Detection of Malicious VBScript Using Static and Dynamic Analysis with Recurrent Deep Learning

Jack W. Stokes†, Rakshit Agrawal*, Geoff McDonald††

†Microsoft Research

**University of California, Santa Cruz*

††Microsoft Corporation

Outline

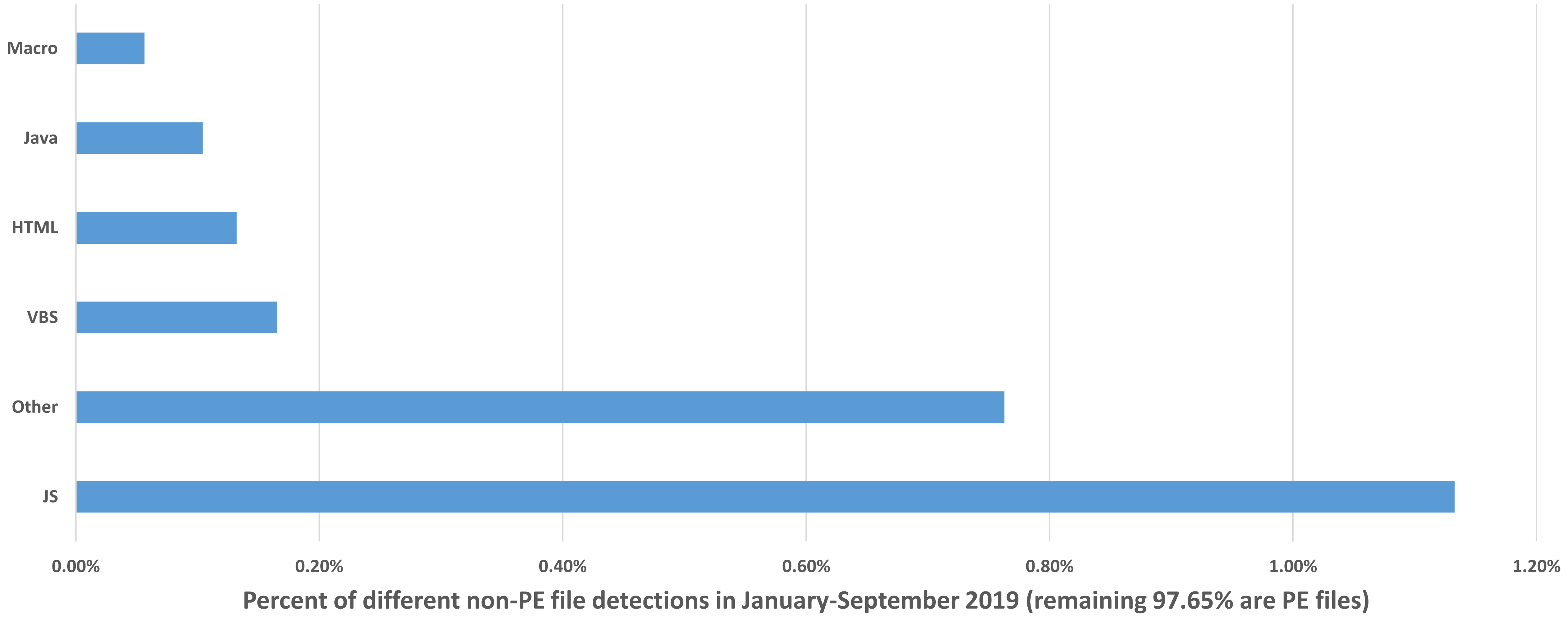
Motivation

System Design and Models

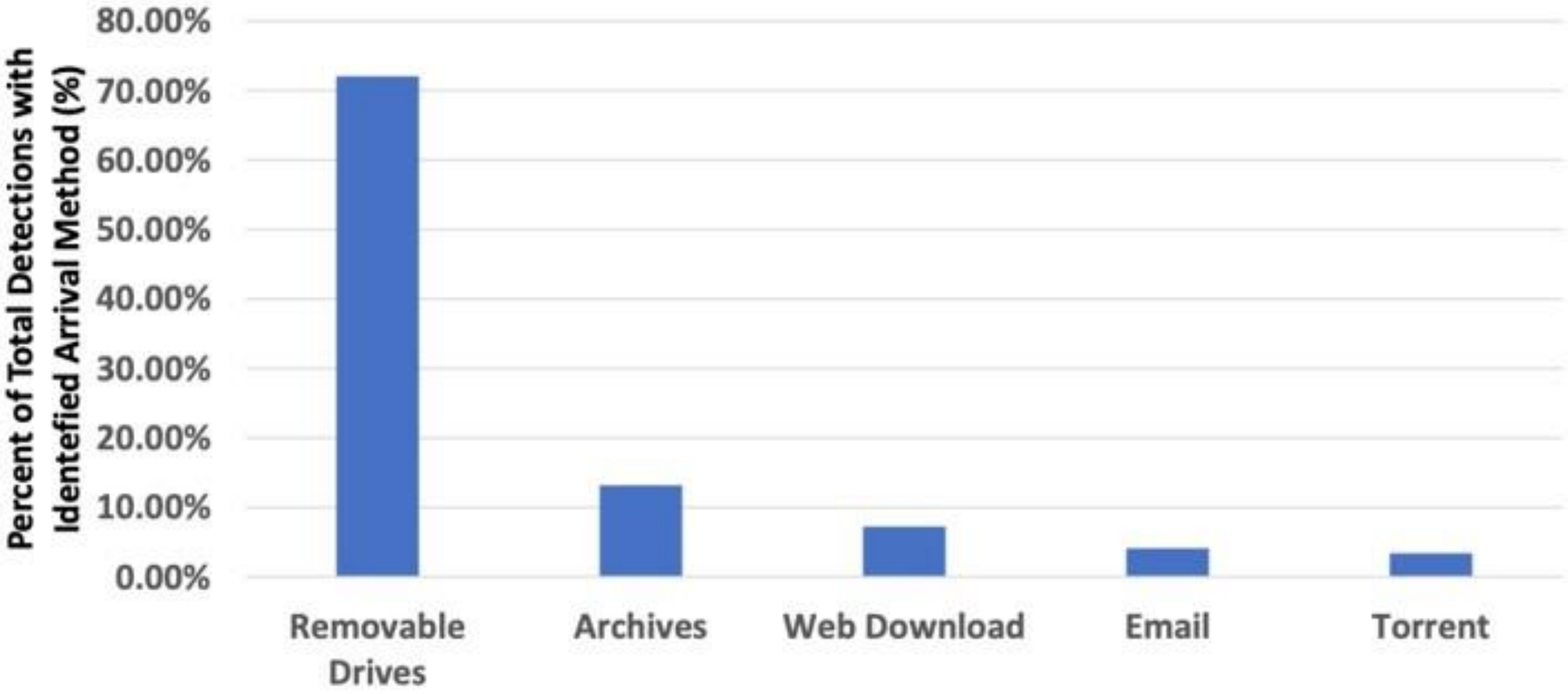
Performance Evaluation

Conclusions

Malicious VBScript Prevalence



Arrival Methods for Malicious VBScript Files

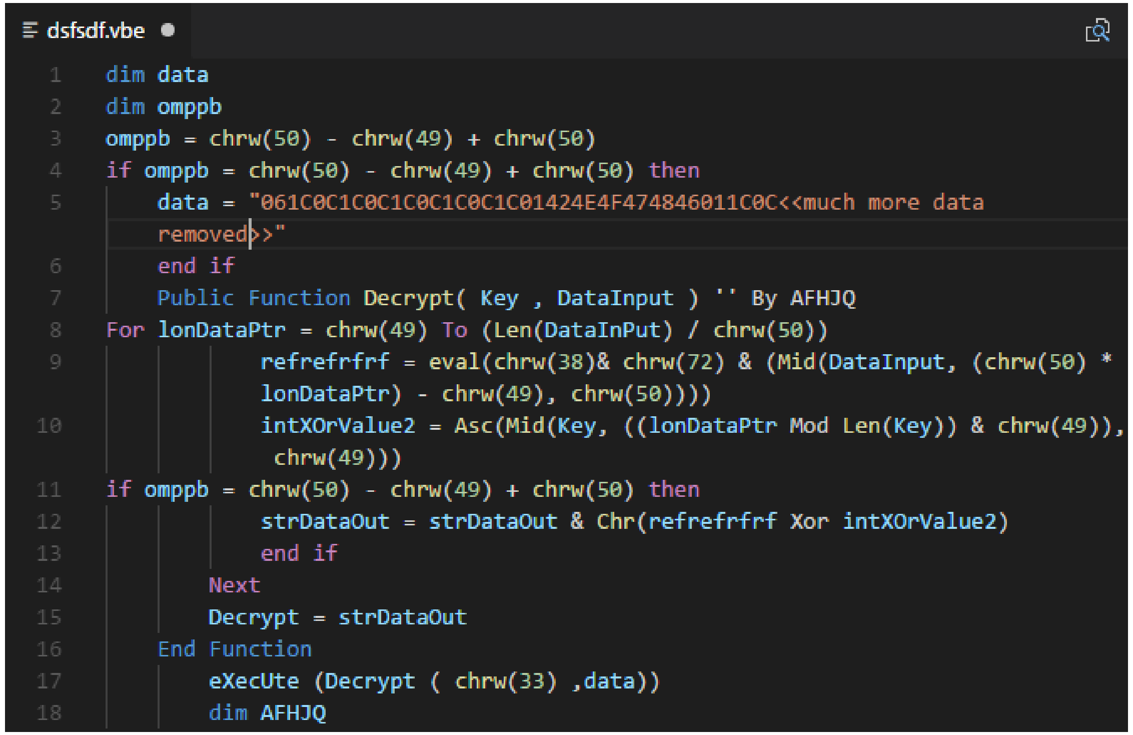


Detected from January through September 2019

Challenges - Obfuscation

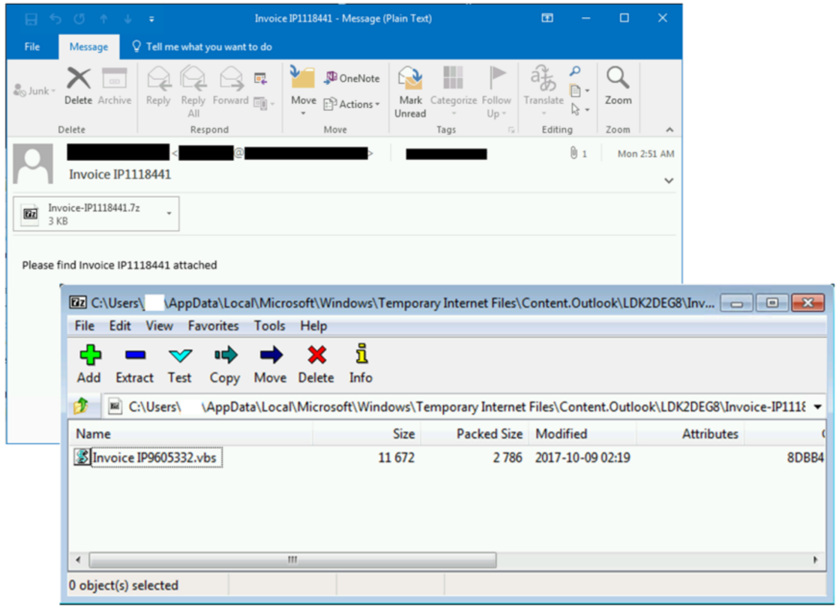
Attackers use obfuscation to hide/drop the malicious content

- Unpack or decrypt only upon execution
- Some obfuscators are used by both benign and malicious VBScript files
- Static analysis of the primary script often fails to detect some malicious activity



```
dim data
dim omppb
omppb = chrw(50) - chrw(49) + chrw(50)
if omppb = chrw(50) - chrw(49) + chrw(50) then
    data = "061C0C1C0C1C0C1C0C1C01424E4F474846011C0C<<much more data
    removed>>"
    end if
    Public Function Decrypt( Key , DataInput ) '' By AFHJQ
For lonDataPtr = chrw(49) To (Len(DataInPut) / chrw(50))
            refrefrfrf = eval(chrw(38)& chrw(72) & (Mid(DataInput, (chrw(50) *
            lonDataPtr) - chrw(49), chrw(50))))
            intXOrValue2 = Asc(Mid(Key, ((lonDataPtr Mod Len(Key)) & chrw(49)),
             chrw(49)))
if omppb = chrw(50) - chrw(49) + chrw(50) then
            strDataOut = strDataOut & Chr(refrefrfrf Xor intXOrValue2)
            end if
        Next
        Decrypt = strDataOut
    End Function
        eXecUte (Decrypt ( chrw(33) ,data))
        dim AFHJQ
```

Other Challenges



[redacted] <[redacted]@[redacted]> [redacted]

1 Mon 2:51 AM

Invoice IP1118441

Invoice-IP1118441.7z
3 KB

Please find Invoice IP1118441 attached

C:\Users\ \AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Outlook\LDK2DEG8\Inv...

C:\Users\ \AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Outlook\LDK2DEG8\Invoice-IP111ε

Name	Size	Packed Size	Modified	Attributes	C
Invoice IP9605332.vbs	11 672	2 786	2017-10-09 02:19		8DBB4

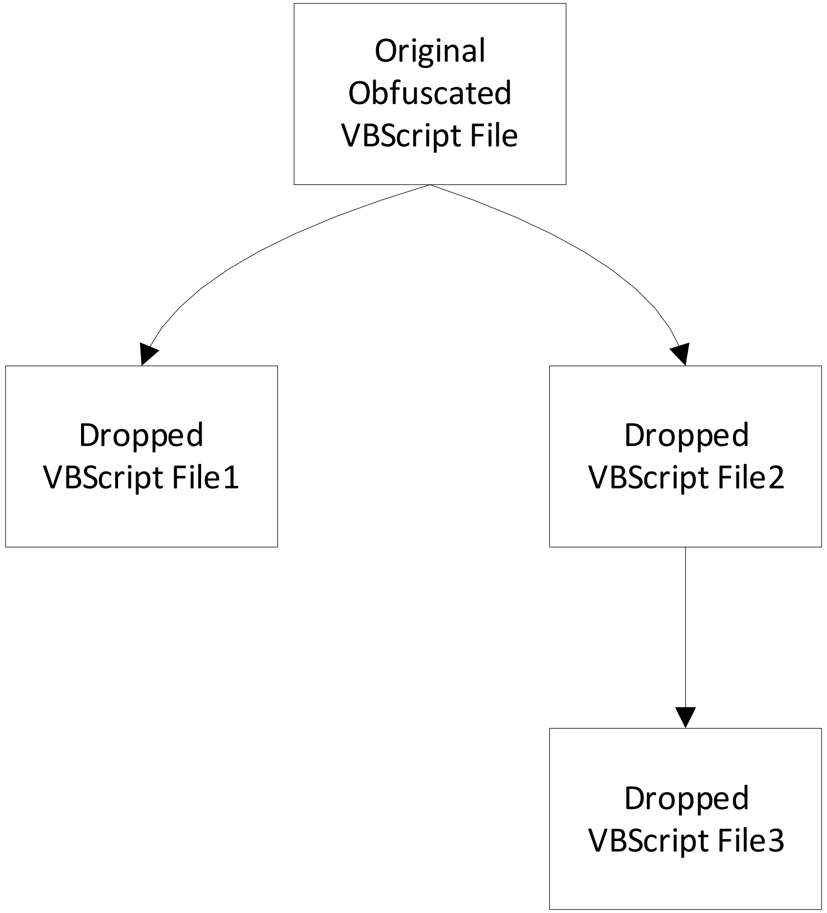
0 object(s) selected

AV automation systems such as sandboxing environments are designed primarily to handle Windows PE files (e.g., .exe and .dll)

AV analysts spend most of their time authoring new signatures for executable malware

- Number of labeled script files is typically much lower than for executable files

Obfuscation and File Dropping



Outline

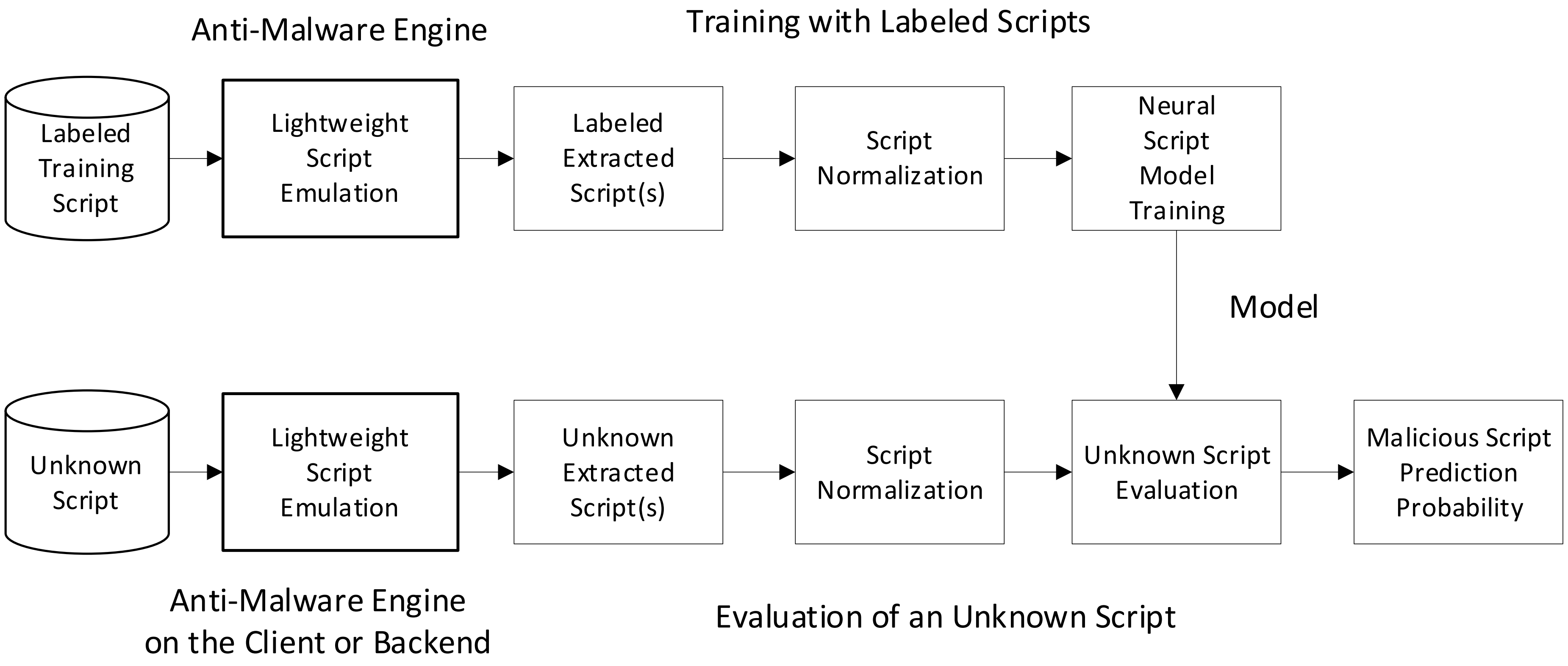
Motivation

System Design and Models

Performance Evaluation

Conclusions

Overview of the VbsNet Neural VBScript Classification System


Anti-Malware Engine
Training with Labeled Scripts
Labeled Training Script
Lightweight Script Emulation
Labeled Extracted Script(s)
Script Normalization
Neural Script Model Training
Model
Unknown Script
Lightweight Script Emulation
Unknown Extracted Script(s)
Script Normalization
Unknown Script Evaluation
Malicious Script Prediction Probability
Anti-Malware Engine on the Client or Backend
Evaluation of an Unknown Script

Script Normalization

First remove all whitespace characters except for line breaks

Text is standardized to lowercase and converted to the US-ASCII character set

All characters are next encoded by their numeric ASCII encoding (e.g., ’97’ for the character ’a’) delimited by commas to avoid storing malicious content on the hard drive

LaMP - LSTM and Max Pooling

Previous Work

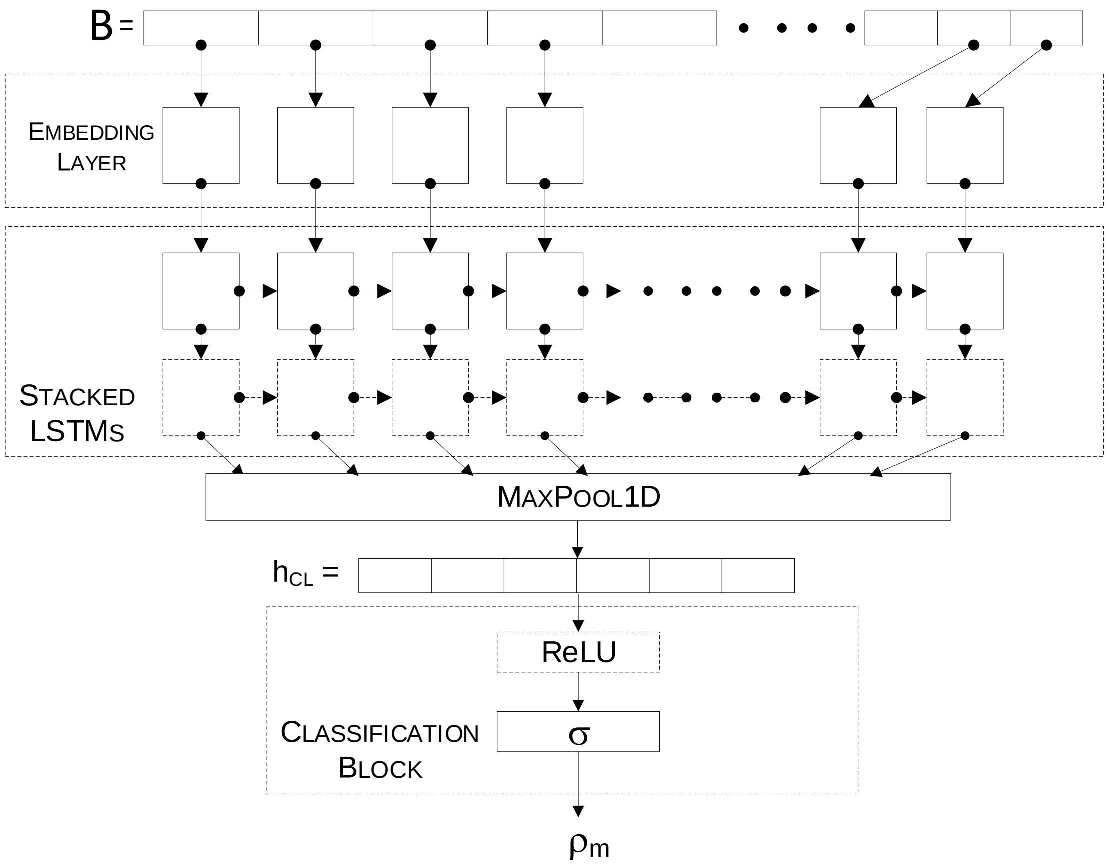
Athiwaratkun 2017

Agrawal 2018

PE Files

Adapted for VBScript

End-to-End Training



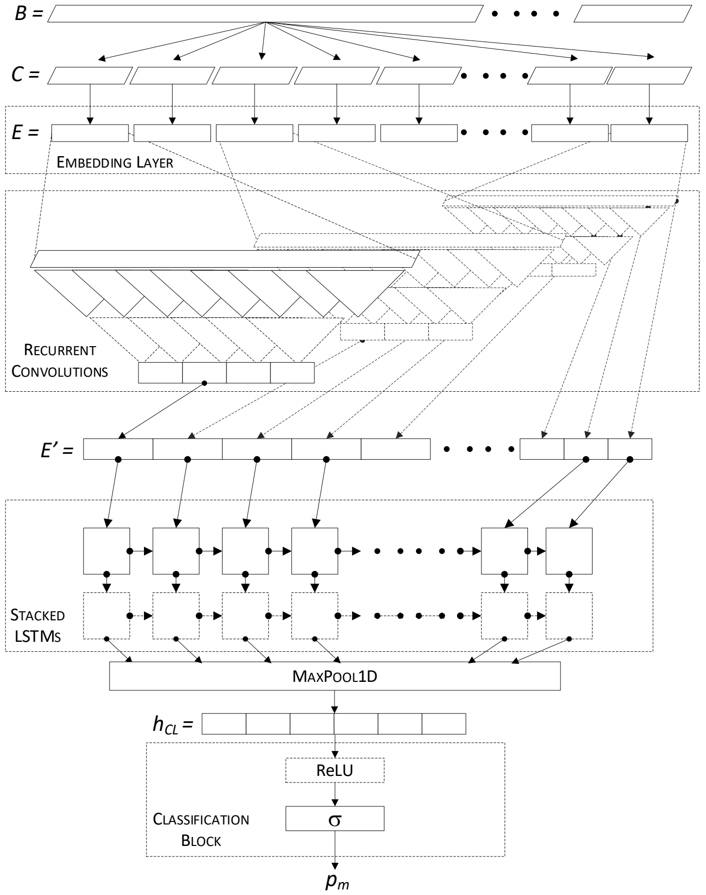
CPoLS - Convoluted Partitioning of Long Sequences

Process the input sequence in parts

- Split it first into smaller pieces of fixed length

Input each of the chunks using a Conv1D layer

Remaining part of the model is similar to LaMP



Agrawal 2018

Outline

Motivation

System Design and Models

Performance Evaluation

Conclusions

Datasets

Provided the first 1000 bytes of each VBScript file

240,504 VBScript files

- 66,028 malicious and 174,476 benign scripts

Randomly split

- Training set: 168,353
- Validation set: 24,050
- Test set: 48,101

Labels are obtained from the production antimalware detection system

Experimental Setup

Keras with the TensorFlow backend

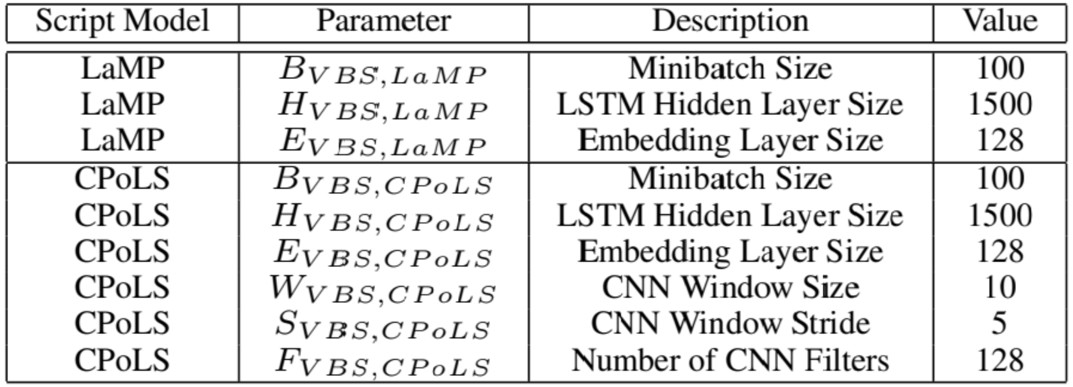
NVIDIA K40 GPU

Training and testing for all models:

- Maximum of 15 epochs with early stopping
- Adam optimizer, Cross entropy loss

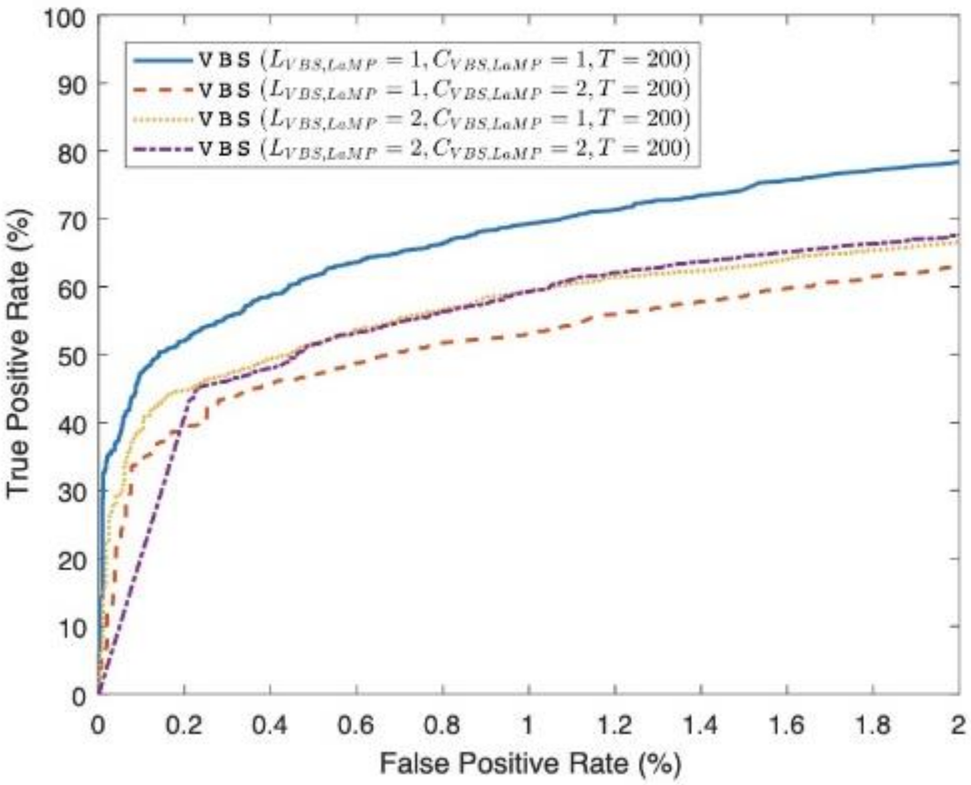
Process first 200 bytes for the LaMP model

Process all 1000 bytes for CPoLS model



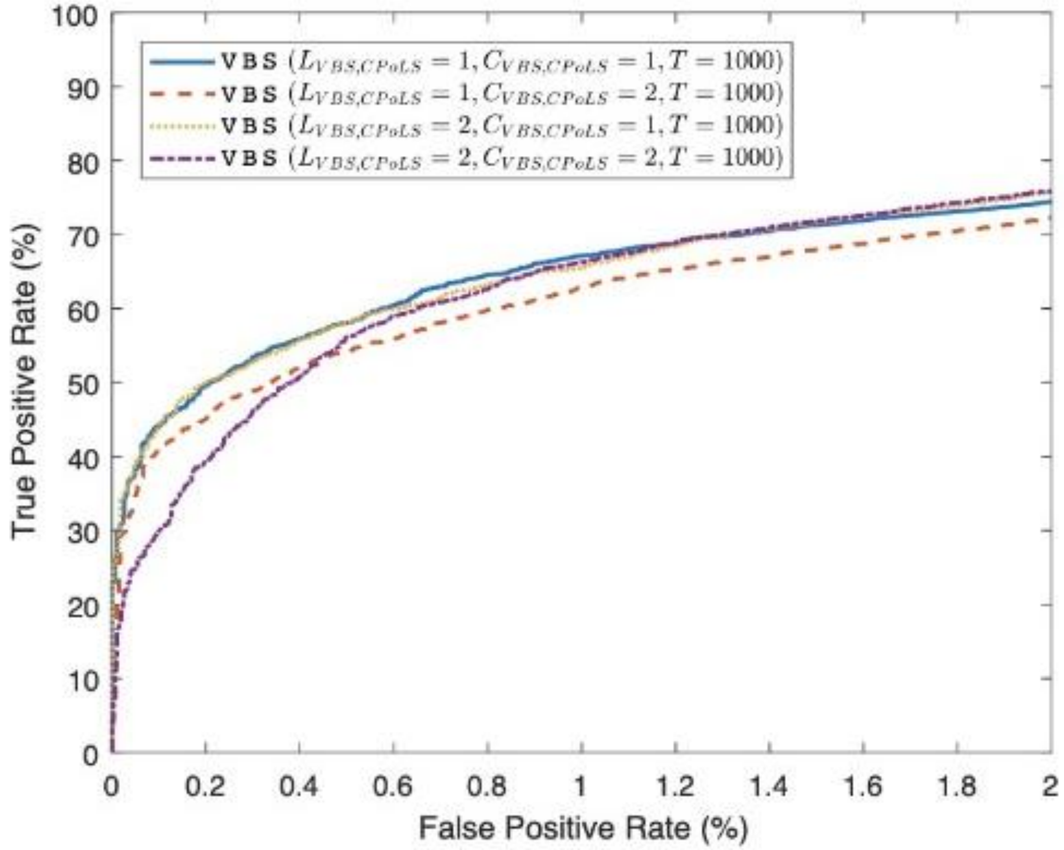
Script Model	Parameter	Description	Value
LaMP	$B_{VBS,LaMP}$	Minibatch Size	100
LaMP	$H_{VBS,LaMP}$	LSTM Hidden Layer Size	1500
LaMP	$E_{VBS,LaMP}$	Embedding Layer Size	128
CPoLS	$B_{VBS,CPoLS}$	Minibatch Size	100
CPoLS	$H_{VBS,CPoLS}$	LSTM Hidden Layer Size	1500
CPoLS	$E_{VBS,CPoLS}$	Embedding Layer Size	128
CPoLS	$W_{VBS,CPoLS}$	CNN Window Size	10
CPoLS	$S_{VBS,CPoLS}$	CNN Window Stride	5
CPoLS	$F_{VBS,CPoLS}$	Number of CNN Filters	128

LaMP Learning Models for VBScripts Zoomed into FPR = 2%



At an FPR of 1.0%, the TPR for the LaMP model is 69.3% with $L_{VBS,CPoLS}$ = 1, $C_{VBS,CPoLS}$ = 1

CPoLS Learning Models for VBScripts



CPoLS yields a TPR of 67.1% with $L_{VBS,CPoLS} = 1$, $C_{VBS,CPoLS} = 1$ at this FPR = 1.0%

Conclusions

Investigate combining static analysis and dynamic analysis

- Dynamic analysis - detect additional files which are dropped during execution of obfuscated commands

VbsNet

- Neural language models can detect malicious VBScript files

Simplest LaMP and CPoLS VBScript models with a single LSTM layer and classifier hidden layer offer the best, or nearly the best, performance

LaMP models trained with only the first 200 bytes outperform the CPoLS models which are trained with the first 1000 bytes

Thank you for viewing our presentation!