Automatic Malware Categorization Using Cluster Ensemble

Yanfang Ye: Xiamen University & Internet Security R&D Center, Kingsoft Corporation
Tao Li: School of Computer Science, Florida International University
Yong Chen: Internet Security R&D Center, Kingsoft Corporation
Qiangshan Jiang: Xiamen University
Talk Outline

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

➢ Huge number of malware samples

➢ "Black Chain" is driven by the economic benefit

➢ Defense against malware

➢ Effective methods for automatic malware categorization is in need
The process of automatic malware categorization:

1. Feature Extraction
   - Application Programming Interface (API) calls
   - Instruction sequences
   - Program behaviors
   - ……

2. Categorization
   - Hierarchical Clustering
   - K-medoids
   - ……

Limitation of current automatic malware categorization methods:

1. Limited effectiveness and efficiency
2. Few have been applied in real anti-malware industry
Cluster Ensemble methods are popular in overcoming instability and increasing performance in machine learning tasks.

Our goal is developing an Automatic Malware Categorization System (AMCS for short) for automatically grouping malware samples into families that share some common characteristics using a cluster ensemble by aggregating the clustering solutions generated by different base clustering algorithms, while the domain knowledge in the form of sample-level constraints can be naturally incorporated in the ensemble framework.

Our case studies on large and real data collections collected by the Anti-malware Lab of Kingsoft corporation demonstrate the usefulness of AMCS, which has already been incorporated into the scanning tool of Kingsoft’s anti-malware software.
Talk Outline

- Introduction and Motivation
- System Architecture and Description
- Experimental Results and Case Studies
Hybrid Hierarchical Clustering algorithm with the tf-idf & tf weighting schemes

AdWare.Win32.Nodef.kp_7409 edaae06e6763dd23c93101ff4662

sub_10001000:
mov, dec, jnz, mov, mov, mov, retn;
sub_100010e0:
sub, push, mov, or, xor, ..., ret;

Weighted K-medoids Partitioned algorithm
Characteristics of AMCS

- Well-Chosen Feature Representations
- Carefully-Designed Base Clusterings
- A Principled Cluster Ensemble Scheme
- Natural Application for Signature Generation
- Human-in-the-loop
## Feature Representations

### The extraction of instruction frequency

```plaintext
Function 1
proc near
push ebx
mov ecx, ecx
mov ebx, ebx
mov ecx, ecx
call sub_401BCD
xor eax, eax
......

Function 2
proc near
push ebx
mov ebx, ebx
mov ecx, ecx
mov ecx, ecx
push offset aKernel32 : “Kernel32”
call dword_4010A0
add eax, 1
......
```

- **Remove operands**

  ```plaintext
  push, mov, mov, mov, call, xor,......
  push, mov, mov, push, call, add,......
  ```

- **Query from the database**

  ```plaintext
  Instruction Query DB
  (Id) (Instruction)
  1 push
  2 mov
  3 call
  4 xor
  5 add
  ......
  ```

- **Compose to a instruction frequency vector**

  ```plaintext
  1:1, 2:3, 3:1, 4:1, ......
  1:2, 2:2, 3:1, 5:1, ......
  ```

### The extraction of function-based 5-gram instruction sequence segments

```plaintext
Function 1
proc near
push ebx
mov ecx, ecx
mov ebx, ebx
mov ecx, ecx
mov ecx, ecx
call sub_401BCD
xor eax, eax
......

Function 2
proc near
push ebx
mov ecx, ebx
mov ecx, ecx
mov ecx, ecx
push offset aKernel32 : “Kernel32”
call dword_4010A0
add eax, 1
......
```

- **Remove operands**

  ```plaintext
  push, mov, mov, mov, call, xor,......
  push, mov, mov, push, call, add,......
  ```

- **Find and Register Sequences in Hash Map**

  ```plaintext
  Hash Map of Sequences
  (Id) (Sequence)
  1 push, mov, mov, mov, call
  2 mov, mov, mov, call, xor
  3 push, mov, mov, push, call
  4 mov, mov, mov, call, add
  ......
  ```

- **Compose to a function-based instruction sequence vector**

  ```plaintext
  1, 2, 3, 4, ......
  1:3, 2:5, 3:2, 4:1, 5:1, ......
  ``
Advantages of the Feature Representations for Malware Categorization

- Great ability for representing variants of a malware family
- Easy for generating signatures for a malware family to detect its variants
- High coverage rate of malware samples
- Semantic Implications
- High efficiency for feature extraction
Base Clusterings ---- Hybrid Hierarchical Clustering (HHC)

DB: 1,434 malware with 1,222 dimensions

2 Dimensions transformed by PCA

3 Dimensions transformed by PCA

The algorithm description of HHC

\[
FS = \sum_{i=1}^{N} \sum_{j=1}^{nc} u_{ij}^{m} (\|x_i - v_j\|_{A}^2 - \|v_j - v\|_{A}^2)
\]

The Fukuyama-Sugeno index (FS)
Input: $N$ points in $d$-dimensional space, number of clusters $k$
Output: $k$ clusters and the corresponding weight vector

Randomly choose $k$ cluster medoids;
set initial weights to be $\frac{1}{d}$;
repeat
| Assign each points to the nearest cluster;
| Update the cluster medoids;
| Update the weight vector using Eq.(1);
| Calculate the validity index;
until the weight vectors and the medoids do not change;

Alg2. The algorithm description of WKM

Eq. (1) The weight of the $j$-th feature for cluster $i$
Cluster Ensemble: aggregate the clustering solutions generated by different both hierarchical and partitioned clustering algorithms.
Cluster ensemble incorporating sample-level constraints

- Define the connectivity matrix $M(P^T)$ for the partition $P'$ as:

$$M_{ij}(P^T) = \begin{cases} 1 & \text{if } x_i \text{ and } x_j \text{ belong to same cluster } C^t \\ 0 & \text{else} \end{cases}$$

$$\tilde{M}_{ij} = \frac{1}{T} \sum_{t=1}^{T} M_{ij}(P^t)$$

- Incorporate sample-level constraints:

1. **Must-link constraints:** $A = \{(x_{i1}, x_{j1}), \ldots, (x_{ia}, x_{ja})\}, a = |A|$

2. **Cannot-link constraints:** $B = \{(x_{p1}, x_{q1}), \ldots, (x_{pb}, x_{qb})\}, b = |B|$

- A convex optimization problem with linear constraints:

$$\min_{P'} J = \frac{1}{T} \sum_{t=1}^{T} \sum_{i,j=1}^{n} [M_{ij}(P^t) - M_{ij}(P^*)]^2$$

s.t.

$$M_{ij}(P^*) = 1, \text{ if } (x_i, x_j) \in A$$

$$M_{ij}(P^*) = 0, \text{ if } (x_i, x_j) \in B$$

$$M_{i,j_s}(P^*) = \begin{cases} \frac{1}{T} \sum_{t=1}^{T} M_{ij}(P^t) & \text{if } (i_s, j_s) \text{ not in } C \\ b_s & \text{else} \end{cases}$$
Talk Outline

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

1. Comparisons of Clustering Methods Based on Instruction Frequency
2. Comparisons of Clustering Methods Based on Instruction Sequences
3. Evaluation of Cluster Ensemble with Constraints
4. Comparisons with Different AV Venders
(1) Comparisons of Malware Clustering Methods Based on Instruction Frequency

- Measures:
  
  \[
  \text{Macro} - F1 = \frac{\sum_{i=1}^{C} F_{1i}}{n} \quad \text{Micro} - F1 = \frac{\sum_{i=1}^{C} 2 \cdot \text{Recall}_i \cdot \text{Precision}_i}{\sum_{i=1}^{C} \text{Recall}_i + \text{Precision}_i} \quad F_1 = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}
  \]

<table>
<thead>
<tr>
<th>Day</th>
<th>Num</th>
<th>D</th>
<th>F</th>
<th>Alg</th>
<th>Macro</th>
<th>Micro</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3546</td>
<td>1226</td>
<td>88</td>
<td>KM_TFIDF</td>
<td>0.6925</td>
<td>0.7376</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>1226</td>
<td>88</td>
<td>HC_TFIDF</td>
<td>0.7501</td>
<td>0.7134</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>1226</td>
<td>88</td>
<td>HHC_TFIDF</td>
<td>0.8218</td>
<td>0.8015</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>1226</td>
<td>88</td>
<td>KM_TF</td>
<td>0.6279</td>
<td>0.6802</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>1226</td>
<td>88</td>
<td>HC_TF</td>
<td>0.7228</td>
<td>0.7237</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>1226</td>
<td>88</td>
<td>HHC_TF</td>
<td>0.8162</td>
<td>0.8128</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>1178</td>
<td>102</td>
<td>KM_TFIDF</td>
<td>0.7033</td>
<td>0.7661</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>1178</td>
<td>102</td>
<td>HC_TFIDF</td>
<td>0.787</td>
<td>0.7921</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>1178</td>
<td>102</td>
<td>HHC_TFIDF</td>
<td>0.8263</td>
<td>0.8101</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>1178</td>
<td>102</td>
<td>KM_TF</td>
<td>0.5687</td>
<td>0.602</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>1178</td>
<td>102</td>
<td>HC_TF</td>
<td>0.6605</td>
<td>0.6845</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>1178</td>
<td>102</td>
<td>HHC_TF</td>
<td>0.7655</td>
<td>0.7468</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>2375</td>
<td>324</td>
<td>KM_TFIDF</td>
<td>0.5942</td>
<td>0.5905</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>2375</td>
<td>324</td>
<td>HC_TFIDF</td>
<td>0.6365</td>
<td>0.6591</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>2375</td>
<td>324</td>
<td>HHC_TFIDF</td>
<td>0.722</td>
<td>0.7335</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>2375</td>
<td>324</td>
<td>KM_TF</td>
<td>0.6398</td>
<td>0.6126</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>2375</td>
<td>324</td>
<td>HC_TF</td>
<td>0.7436</td>
<td>0.7228</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>2375</td>
<td>324</td>
<td>HHC_TF</td>
<td>0.7895</td>
<td>0.7957</td>
</tr>
</tbody>
</table>

Table 1: Based on instruction frequency, the categorization results of different categorizers on the real daily new malware collection from Jan 11th, 2010 to Jan 13th, 2010.  
Remark: "Num"-the total number of the malware samples, "D"-Dimensions of the data set, "F"-the real malware families, "Macro"-Macro-F1 measure and "Micro"-Micro-F1 measure.
(2) Comparisons of Malware Clustering Methods Based on Instruction Sequences

• Measures:

\[
Macro - F1 = \frac{\sum_{i=1}^{C} F_{ii}}{n}
\]

\[
Micro - F1 = \frac{\sum_{i=1}^{C} 2 \cdot \text{Recall}_i \cdot \text{Precision}_i}{\sum_{i=1}^{C} \text{Recall}_i + \text{Precision}_i}
\]

\[
F_1 = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}
\]

<table>
<thead>
<tr>
<th>Day</th>
<th>Num</th>
<th>D</th>
<th>F</th>
<th>Alg</th>
<th>Macro</th>
<th>Micro</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3546</td>
<td>7208</td>
<td>88</td>
<td>KM</td>
<td>0.6135</td>
<td>0.6747</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>7208</td>
<td>88</td>
<td>WKM</td>
<td>0.8196</td>
<td>0.8559</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>6923</td>
<td>102</td>
<td>KM</td>
<td>0.6882</td>
<td>0.6421</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>6923</td>
<td>102</td>
<td>WKM</td>
<td>0.8071</td>
<td>0.8015</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>11054</td>
<td>324</td>
<td>KM</td>
<td>0.6279</td>
<td>0.6252</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>11054</td>
<td>324</td>
<td>WKM</td>
<td>0.7874</td>
<td>0.8147</td>
</tr>
</tbody>
</table>

Table 2: Based on function-based instruction sequences, the categorization results of different categorizers on the real daily new malware collection from Jan 11th, 2010 to Jan 13th, 2010.
(3) Evaluation of Cluster Ensemble with Constraints

• Effectiveness

<table>
<thead>
<tr>
<th>Day</th>
<th>Num</th>
<th>F</th>
<th>Alg</th>
<th>Macro</th>
<th>Micro</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3546</td>
<td>88</td>
<td>HHC_TFIDF</td>
<td>0.8218</td>
<td>0.8015</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>88</td>
<td>HHC_TF</td>
<td>0.8162</td>
<td>0.8128</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>88</td>
<td>WKM</td>
<td>0.8196</td>
<td>0.8559</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>88</td>
<td>NCE</td>
<td>0.9017</td>
<td>0.9137</td>
</tr>
<tr>
<td>1</td>
<td>3546</td>
<td>88</td>
<td>CE</td>
<td>0.9302</td>
<td>0.9437</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>102</td>
<td>HHC_TFIDF</td>
<td>0.8263</td>
<td>0.8101</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>102</td>
<td>HHC_TF</td>
<td>0.7655</td>
<td>0.7468</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>102</td>
<td>WKM</td>
<td>0.8071</td>
<td>0.8015</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>102</td>
<td>NCE</td>
<td>0.8989</td>
<td>0.8669</td>
</tr>
<tr>
<td>2</td>
<td>3005</td>
<td>102</td>
<td>CE</td>
<td>0.9245</td>
<td>0.9113</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>324</td>
<td>HHC_TFIDF</td>
<td>0.722</td>
<td>0.7335</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>324</td>
<td>HHC_TF</td>
<td>0.7895</td>
<td>0.7957</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>324</td>
<td>WKM</td>
<td>0.7874</td>
<td>0.8147</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>324</td>
<td>NCE</td>
<td>0.8605</td>
<td>0.8896</td>
</tr>
<tr>
<td>3</td>
<td>5162</td>
<td>324</td>
<td>CE</td>
<td>0.9183</td>
<td>0.9181</td>
</tr>
</tbody>
</table>

Table 2: Evaluation of the malware categorization results of clustering ensemble.

Remark: "NCE"-cluster ensemble without constraints, "CE"-cluster ensemble with constraints.

Example of sample-level Cannot-link constraints

Comparisons of malware categorization results of cluster ensembles without and with constraints
(4) Comparisons with Different AV Vendors

• Effectiveness

The categorization results of different AV software on the whole data collection of 42,180 malware samples.

<table>
<thead>
<tr>
<th>AV</th>
<th>Detected</th>
<th>Families</th>
<th>MacroF1</th>
<th>MicroF1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kasp</td>
<td>35,433</td>
<td>1,998</td>
<td>0.6736</td>
<td>0.6246</td>
</tr>
<tr>
<td>Nod32</td>
<td>33,634</td>
<td>1,598</td>
<td>0.6498</td>
<td>0.7233</td>
</tr>
<tr>
<td>Mcafee</td>
<td>35,500</td>
<td>1,859</td>
<td>0.6253</td>
<td>0.6856</td>
</tr>
<tr>
<td>BD</td>
<td>39,723</td>
<td>1,916</td>
<td>0.7215</td>
<td>0.7761</td>
</tr>
<tr>
<td>Rising</td>
<td>35,712</td>
<td>1,885</td>
<td>0.5983</td>
<td>0.6257</td>
</tr>
<tr>
<td>AMCS</td>
<td>41,623</td>
<td>2,271</td>
<td>0.9274</td>
<td>0.9369</td>
</tr>
</tbody>
</table>

• Efficiency

(1) Categorizing 3,546 malware samples by our AMCS system including feature extraction needs 3 minutes;

(2) The whole process of 42,180 malware samples needs 15 minutes.
(4) Automatic Signature Generation

- **Signature**: the instruction sequences (with operands) frequently appeared within a malware family but rarely appeared in other families.

The signature with id “72237142”:

- [ScriptScan]
  - mov Reg00, 0x00
  - mov Reg01, 0x01
  - mov Reg02, 0x02
  - mov Reg03, 0x03
  - mov Reg04, 0x04
  - Call PE.GetImageBase Reg05
  - ReturnIfFalse

- SetMatchBuffer NormalEntryBufferReg, NormalEntryBufferSizeReg
- mov SearchPosReg, 0x59
- mov SearchLenReg, 0x20
- mov Reg06, 0x35FF
- SearchWORD Reg06

...
System development and operation

- **Expense:** Kingsoft has spent over $500K in the development of the AMCS system and about $100K on the hardware equipment.

- **Usefulness:** Over 30 virus analysts at Kingsoft’s Anti-Virus lab are utilizing the system on the daily basis. In practice, a virus analyst has to spend at least 10 hours to manually analyze 100 malware samples for categorization. Using the AMCS system, the categorization of about 30,000 malware samples can be performed within 20 minutes. The high efficiency of our AMCS system can greatly save human labors and reduce the staff cost.

- **Benefited users:** This would benefit over 10 million Internet users of Kingsoft’s client anti-malware products.
The AV products of Kingsoft incorporated AMCS

The detection of malware using the signature generated by AMCS
Summary

- AMCS categorizes malware samples into families that share some common traits by an ensemble of different clustering solutions generated by different clustering methods.

- The domain knowledge in the form of sample-level constraints can be naturally incorporated in the ensemble framework.

- The case studies on large and real daily malware sample collections obtained from the Anti-malware Lab of Kingsoft corporation demonstrate the effectiveness and efficiency of our AMCS system.

- AMCS is a practical solution for automatic malware categorization from the huge malware sample collections and has already been incorporated into the scanning tool of Kingsoft's Anti-malware software.

Future Work

- Conduct further study to construct a more streamline signature library for better malware detection on the client anti-malware products based on our AMCS system.
• E-mail: yeyanfang@yahoo.com.cn