Intelligent File Scoring System for Malware Detection from the Gray List

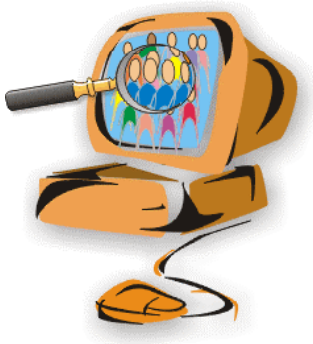
Yanfang Ye: Xiamen University & Anti-virus Laboratory, Kingsoft Corporation
Tao Li: School of Computer Science, Florida International University
Qiangshan Jiang & Zhixue Han: Xiamen University
Li Wan: Anti-virus Laboratory, Kingsoft Corporation

Talk Outline

- ***Introduction and Motivation***
- System Architecture and Description
- Experimental Results and Case Studies

Motivation

- Malware is a generic term to denote all kinds of unwanted software (e.g., virus, spayware, and worms)
- The most significant lie of defense is the ***signature-based*** method
  - Used in many anti-virus products
  - Common Practices
    - Authenticating valid software from a whitelist
    - Blocking invalid software from a blacklist
    - Running any unknown software (i.e., ***the gray list***) in a controlled manner



The Gray List

- Contains unknown software programs which could be either normal or malicious
- Usually authenticated or rejected manually by virus analysts
- The number of file samples in the gray list that need to be analyzed by virus analysts on a daily basis is constantly increasing
  - The gray list collected by the Anti-virus Lab of Kingsoft corporation usually contains more than ***100,000*** file samples per day
- The gray list is not only large in size, but also very complicated
  - Contains the variants of known malware and previously unknown malware samples
  - Imbalanced data distributions: majority samples are normal
  - Data characteristics change over time (due to malware rewriting techniques)
- It is thus of paramount importance to be quickly and automatically analyze the gray list and detect malware samples
  - Intelligent malware detection

Intelligent Malware Detection

- Many research efforts have been conducted on developing intelligent malware detection systems

- The detection process is divided into two steps: ***feature extraction*** and ***automatic categorization***

- Feature Extraction: extracting features to capture the characteristics of file samples

  - Application Programming Interface (API) calls
  - Program behavior strings

- Automatic Categorization: automatically categorize the file samples into different classes based on computational analysis on the feature representations

  - Decision Tree
  - Support Vector Machines (SVMs)
  - Naïve Bayes
  - Associative Classifiers

Limitations

- Different feature representation capture different characteristics of file samples
  - API calls reflect the behavior of program code pieces
  - Program strings consisted of reused code fragments, author signatures, file names and system resource information
- None of the single feature set can immune or resistant to mimicry designed to confuse the anti-virus software
- Different categorization methods have their own advantages and limitations
- Can we combine different feature representations and categorizations methods to improve the performance of malware detection

Two Heads are Better than One

- Ensemble methods are popular in overcoming instability and increasing performance in machine learning tasks

- ***Our Goal:*** develop an intelligent file scoring system (***IFSS*** for short) for malware detection from the gray list by an ensemble of the ***heterogeneous*** classifiers using ***diverse*** feature representations on ***dynamic*** training sets

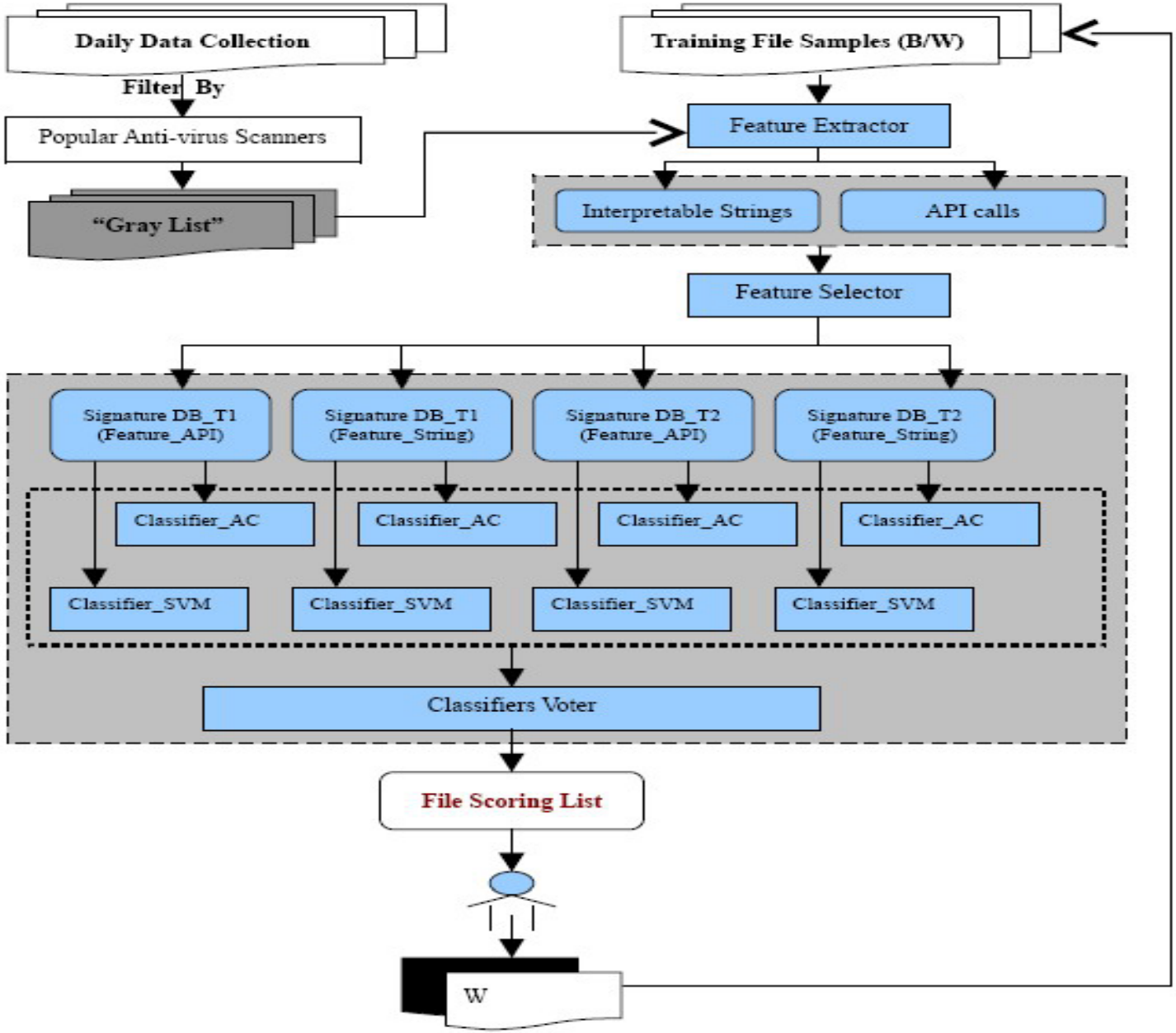
- Our ***case studies*** on large and real data collections collected by the Anti-virus Lab of Kingsoft corporation demonstrate the usefulness of IFSS

  - IFSS has already been incorporated into the scanning tool of Kingsoft's anti-virus software

Talk Outline

- Introduction and Motivation
- ***System Architecture and Description***
- Experimental Results and Case Studies

System Architecture



Malware Detection Procedure: Training

- ***Feature extraction***: Extracting the API calls and interpretable strings from the collected Windows Portable Executable (PE) files of blacklist and white list

- ***Feature selection***: Identifying the most representative features

- ***Base classifiers construction***: 8 base classifiers
  - Different classifier models
    - SVM
    - Associative classifiers
  - Different feature representation
    - API calls
    - Interpretable strings
  - Different training sets
    - T1: consists of file samples from historical data collection
    - T2: contains most recent file samples

- ***Scoring:*** A simple voting scheme is used to combine base classifiers.

Malware Detection Procedure: Detecting

- ***The gray list generation***: The daily collection is first scanned by the existing popular anti-virus software products

- ***Individual classification***: After feature extraction and selection, 8 different classifiers are applied to the gray list

- ***File scoring list***

  - A simple voting scheme is used to combine base classifiers and generate a file scoring list.

  - The file score list ranks the input file samples from the gray list.

- ***Human-in-the-Loop***:

  - Virus analysts can then look at the top ranked file samples and manually authenticated and rejected those samples.

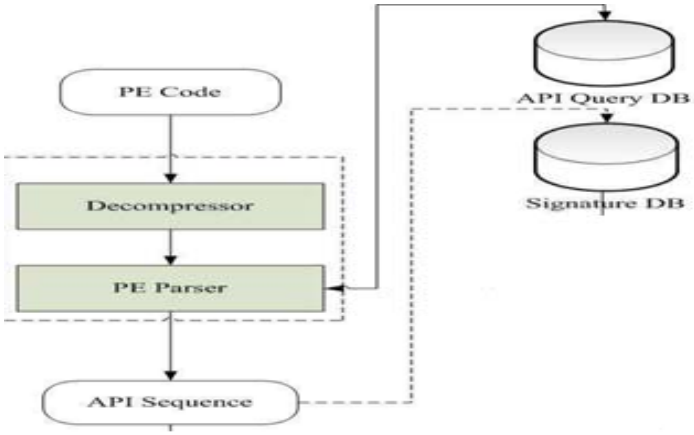
  - The manually labeled file samples can then be used to update the training sets.

Characteristics of IFSS

- Diverse feature representation
- Dynamic training sets
- Heterogeneous Classifiers
- Human-in-the-Loop
- Simultaneous model construction and testing

Feature Extraction and Selection: API Calls

- Our IFSS system is performed on windows PE code
- Feature Extraction
  - Flowchart

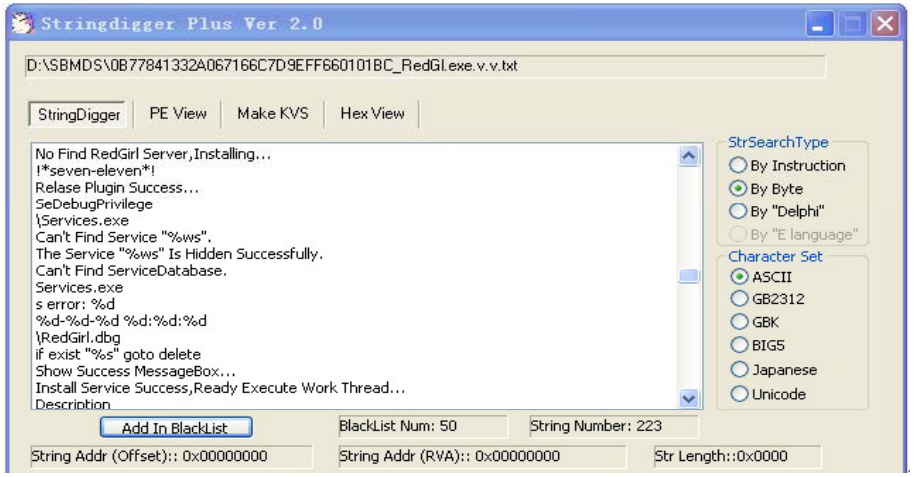


  - The API “KERNEL32.DLL, OpenProcess" executes the function that returns a handle to an existing process object and it can be encoded as 0x00500E16.

- Feature Selection
  - Not all of the API calls are contributing to malware detection
  - Rank each API call using Max-relevance algorithm and select a set of API class with the highest relevance to the target class

Feature Extraction and Selection: Interpretable Strings

- Our IFSS system is performed on windows PE code
- Feature Extraction
  - Extracted using a feature parser



- Feature Selection
  - First use the corpus of natural language to filter the candidate interpretable strings.
    - If the string consists most of the unusual characters which are not in the corpus, like "!0&0h0m0o0t0y0", it will be pruned by our feature parser.
  - Then apply Max-Relevance algorithm to select a set of the most representative strings

Feature Extraction Examples

➢One string shared by many malware samples

Trojan.Down.Winlagons
Win32.Troj.SysJunk2.ak
Trojan.Sleeper.a
Rootkit.Junk.a

Id	StrName
50369	winlogon.exe
50391	.EXE
50456	iexplore.exe

The following samples share similar features:

Backdoor.Gpigeon.GEN
Backdoor.Win32.Gpigeon2007.jlp
Backdoor.Win32.Gpigeon2008.bm
Backdoor.Win32.Hupigon.cyct
Backdoor.Win32.Hupigon.dkrr
Backdoor.Win32.Undef.az
Suspicious.Backdoor.Win32.Delgen.a
Suspicious.Backdoor.Win32.Gpigeon.a
Win32.Troj.IAgent.mv

Id	APIName	Max_Relevance	cc
1	user32.dll,createwindowexa;	0.007712	-13.361062
20	wsock32.dll,wsacleanup;	0.006091	11.191077
30	wininet.dll,internetreadfile;	0.005020	10.610623
38	user32.dll,getkeyboardtype;	0.009696	14.595705
59	kernel32.dll,raiseexception;	0.000079	-1.356956
61	advapi32.dll,regqueryvalueexa;	0.001218	-5.338272
79	kernel32.dll,loadlibrarya;	0.000811	-4.353012
81	kernel32.dll,getprocaddress;	0.002727	-7.963321
121	advapi32.dll,startservicea;	0.003825	9.297204
143	shell32.dll,shellexecutea;	0.001028	-4.901050
260	kernel32.dll,getmodulehandlea;	0.005849	-11.624446
327	urlmon.dll,urldownloadtofilea;	0.002333	7.226152
380	netapi32.dll,netbios;	0.000430	3.144448
382	advapi32.dll,setsecurityinfo;	0.000745	4.105960
885	advapi32.dll,reporteventa;	0.000258	-2.449434
3509	ws2_32.dll,gethostname;	0.000408	2.853488

Base Classifiers

- Associative Classification
  - Built on rules with high support and confidence
  - Example:
    (Kernel32.dll,*OpenProcess*;*CopyFileA*;*CloseHandle*;*GetVersionExA*;*GetModuleFileNameA*;*WriteFile*) $\rightarrow$ Obj = (*Group* = *Malicious*) (os = 0.29, oc = 0.99)
  - Post-processing for associative classifier construction
    - Rule Pruning
    - Rule Re-ordering
- Support Vector Machine (SVM)
  - Seeking a hyperplane which maximize the margin between the two different classes
  - Applying kernel functions to map the data into a high dimensional space
- Both classification methods have been used in malware detection

Some Examples of Associative Classifier

Rule1:(50455,50454,50450,50048)-> filesort = malware (os=0.011817, oc=1)

id	strname	riskvalue	virusname
50048	SeDebugPrivilege	5	Win32.Troj.Inject.pe
50450	readprocessmemory	5	
50454	internet exploreriexplore.exe	5	
50455	:program filesinternet exploreriexplore.exe	5	

Rule2:(50455,50454,50450,50048)-> filesort = benign (os=0.00511, oc=1)

id	strname	riskvalue	virusname
50288	.htm	5	Worm.AutoRuns.sy
50301	ControlSet001	15	Win32.Hack.PcClient.c
50302	ControlSet002	20	Win32.Hack.PcClient.c
50341	Windows NT	5	Win32.Hack.Agent.ge

Rule3:(85,38,20)-> filesort = malware (os=0.00511, oc=1)

Id	APIName	Max_Relevance	cc
20	wsock32.dll,wsacleanup;	0.006091	11.191077
38	user32.dll,getkeyboardtype;	0.009696	14.595705
85	kernel32.dll,getcurrentprocessid;	0.000003	0.271554

Ensemble Classifier

- Base classifiers are constructed by applying associative classifier and SVM using different feature representations on different training sets
- There are total 8 different base classifiers
- A simple voting scheme is used to combine base classifiers.
  - For an input file, each base classifier casts a vote for its prediction: i.e., 1 if the input file is predicted to be malicious and 0 otherwise.
  - After classifier voters, each file sample can obtain a score ranging from 8 to 0.
  - If two file sample have the same score, they will be ranked by their matching association classification rules' s chi-square values in descending order
- IFSS system then generates a file scoring list which is a ranked list of all input file samples from the gray list.
  - simple for virus analysts to interpret and understand

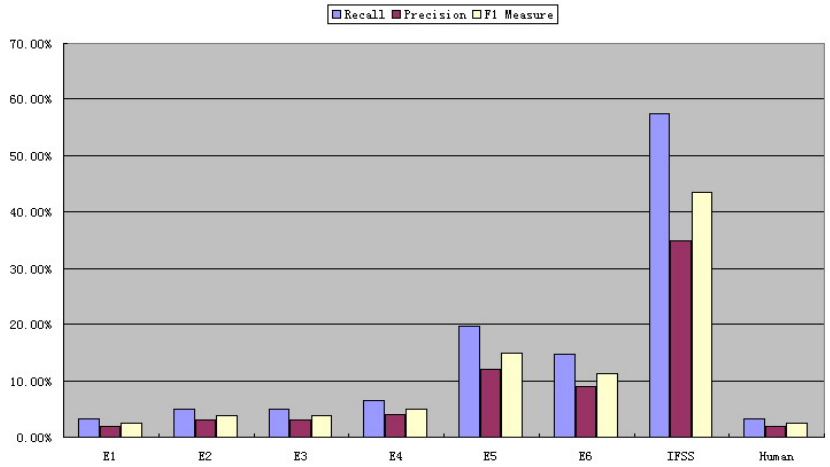
Talk Outline

- Introduction and Motivation
- System Architecture and Description
- ***Experimental Results and Case Studies***

Comparison of Different Classification Methods

- Data descriptions
  - Features
    - 7909 API calls and 32,123 interpretable strings
  - Training sets
    - Training set T1: Consisting of 491,733 historical PE file samples
    - Training set T2: Containing 530,448 PE files of the week from Jan. 1st, 2009 to Jan. 7th, 2009
  - Test set
    - A 10% random sample from the gray list of Jan. 8, 2009
    - Contains 12,365 files, 61 of which are malicious
- Experiment Design: select the top 100 files from the rank list generated by each ensemble and evaluate the performances of different ensembles. Our virus analysts also select 100 files from the gray list to analyze.
- ***IFSS our performs other ensembles as well as human experts***

Ensemble	TP	Recall	Precision	F1
E1:C1+C2	2	3.28%	2%	0.0248
E2:C3+C4	3	4.92%	3%	0.0373
E3:C5+C6	3	4.92%	3%	0.0373
E4:C7+C8	4	6.56%	4%	0.0497
E5:C1+C2+C3+C4	12	19.67%	12%	0.1491
E6:C5+C6+C7+C8	9	14.75%	9%	0.1118
IFSS:C1-C8	**35**	**57.38%**	**35%**	**0.4348**
Human Expert	2	3.28%	2%	0.0248

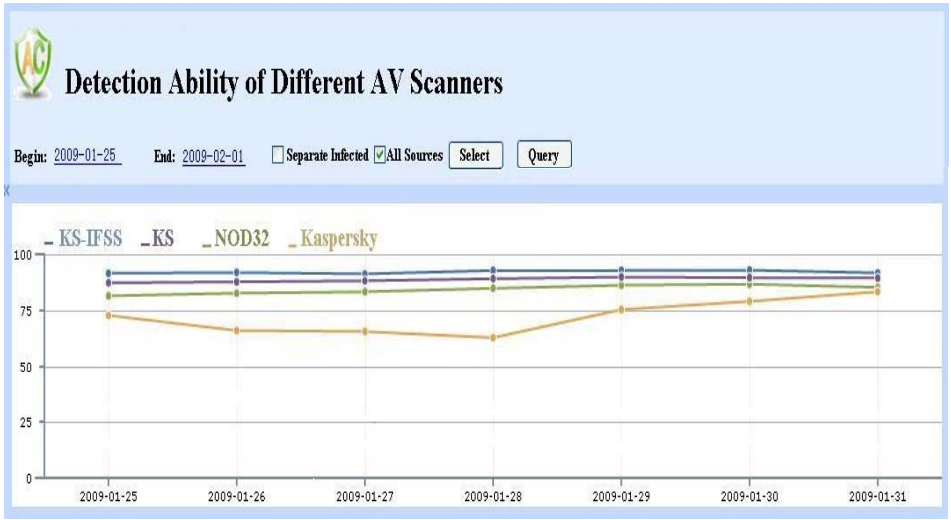


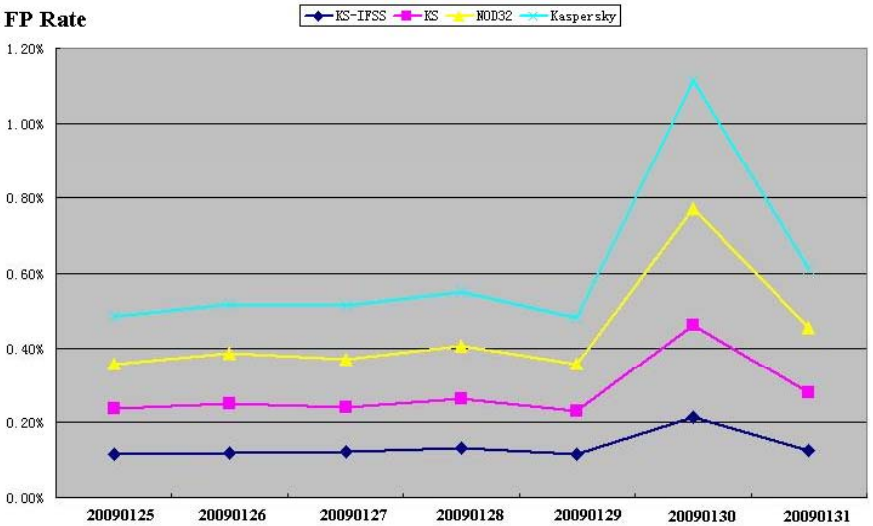
Detection Ability of KS-IFSS

- KS-IFSS: IFSS has been incorporated into Kingsoft's anti-virus software (KS)
- Data descriptions
  - Daily data collection for the week of Jan. 25th, 2009 and Jan 31, 2009

Day	Date	All Files	Malware	Benign Files
1	2009/01/25	407,882	42,608	51,595
2	2009/01/26	516,715	44,204	59,245
3	2009/01/27	551,120	44,813	50,297
4	2009/01/28	597,767	47,796	38,982
5	2009/01/29	312,372	49,077	65,113
6	2009/01/30	520,761	57,144	66,022
7	2009/01/31	705,620	55,523	54,489
	Sum	3,612,237	341,165	385,723

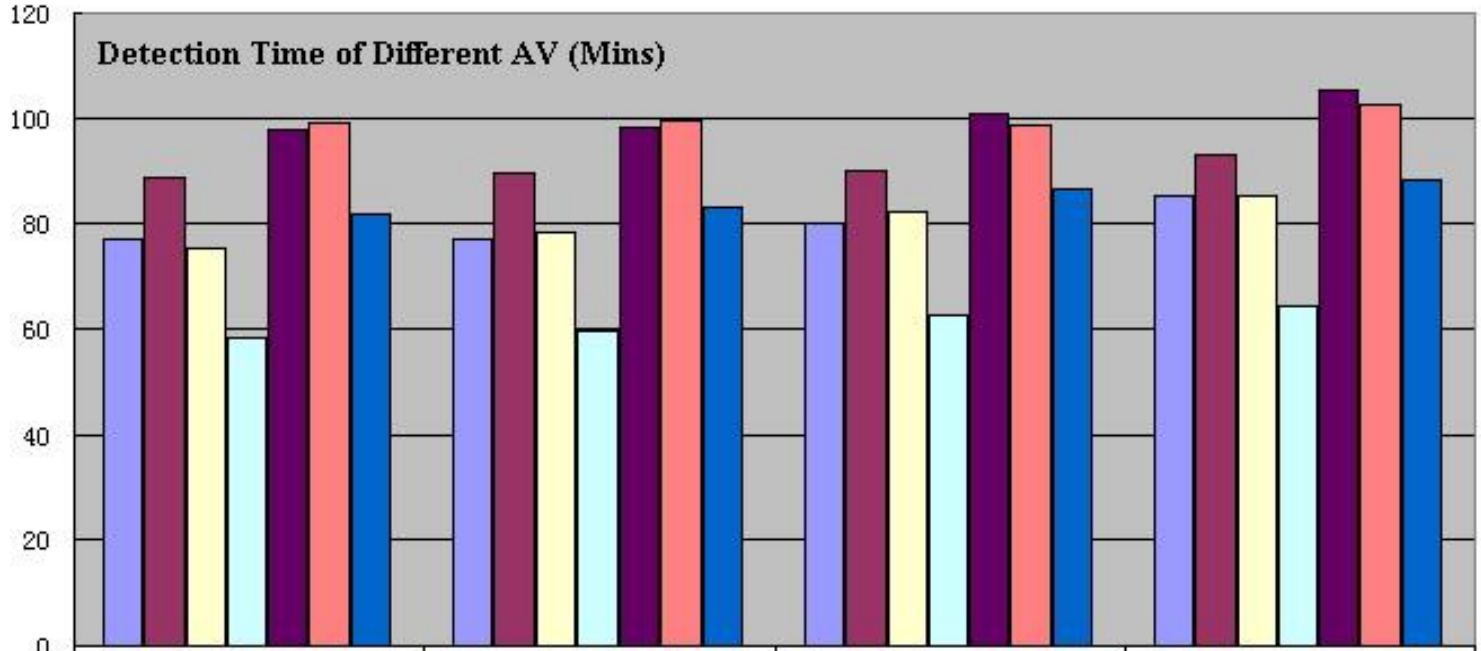
- Experiment comparison: compare KS-IFSS with NOD32, Kaspersky and KS. ***KS-IFSS outperforms other anti-virus scanners.***





Detection Efficiency of KS-IFSS

- KS-IFSS achieves high efficiency than other scanners



	KS-IFSS	KS	NOD32	Kaspersky
Day1	77.12	77.18	80.15	85.2
Day2	88.87	89.71	90.12	93.2
Day3	75.45	78.32	82.5	85.5
Day4	58.47	59.5	62.38	64.36
Day5	97.67	98	100.5	105.5
Day6	99	99.5	98.5	102.5
Day7	81.74	83.2	86.5	88.45

Conclusion

- Summary
  - IFSS uses an ensemble framework and it has several favorable traits including diverse feature representations, dynamic training sets, heterogeneous base classifiers and human-in-the-Loop, simultaneous model construction and testing.
  - The case studies on large data collections on the the gray list and real daily data collection obtained from the Anti-virus Lab of Kingsoft corporation demonstrate the effectiveness and efficiency of our IFSS system.
  - IFSS is a practical solution for virus analysts to identify malware samples from the huge gray list and has already been incorporated into the scanning tool of Kingsoft's Anti-Virus software.

Q & A ?

- Email: taoli@cs.fiu.edu
- http://www.cs.fiu.edu/~taoli