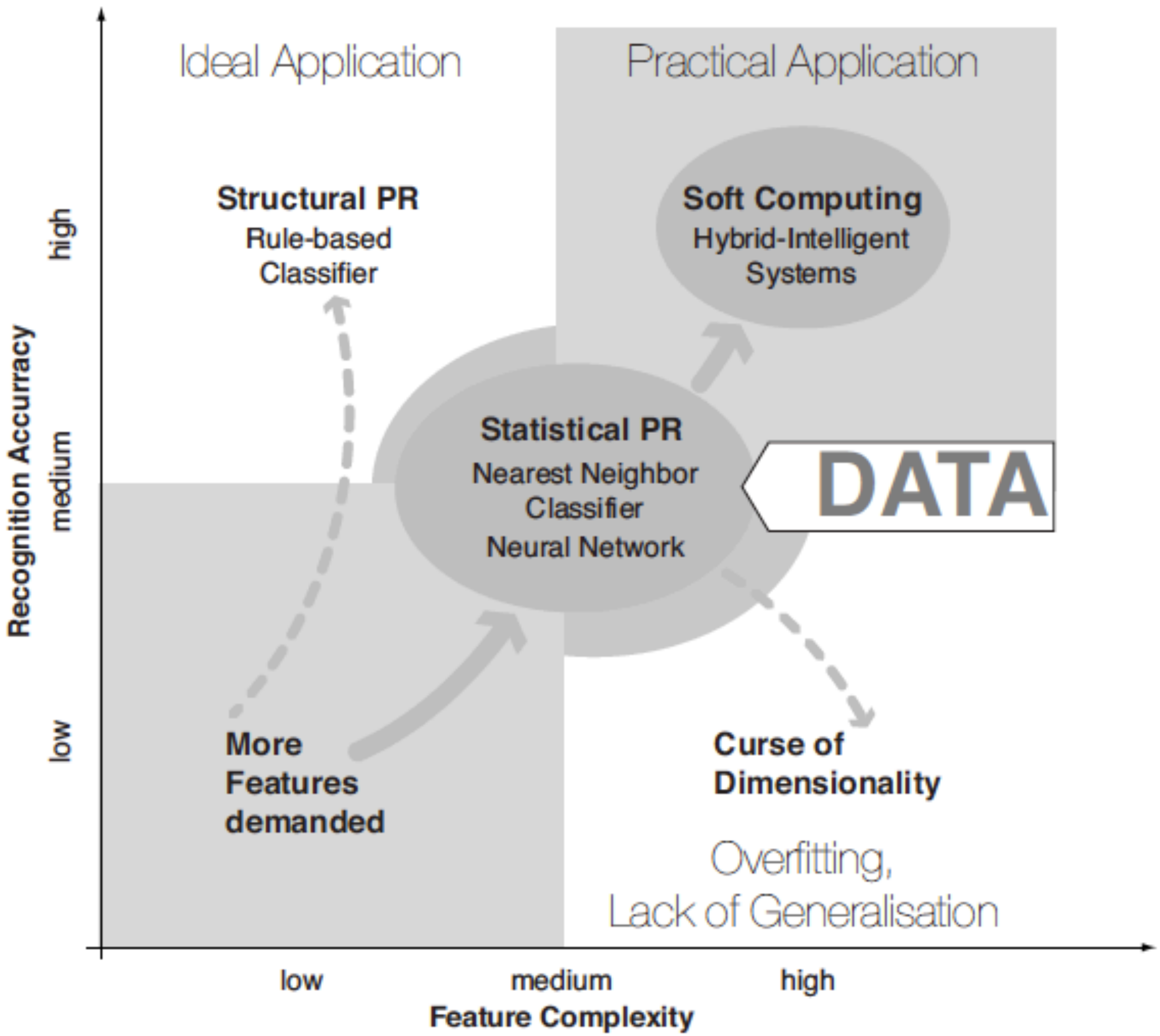
Imprecision, Uncertainty, Partial Truth

→ ***Computational Intelligence***

NN: Neuronal Networks
FL: Fuzzy Logic
EC: Evolutionary Computation

Computational Forensics

Data-driven Approaches



Big Data Analysis

Inter-relation of feature complexity and expected recognition accuracy.
(Franke 2005)

Reverse Engineering Malware

Katrin Franke, Hai Thanh Nguyen,
Lars Arne Sand, Petter Ekstrand Berg, Jarle Kittelsen
Norwegian Information Security Laboratory (NISlab)
Gjøvik University College
www.nislab.no

Taxonomy

- **Malware**
  - Short for malicious software
  - Defined as any program or file that is harmful to a computer environment or its user
- **Software –> Benign**
  - Software is a general term for the various kinds of programs used to operate computers and related devices

Obfuscation Techniques

- **Compression & Encryption**
  - Typically consist of a encrypted/compressed part and a decryption/decompression routine
- **Polymorphism**
  - Evade detection by obfuscating decryption algorithm
- **Metamorphism**
  - Evade detection by ***obfuscating entire malware***
- **Packers**
  - Software applications that store encrypted or compressed executables (packed), in such a way that when executed, the packed executable is loaded into memory and executed

Application Examples: Reverse Engineering Malware

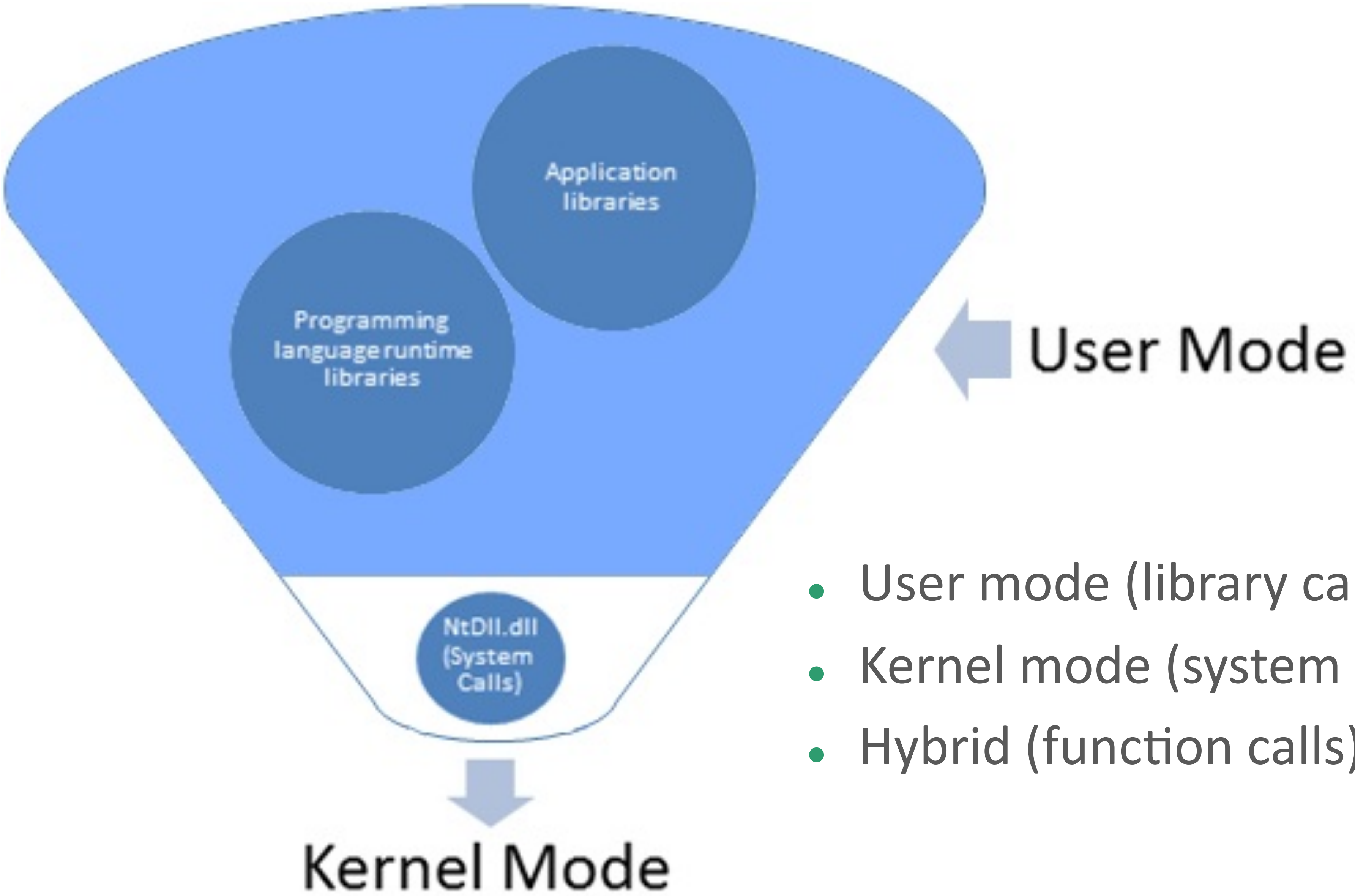
- **Static analysis**
- System artifacts
- **Dynamic analysis**
- Debugging
- Analyzing malicious content
  - **PDFs**
  - **JavaScripts**
  - Office documents
  - Shellcode
  - Network traffic

Behavioral Malware Detection
(static, dynamic, combined)

Lars Arne Sand, Peter Ekstrand Berg
Katrin Franke, Hai Thanh Nguyen
Norwegian Information Security Laboratory (NISlab)
Gjøvik University College
www.nislab.no

Computational Forensics

Layers of Detection



- User mode (library calls)
- Kernel mode (system calls)
- Hybrid (function calls)

Static analysis

- Static analysis
  - Does not execute malware
  - Analyze:
    - System artifacts
    - Debugging
    - Source code (not included)
    - Disassembled code (not included)

Dynamic analysis #1

- Definition
  - *Dynamic analysis is the process of executing malware in a monitored environment to observe its behaviors*
- Deals with finding and understanding the changes made to the system
- Pro:
  - Provide quick information about created and changed files, registry keys, processes, handles, contacted websites, etc.
- Con:
  - Excessive and overwhelming results
  - Need to know the normal behavior of a system

Dynamic analysis #2

- Methods of change detection:
  - Hook-based
    - Hook API functions in user mode or kernel mode to detect changes on a system
    - Tools: Process monitor
  - Difference-based
    - Monitor tools that take and compare snapshots from pre and post execution of malware
    - Tools: Regshot, InCtrl5, Winanalysis
  - Notification-based
    - Detect notification routines that the system automatically generate during specific events
    - Tools: Process monitor, Preservation

Computational Forensics

Motivation & Research questions

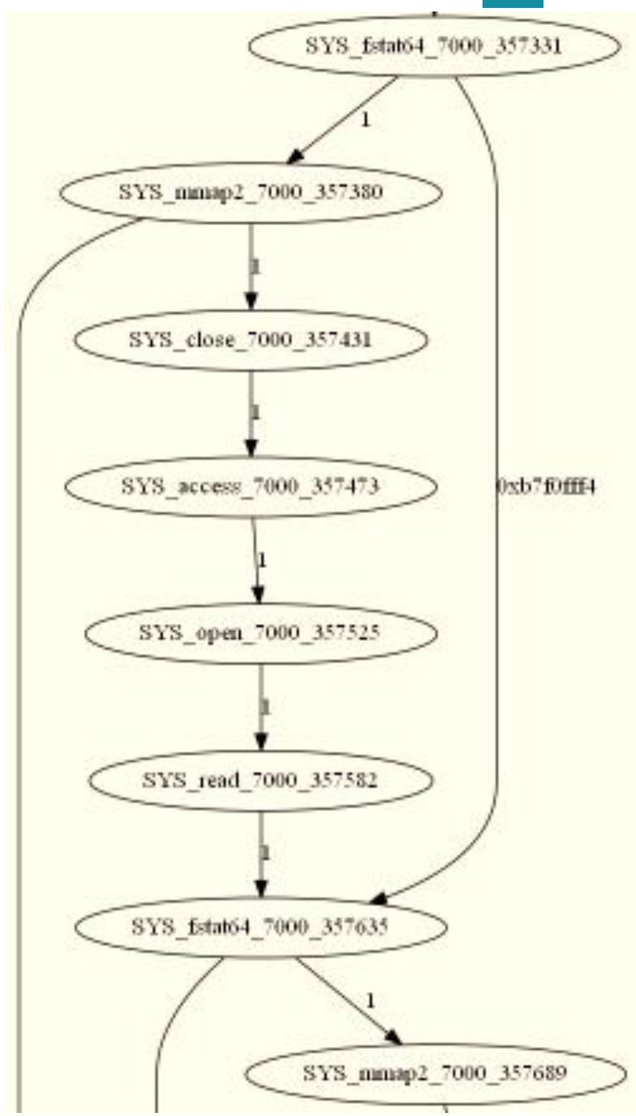
- Motivation
  - Dynamic, anomaly-based detection
  - High obfuscation resilience

- Research questions
  - Is the use of information-based dependency matching reliable?
  - Will information-based dependency matching increase obfuscation resilience?
  - How does system call detection differentiate to library call detection, wrt. detection rate, obfuscation resilience, throughput

Forensics Lab TESTIMON

Information-based dependency matching #1

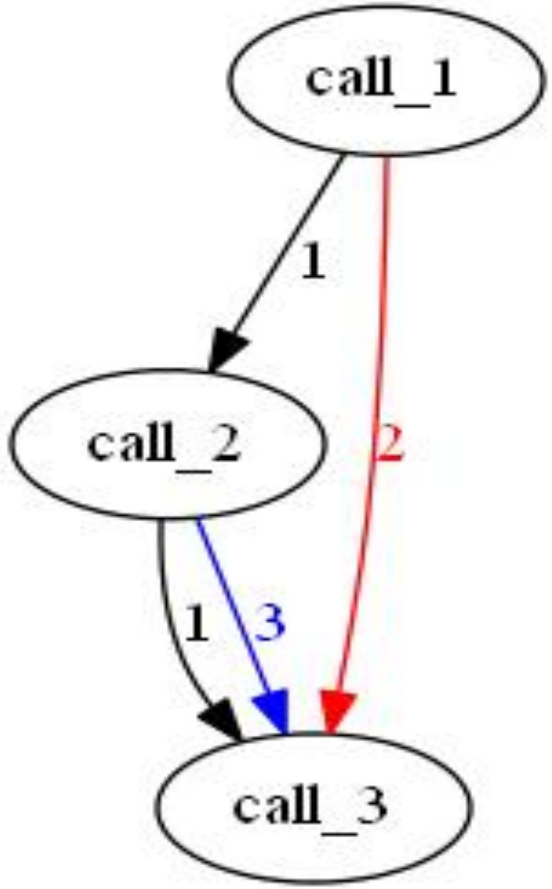
- 3 dependency rules
  - Ordering dependencies
    - Infer dependency based on sequence
  - Value dependencies
    - Compare parameter of the function call with parameters of previous function calls
  - Def-use dependencies
    - Compare parameter of the function call with return values of previous function calls
- Only memory values are used



Computational Forensics

Information-based dependency matching #2

- Ordering dependency (**1**)
  - sequence
- Value dependency (**2**)
  - parameters
- Def-use dependency (**3**)
  - Parameter and return value
- Sample:
  - call_1(parameter1,**ffff0000**)=0
  - call_2(par)=**0x4fff0418**
  - call_3(**0x4fff0418**,**0xffff0000**)=0



Example #1

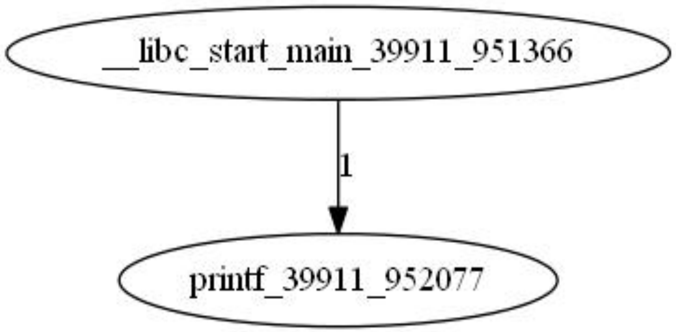
- Library calls (Hello World.c)
  - Code

```c
#include <stdio.h>

int main() {
    printf("Hello world!!!");

    return 0;
}
```

  - Trace

```
11:05:11.951366 __libc_start_main(0x80483c4, 1, 0xbf96afa4,
0x8048400, 0x80483f0 <unfinished ...>
11:05:11.952077 printf("Hello world!!!")          = 14
11:05:11.953227 +++ exited (status 0) +++
```

  - Graph



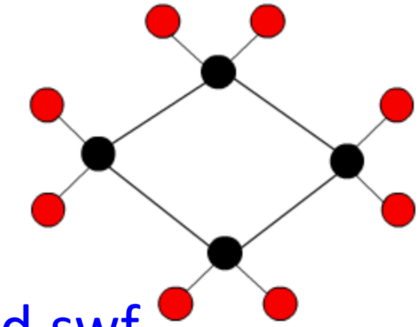
Example #2

- System calls (Hello world.c)
  - Trace
    - Much more extensive due to memory mapping
    - Example trace
  - Graph
    - Example Graph

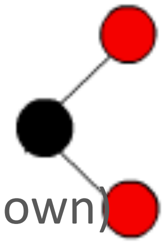
Example #3

- Actual malware example
  - Malware system call Graph Examples
    - Virus.Linux.Snoopy.a
    - Rootkit.Linux.Matrics.a
    - Exploit.Linux.Small.k

Experimental Design & Data Set #1



- Graph-based Matching
  - http://ailab.wsu.edu/subdue/unsupervised.swf
  - Subdue finds substructures by compressing graphs
  - Supervised Learning is performed by finding substructures that occur frequently in one class but seldom in another



- Dataset
  - Malware
    - Extracted from: vx.netlux.org/index.html (currently down)
    - 190 samples: 7150 vertices, 7790 edges
  - Benign Software
    - Ubuntu binaries
    - 75 samples: 9025 vertices, 9395 edges

Computational Forensics

Preliminary Results #1: Graph-based Matching

- Detection rate of 98,9%
- Confusion matrix

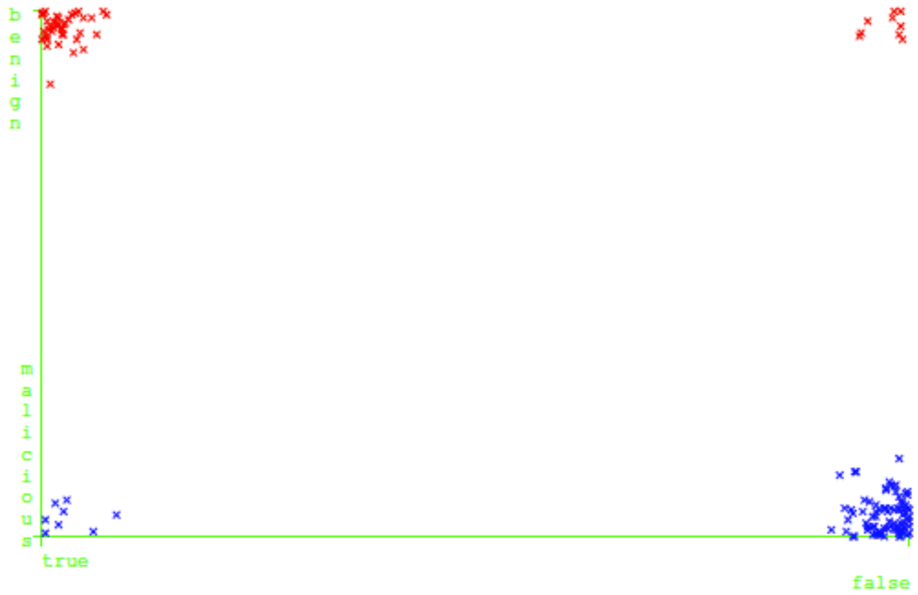
System calls		
	Classified as	
Correct class	Malware	Software
Malware	190	0
Software	3	72
	1	0,96

  - 190/190 malware correctly classified
  - 72/75 software correctly classified

Experimental Design & Data Set #2

- Feature Encoding and Classification
  - Naive-Bayes, K-nearest neighbors, C4.5, SVM and Bayesian Network
- Botnet Malware
  - Vxheavens, packetstorm, offensivecomputing
  - Three malware families: *SpyBot, Torpig, SdBot*
- Benign Software
  - Pendriveapps

Experimental Results #2



- Feature Correlation

Feature Sets	*Bigger-than 0.1*	*Less-than 0.09*
Static	86.96	13.04
Dynamic	97.24	2.76

- Feature Selection

Feature Sets	**Full-set**	**GeFS$_{CFS}$**	**GA$_{CFS}$**
Static	1814	7	11
Dynamic	5494	5	19
Combination	7308	9	30

Experimental Results #2

- True positives (TP)
- True negatives (TN)
- False positives (FP)
- False negatives (FN)

$$\text{DetectionRate} = \frac{TP}{TP + FN}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Feature Sets	Average Detection Rate			Average Accuracy		
	Full-set	$GeFS_{CFS}$	GA_{CFS}	Full-set	$GeFS_{CFS}$	GA_{CFS}
Static	97.84	97.2	97.85	88.05	91.33	92.92
Dynamic	88.60	87.74	87.74	86.01	89.37	90.77
Combination	93.76	95.11	93.77	87.41	92.88	92.87

Detecting Malicious PDF

Jarle Kittelsen, Katrin Franke, Hai Thanh Nguyen

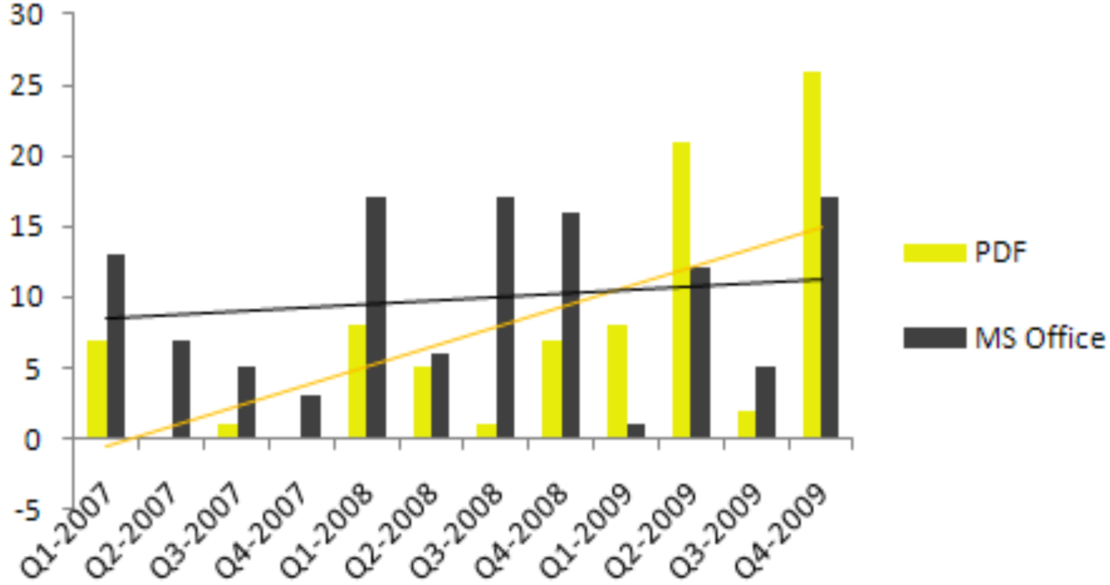
Norwegian Information Security Laboratory (NISlab)

Gjøvik University College

www.nislab.no

Analyzing malicious content #1

- Frequent analysis:
  - **PDF**
  - **JavaScript**
  - Office Documents
  - Flash (not included)
  - Shellcode
  - Network Traffic



Analyzing malicious content #2

- Challenges
  - Proprietary file formats
  - Obfuscation techniques
  - Vast volume of vulnerabilities and exploit methods
  - Dependent circumstances, e.g.
    how to trigger and enable vulnerability or malware
  - Embedded malicious code?
  - Rapidly evolving threats

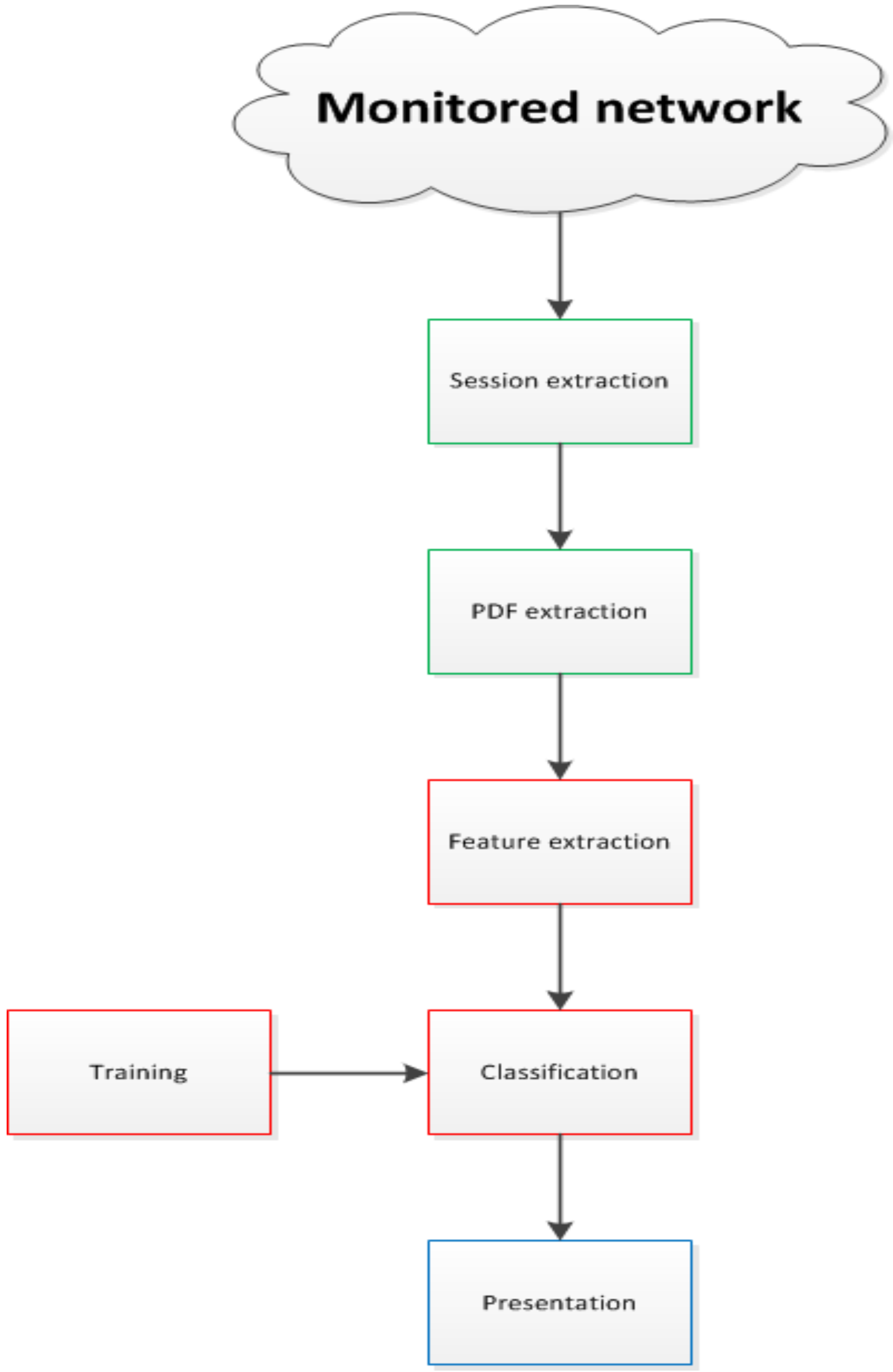
Analyzing malicious content #3

- General Approach:
  - Locate potentially malicious embedded code, such as shellcode VBA macros or JavaScript
  - Extract suspicious code segments from file
  - If relevant:
    - Deobfuscate
    - Disassemble
    - Debug
  - Understand next steps in infection chain

Research Questions

- Which **features** are significant for detecting malicious PDF documents?
- Which **classifier design** and configuration yields optimal performance in malicious PDF detection?
- How can a real-world IDS, capable of detecting malicious PDFs in network traffic, be implemented?

Method Overview



Research Approach

- Collect extensive dataset for experimentation.
- Perform literature study to reveal STOA and "expert knowledge" on detecting/analyzing malicious PDF.
- Extract an expert knowledge feature vector.
- Perform feature selection on the feature vector.
- Train, test and optimize a machine learning classifier for detecting malicious PDF.
- Implement and evaluate real-world PDF analysis.

Data Collection

- PDFs collected within the malware research community and through webcrawling, e.g.,
  - Websense
  - Abuse.ch
  - Sourcefire
- Malicious samples have been submitted globally and detected in various ways, some of the samples are under NDA.
- Data set in total:
  - **7,454 unique benign** PDF samples.
  - **16,296 unique malicious** PDF samples.

Expert-Knowledge Features (KPI)

- Keys from the PDF format (ISO 32000) relevant to malicious PDFs, e.g.,
  - /JavaScript
  - /OpenAction
  - /AcroForm
- Key selection based upon the independed research by (i) Didier Stevens, (ii) Paul Baccas.
- **18 features (keys)** are selected to initialize.
- Additional **feature-set for Javascript**.

Experiments (Exp 1…4)

1. Feature & Classifier Selection
2. Classifier Optimalization and Testing
3. Real-world testing
4. Embedded javascripts

Exp 1: Feature & Classifier Selection

Original feature vector (18):

AA, RichMedia, xref, Encrypt, JBIG2Decode, Launch, JavaScript, OpenAction, Colors,JS, obj_mis, startxref, AsciiHexDecode, ObjStm, AcroForm, stream_mis, Page, trailer

Golub-score feature selection (7):

$$F(x_i) = \left| \frac{\mu_i^+ - \mu_i^-}{\sigma_i^+ + \sigma_i^-} \right|$$

JavaScript, OpenAction, JS, obj_mis, AcroForm, Page, trailer

Generic feature selection GeFS (5):

JavaScript, JS, startxref, Page, trailer

$$GeFS(x) = \frac{a_o + \sum_{i=1}^{n} A_i(x)\; x_i}{b_o + \sum_{i=1}^{n} B_i(x)\; x_i}$$

Computational Forensics

Exp 1: Feature & Classifier Selection

Tested perfomance using 5 different classifiers:

	BayesNet			C45/J48			RBFNet		
	18	**7**	**5**	**18**	**7**	**5**	**18**	**7**	**5**
Bal succ	0.973	0.94	0.976	0.995	0.995	0.975	0.718	0.797	0.874
Auc	0.996	0.995	0.996	0.997	0.998	0.994	0.879	0.922	0.926

	MLP			SVM		
	18	**7**	**5**	**18**	**7**	**5**
Bal succ	0.96	0.966	0.920	0.995	0.995	0.977
Auc	0.985	0.987	0.978	0.995	0.996	0.974

Choose **7 features** from Golub-score selection, **SVM* classifier** for further experimentation.

*SVM - Support Vector Machine

*Bal succ - Balanced Successrate *AUC - Area Under (ROC) Curve

Exp 2.1: Classifier Optimalization

Goal: Find optimal values for the "*inverse-width paramerter (gamma)*" and *"the penalty value (C)"* of the SVM kernel by means of empirical testing.

- Performed grid search

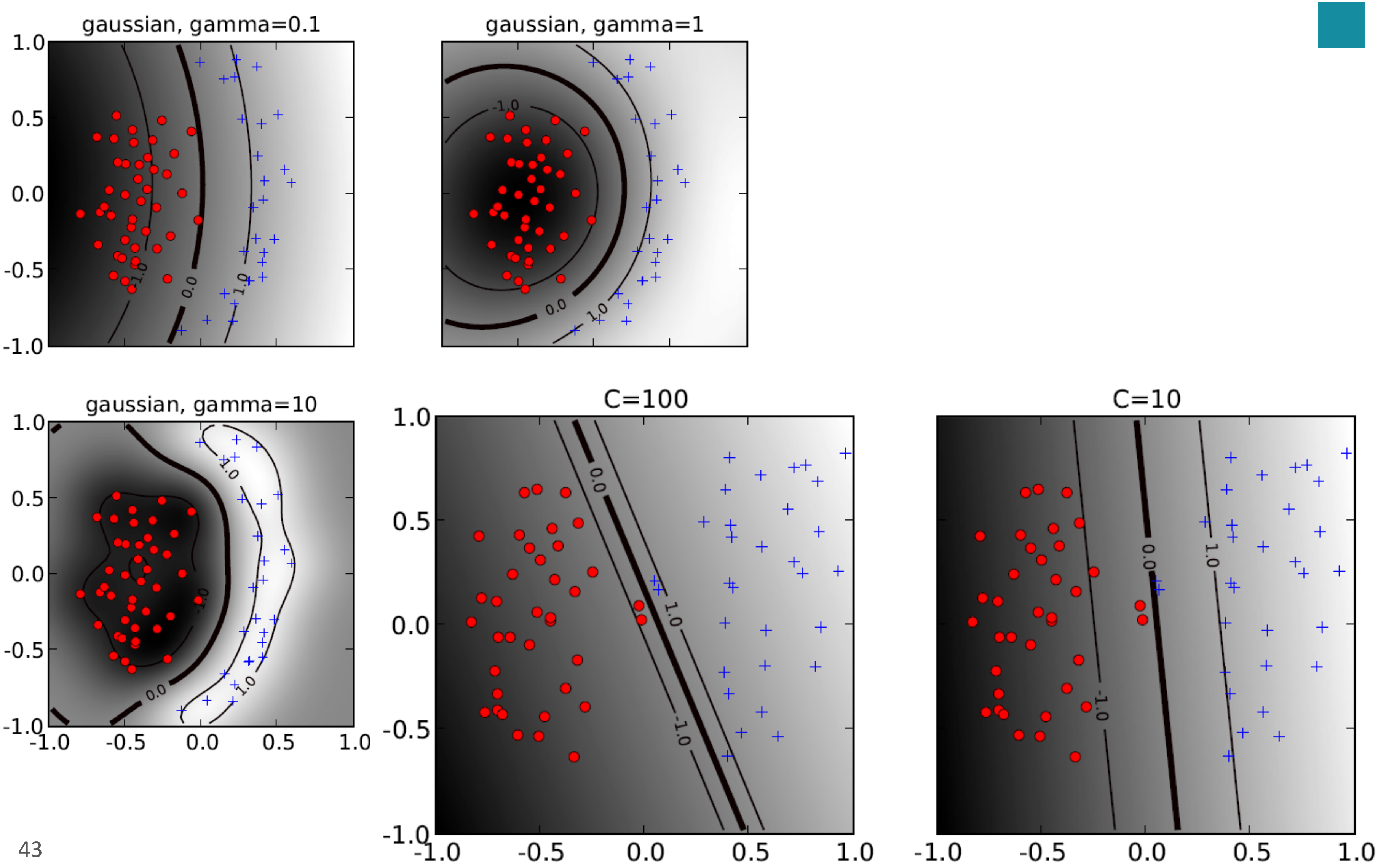
  C={0.1,1,10,100,1000} γ={0.01,0.1,1,10}

- Optimal values:

  C=100

  γ= 0.1

Exp 2.1: Classifier Optimalization



Exp 2.2: Classifier Reliability

- Goal: Test the reliability of the classifier over multiple iterations of training and testing.
- Performed 5 iterations of 10-fold cross-validation:

	Bal. succ.	AUC
1	0.9945	0.9963
2	0.9947	0.9969
3	0.9948	0.9970
4	0.9947	0.9965
5	0.9946	0.9974

- Classifier deemed reliable.

Exp 2.3: Classifier Generalization

- Goal: Test whether the classifier is able to correctly classify PDF samples guaranteed to not be a part of the training dataset.
- Nine (9) new exploits created after the date of dataset creation.
- Classifier detects 7 out of 9.
  - DoS and passes 50.000 bytes of random data.
  - Dumps and executes an .exe file. Could be detected with a new feature. (Will be implemented)

Exp 3: Real-world Test

- Goal: Test performance in real-world setting.
- Performed at Norwegian Defence Center for Protection of Critical Infrastructure.
- SNORT -> tcpflow -> Header removal script -> Extraction script -> Classification script
- Experiment ongoing.

Exp 4: Embedded Javascript

- Goal: Explore the possibilites of classifying javascripts embedded in PDF documents to serve as a starting point for future research

- 90% of malicious PDF contain malicious JS.
- Created our own new JS feature vector:
  - function, eval_length,max_string, stringcount, replace, substring, eval, fromCharCode
- Using full feature set and same SVM setup:
  - Balanced successrate: **0.90**

Computational Forensics

Discussion and Summary

- The dataset
  - Difficulties controlling factors
  - Best solution: MD5, generalization experiment, big dataset from many sources.
- Changes over time
  - Need for re-learning
  - Online learning
- Detecting malicious PDF documents is feasible
  - using reduced expert feature set, javascript features, SVM
- Aquired knowledge & lessons learned:
  - A PDF dataset (**16.296 / 7,454**) for future reseach.
  - Knowledge on significant features for PDF classification.
  - A method for automated detection of malicious PDF in network traffic.
  - A starting point for future research on malicious javascript detection.

Concluding Remarks

- Computational forensics holds the potential to greatly benefit all of the forensic sciences.
- For the computer scientist it poses a new frontier where new problems and challenges are to be faced.
- The potential benefits to society, meaningful inter-disciplinary research, and challenging problems should attract high quality students and researchers to the field.

5th International Workshop on Computational Forensics

Tsukuba, Japan, November 11, 2012

in conjunction with Intern. Conf. of Pattern Recognition

http://iwcf12.arsforensica.org

Forensics Lab TESTIMON

Further Reading

- NAS Report: *Strengthening Forensic Science in the United States: A Path Forward* http://www.nap.edu/catalog/12589.html
- van der Steen, M., Blom, M.: *A roadmap for future forensic research*. Technical report, Netherlands Forensic Institute (NFI), The Hague, The Netherlands (2007)
- M. Saks and J. Koehler. *The coming paradigm shift in forensic identification science.* Science, 309:892-895, 2005.
- Starzecpyzel. *United states vs. Starzecpyzel. 880 F. Supp. 1027 (S.D.N.Y),* 1995.
- http://en.wikipedia.org/wiki/Daubert_Standard
- C. Aitken and F. Taroni. *Statistics and the Evaluation of Evidence for Forensic Scientists. Wiley,* 2nd edition, 2005.
- K. Foster and P. Huber. *Judging Science.* MIT Press, 1999.
- Franke, K., Srihari, S.N. (2008). *Computational Forensics: An Overview*, in Computational Forensics - IWCF 2008, LNCS 5158, Srihari, S., Franke, K. (Eds.), Springer Verlag, pp. 1-10.
- http://sites.google.com/site/compforgroup/
- Nguyen, H., Franke, K., Petrovic, S. (2010). *Towards a Generic Feature-Selection Measure for Intrusion Detection*, In Proc. International Conference on Pattern Recognition (ICPR), Turkey.
- Nguyen, H., Petrovic, S. Franke, K. (2010). *A Comparison of Feature-Selection Methods for Intrusion Detection*, In Proceedings of Fifth International Conference on Mathematical Methods, Models, and Architectures for Computer Networks Security (MMM-ACNS), St.Petersburg, Russia, September 8-11. (accepted for publication)
- Nguyen, H., Franke, K., Petrovic, S. (2010). *Improving Effectiveness of Intrusion Detection by Correlation Feature Selection*, International Conference on Availability, Reliability and Security (ARES), Krakow, Poland, pp. 17-24.
- Flagien, A.O., Arnes, A., Franke, K., (2010). *Cross-Computer Malware Detection in Digital Forensics*, Techn. Report, Gjøvik University College, June 2010.

Thank you for your consideration of comments!

Getting in touch
WWW: kyfranke.com
Email: kyfranke@ieee.org
Skype/gTalk: kyfranke

Computational Forensics

Katrin Franke, PhD, Professor

- Professor of Computer Science, 2010
  PhD in Artificial Intelligence, 2005
  MSc in Electrical Engineering, 1994
- Industrial Research and Development (16+ years)
  Financial Services and Law Enforcement Agencies
- Courses, Tutorials and post-graduate Training:
  Police, BSc, MSc, PhD
- Chair IAPR/TC6 – Computational Forensics
- IAPR Young Investigator Award, 2009
  International Association of Pattern Recognition

kyfranke.com

Forensics Lab TESTIMON