Opcode statistics for detecting compiler settings

Kenneth van Rijsbergen[1]

[1] MSc student System and Network Engineering
Faculty of Science
University of Amsterdam

5 February 2018

Introduction

- Reproducible builds
  - How to match the binary with the source code ?
    - Reproducible builds : binaries that can be reproduced from source code byte-for-byte

Introduction

- Reproducible builds
  - How to match the binary with the source code ?
    - Reproducible builds : binaries that can be reproduced from source code byte-for-byte
- Build-environment
  - Used tool-chains, version of the compiler, compiler flags
  - Lost after compilation and stripping

Introduction

- Reproducible builds
  - How to match the binary with the source code ?
    - Reproducible builds : binaries that can be reproduced from source code byte-for-byte
- Build-environment
  - Used tool-chains, version of the compiler, compiler flags
  - Lost after compilation and stripping
- Opcode statistics
  - Main approach
  - Related work in metamorphic malware detection

Related work / Background

- *Bilar [2007] :* Distribution of opcodes and statistical differences between goodware and malware
- *Austin et al [2013] :* 90% accuracy in distinguishing different compilers, using Hidden Markov models (HMM).

Related work / Background

- *Bilar [2007] :* Distribution of opcodes and statistical differences between goodware and malware
- *Austin et al [2013] :* 90% accuracy in distinguishing different compilers, using Hidden Markov models (HMM).

Hidden Markov Model, Graph embedding, ML classifiers

- *Wong & Stamp [2006], Santos et al., and many others.*
- *Mohammad et al [2016] :* Using Feature extraction and DT (Random Forest) scored 100% accuracy.

Related work / Background

- *Bilar [2007] :* Distribution of opcodes and statistical differences between goodware and malware
- *Austin et al [2013] :* 90% accuracy in distinguishing different compilers, using Hidden Markov models (HMM).

Hidden Markov Model, Graph embedding, ML classifiers

- *Wong & Stamp [2006], Santos et al., and many others.*
- *Mohammad et al [2016] :* Using Feature extraction and DT (Random Forest) scored 100% accuracy.

N-gram analysis

- N-gram is a sequence of n-items or larger
- *Santos et al [2010]. Santos et al [2013]. Kang et al [2016].*
- *Kang et al [2016] :* Showed using a 4-gram was best, detecting Android Malware, using SVM (Support vector machine).

Research questions

Research questions :

1. *How significant are the differences in the opcode frequencies when using different compiler versions ?*
2. *How significant are the differences in the opcode frequencies when using different compiler flags ?*
3. *What opcodes are responsible for the differences in the opcode frequencies ?*
4. *Are differences significant enough to detect what compiler flag or version is used for a binary ?*

Methodology

Approach :

- Compiled a **collection** of applications
  - 6 different optimisation flags
  - 8 different GCC versions
- Count the opcodes of the collections
  - Single opcodes (1-gram)
  - Opcode pairs (2-gram)
- Statistical analysis

Compiled programs

Compiled programs :

- barcode - *part of barcode-0.99*
- bash - *part of bash-4.4*
- cp - *part of coreutils-8.28*
- enscript - *part of enscript-1.6.6*
- find - *part of findutils-4.6.0*
- gap***** - *part of gap-4.8.9*
- gcal2txt - *part of gcal-4*
- gcal - *part of gcal-4*
- git-shell - *part of git 2.7.4*
- git - *part of git 2.7.4*
- lighttpd - *part of lighttpd-1.4.48*
- locate - *part of findutils-4.6.0*
- ls - *part of coreutils-8.28*
- mv - *part of coreutils-8.28*
- openssl***** - *part of openssl-1.0.2n*
- postgresql***** - *part of postgresql-10.1*
- sha256sum - *part of coreutils-8.28*
- sha384sum - *part of coreutils-8.28*
- units - *part of units-2.16*
- vim - *part of vim version 8.0.1391*

(Not included in the flag dataset **(*)**)

Sizes of programs

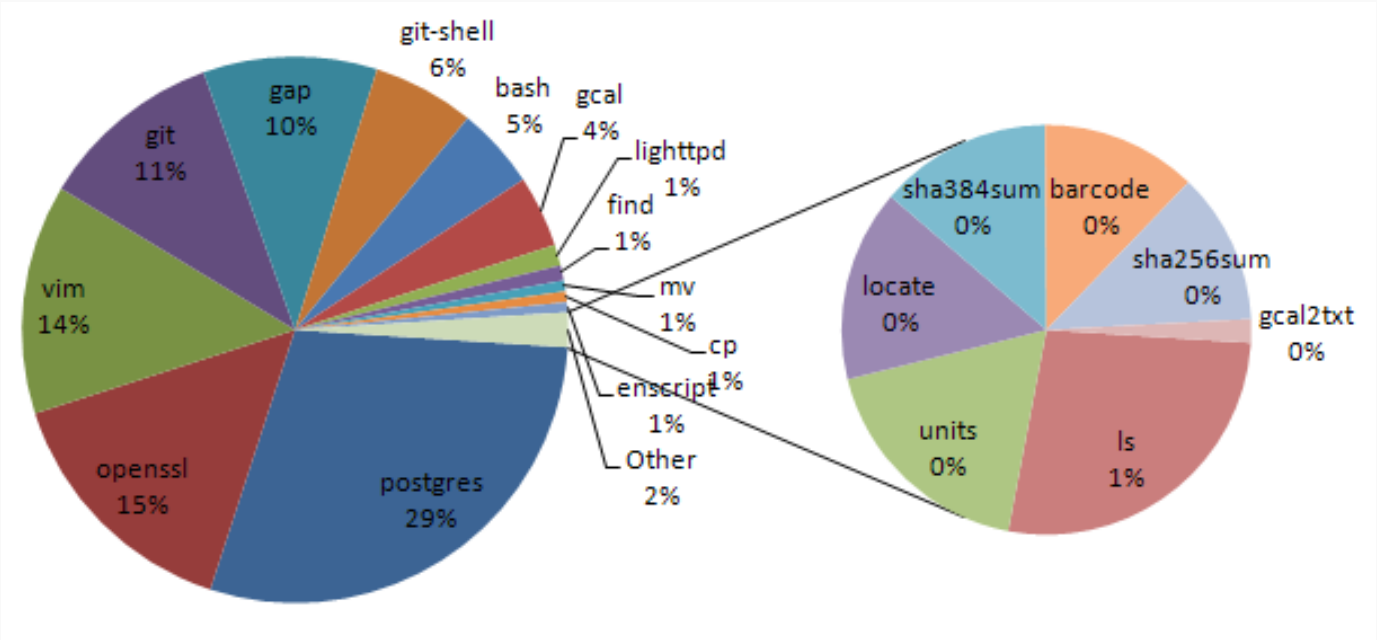


Figure – Sizes of programs

Compiler versions

Compiler versions :

- GCC : (Ubuntu/Linaro 4.4.7-8ubuntu7) 4.4.7
- GCC : (Ubuntu/Linaro 4.6.4-6ubuntu6) 4.6.4
- GCC : (Ubuntu/Linaro 4.7.4-3ubuntu12) 4.7.4
- GCC : (Ubuntu 4.8.5-4ubuntu2) 4.8.5
- GCC : (Ubuntu 4.9.4-2ubuntu1 16.04) 4.9.4
- GCC : (Ubuntu 5.4.1-2ubuntu1 16.04) 5.4.1 20160904
- GCC : (Ubuntu/Linaro 6.3.0-18ubuntu2 16.04) 6.3.0 20170519
- GCC : (Ubuntu 7.2.0-1ubuntu1 16.04) 7.2.0

Optimization flags

Table – Optimization flags

Flag		Description
-O0	Default	
-O1	Light optimization	Acts as a macro.
-O2	Increased optimization	All optimization of -O1 Plus additional flags without space trade-off.
-O3	Additional optimization	All optimizations of -O2 Plus additional flags.
-Os	Optimize for size	All the -O2 optimizations Plus other flags that reduce the size.
-Ofast	Optimize for speed	All the -O3 optimizations Plus other flags such as -fast-math. Some program refuse to compile.

Statistical analysis

Chi-squared test :

- Measures the difference or fit of data
- Difference between the actual data and the expected data
- Need Cramer's V due to large dataset

Statistical analysis

Chi-squared test :

- Measures the difference or fit of data
- Difference between the actual data and the expected data
- Need Cramer's V due to large dataset

Cramer's V :

- Indicates strength of relationship between 0 and 1
  - <0.10 indicates a weak relationship between the variables
  - 0.10 - 0.30 indicates a moderate relationship
  - >0.30 indicates a strong relationship

Statistical analysis

Chi-squared test :

- Measures the difference or fit of data
- Difference between the actual data and the expected data
- Need Cramer's V due to large dataset

Cramer's V :

- Indicates strength of relationship between 0 and 1
  - <0.10 indicates a weak relationship between the variables
  - 0.10 - 0.30 indicates a moderate relationship
  - >0.30 indicates a strong relationship

Z-scores :

- Number of std.dev an observation deviates from the mean
  - 0 = no deviation.
  - -2 or 2 = deviates 2 std.dev. from the mean
- The greater the Z-score, the more a value deviates from the mean

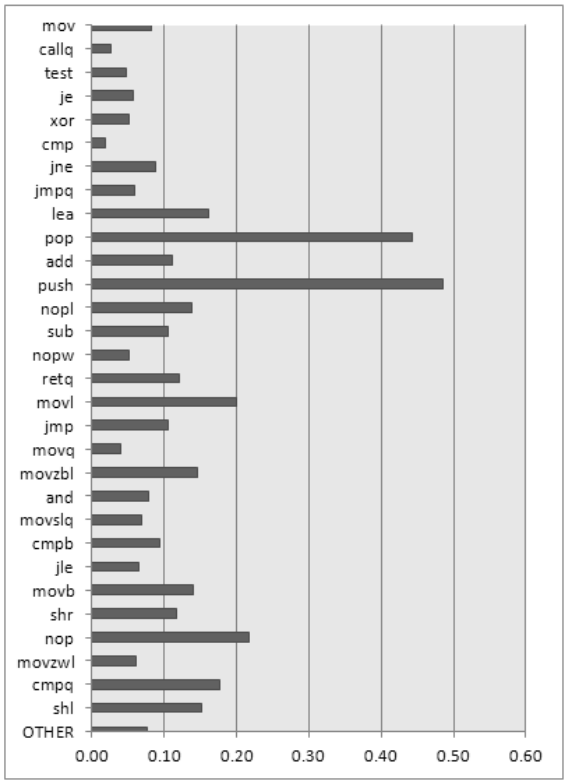
Results

GCC versions 1-gram

GCC versions 1-gram

Pearson's chi-squared test (χ2)		116455.3							
Cramér's V		0.025513							
p		0							
Opcode	**Average**	**GCC 4.4**	**GCC 4.6**	**GCC 4.7**	**GCC 4.8**	**GCC 4.9**	**GCC 5**	**GCC 6**	**GCC 7**
mov	34.90%	36.78%	36.56%	36.22%	34.27%	33.94%	33.98%	34.07%	33.70%
callq	8.30%	8.23%	8.43%	8.30%	8.21%	8.31%	8.28%	8.29%	8.30%
test	5.00%	4.94%	5.11%	5.01%	4.95%	4.86%	4.92%	4.95%	4.98%
je	4.70%	4.74%	4.81%	4.85%	4.68%	4.58%	4.66%	4.67%	4.70%
xor	4.60%	4.42%	4.66%	4.43%	4.45%	4.64%	4.63%	4.59%	4.59%
cmp	3.30%	3.33%	3.31%	3.36%	3.31%	3.29%	3.32%	3.30%	3.33%
jne	3.10%	2.93%	3.00%	3.00%	3.00%	3.15%	3.15%	3.15%	3.21%
jmpq	3.00%	3.02%	2.91%	3.10%	3.04%	3.02%	3.03%	3.03%	3.01%
lea	3.00%	2.62%	2.71%	2.87%	3.11%	3.12%	3.06%	3.06%	3.13%
pop	2.90%	2.05%	2.07%	2.10%	3.67%	3.46%	3.43%	3.34%	3.33%
add	2.90%	2.98%	3.14%	3.09%	2.87%	2.88%	2.85%	2.84%	2.79%
push	2.40%	1.55%	1.57%	1.59%	2.53%	3.01%	2.98%	2.97%	2.95%
nopl	2.30%	2.55%	2.25%	2.31%	2.33%	2.23%	2.20%	2.20%	2.31%
sub	1.50%	1.51%	1.51%	1.52%	1.41%	1.55%	1.55%	1.58%	1.58%
nopw	1.40%	1.41%	1.38%	1.44%	1.41%	1.39%	1.37%	1.36%	1.43%
retq	1.10%	1.07%	1.08%	1.12%	1.22%	1.12%	1.12%	1.10%	1.11%
movl	1.10%	1.17%	1.18%	1.18%	1.18%	1.02%	1.03%	1.03%	0.94%
jmp	1.00%	1.03%	0.92%	0.99%	0.99%	0.97%	0.98%	0.97%	0.94%
movq	1.00%	0.95%	0.97%	0.97%	0.96%	0.93%	0.95%	0.95%	0.96%
movzbl	0.90%	1.04%	0.95%	0.93%	0.93%	0.91%	0.89%	0.98%	0.91%
and	0.70%	0.65%	0.63%	0.66%	0.66%	0.67%	0.68%	0.69%	0.68%
movslq	0.60%	0.67%	0.64%	0.66%	0.67%	0.63%	0.64%	0.64%	0.63%
cmpb	0.60%	0.54%	0.60%	0.60%	0.59%	0.59%	0.58%	0.57%	0.56%
jle	0.50%	0.48%	0.48%	0.47%	0.48%	0.47%	0.47%	0.47%	0.45%
movb	0.50%	0.46%	0.48%	0.48%	0.48%	0.46%	0.46%	0.46%	0.41%
shr	0.40%	0.37%	0.39%	0.41%	0.42%	0.41%	0.42%	0.42%	0.40%
nop	0.40%	0.47%	0.41%	0.41%	0.41%	0.37%	0.37%	0.37%	0.39%
movzwl	0.40%	0.37%	0.39%	0.38%	0.38%	0.37%	0.37%	0.37%	0.37%
cmpq	0.40%	0.38%	0.38%	0.41%	0.36%	0.36%	0.36%	0.36%	0.33%
shl	0.40%	0.40%	0.38%	0.38%	0.37%	0.37%	0.35%	0.34%	0.34%
OTHER	6.90%	6.88%	6.70%	6.76%	6.67%	6.91%	6.91%	6.90%	7.23%

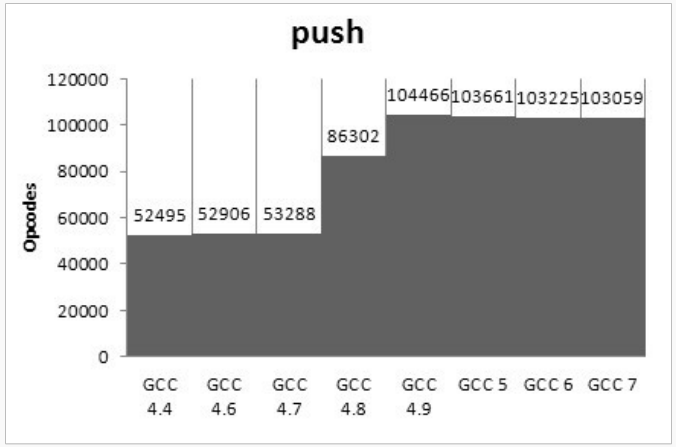
Differences in relative frequencies
0.08
0.03
0.05
0.06
0.05
0.02
0.09
0.06
0.16
0.44
0.11
0.49
0.14
0.11
0.05
0.12
0.20
0.11
0.04
0.15
0.08
0.07
0.09
0.06
0.14
0.12
0.22
0.06
0.18
0.15
0.08

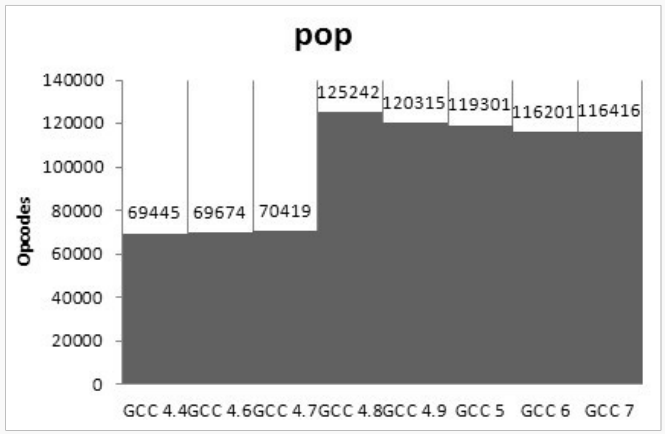


Relative frequencies of opcodes for different GCC versions (1-gram).

GCC versions 1-gram

Opcode	GCC 4.4	GCC 4.6	GCC 4.7	GCC 4.8	GCC 4.9	GCC 5	GCC 6	GCC 7
mov	1.74	1.08	0.58	-1	-0.65	-0.54	-0.46	-0.75
callq	-1.13	-0.25	-1.28	-0.89	0.89	0.76	0.82	1.09
test	-1.21	0.6	-0.99	-0.65	-0.55	0.34	0.79	1.65
je	-0.38	0.07	0.67	-1.12	-1.55	0.27	0.48	1.56
xor	-1.2	0.07	-1.41	-0.82	0.95	0.96	0.65	0.8
cmp	-0.49	-1.51	-0.76	-0.61	0.34	1.02	0.59	1.42
jne	-1.2	-0.91	-0.95	-0.58	0.76	0.81	0.8	1.28
jmpq	-0.44	-2.26	0.09	0.04	0.55	0.77	0.65	0.6
lea	-1.55	-1.28	-0.65	0.55	0.81	0.61	0.58	0.93
pop	-1.22	-1.21	-1.18	0.94	0.75	0.71	0.59	0.6
add	0.29	1.8	1.08	-0.88	-0.15	-0.5	-0.57	-1.07
push	-1.19	-1.17	-1.16	0.15	0.88	0.84	0.83	0.82
nopl	2.18	-0.9	-0.34	0.2	-0.4	-0.6	-0.69	0.54
sub	-0.47	-0.61	-0.61	-1.71	0.61	0.68	1	1.12
nopw	-0.14	-1.63	0.31	0.15	0.31	-0.41	-0.51	1.92
retq	-1.19	-1.18	-0.51	1.9	0.4	0.34	0.05	0.19
movl	0.83	0.85	0.74	1.08	-0.73	-0.55	-0.55	-1.66
jmp	1.17	-2.18	-0.1	0.37	0.36	0.61	0.2	-0.44
movq	-1.03	-0.68	-0.41	-0.12	-0.89	0.59	0.59	1.95
movzbl	1.97	-0.23	-0.74	-0.38	-0.37	-0.89	1.11	-0.47
and	-0.88	-1.63	-0.65	-0.28	0.62	0.93	1.11	0.77
movslq	1.49	-1.57	0.1	1.17	-0.54	0.1	0.05	-0.8
cmpb	-2.22	0.46	0.36	0.52	0.9	0.55	-0.02	-0.56
jle	1.22	-0.64	-1.28	0.49	0.21	0.91	0.54	-1.44
movb	-0.23	0.36	0.45	0.84	0.19	0.45	0.31	-2.36
shr	-1.62	-1.3	-0.31	0.76	0.63	0.92	0.87	0.06
nop	2.14	-0.04	0.16	0.45	-0.77	-0.82	-0.95	-0.18
movzwl	-1.27	1.83	-1.15	0.59	-0.34	0.47	-0.18	0.04
cmpq	0.51	0.02	1.82	-0.29	-0.07	-0.12	-0.04	-1.82
shl	1.82	0.42	0.1	-0.16	0.72	-0.66	-1.23	-1
OTHER	-0.24	-1.14	-0.98	-0.88	0.49	0.51	0.47	1.78





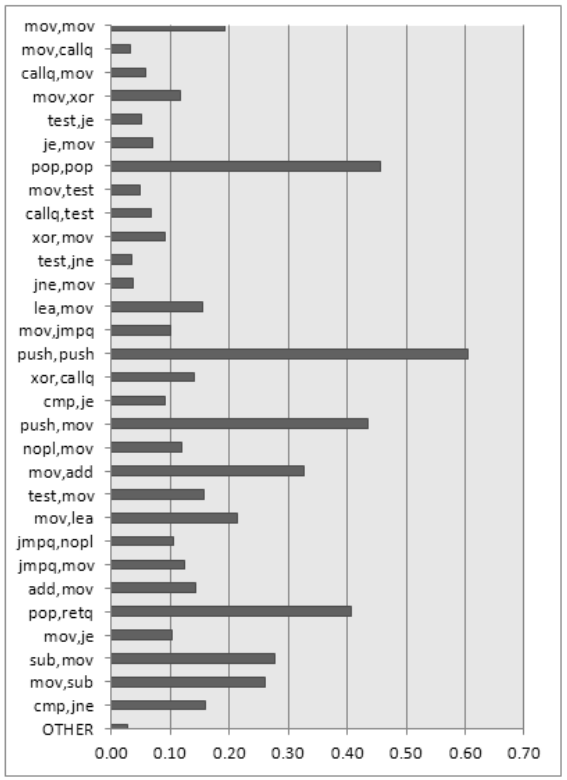
Z-scores and the 2 greatest deviators for different GCC versions (1-gram).

Results

GCC versions 2-gram

GCC versions 2-gram

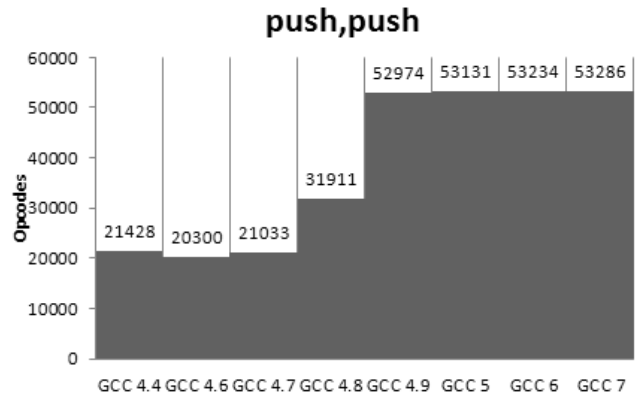
Pearson's chi-squared test (χ2)		146756.3								
Cramér's V		0.036947								
p		0								Differences in
Opcode	Average	GCC 4.4	GCC 4.6	GCC 4.7	GCC 4.8	GCC 4.9	GCC 5	GCC 6	GCC 7	relative frequencies
mov,mov	15.50%	17.54%	17.23%	16.96%	14.84%	14.43%	14.49%	14.51%	14.15%	0.19
mov,callq	5.49%	5.56%	5.57%	5.60%	5.51%	5.45%	5.42%	5.42%	5.43%	0.03
callq,mov	3.97%	4.06%	4.12%	4.02%	4.01%	3.90%	3.89%	3.88%	3.90%	0.06
mov,xor	2.18%	2.05%	2.23%	2.03%	2.02%	2.28%	2.28%	2.28%	2.28%	0.12
test,je	2.11%	2.08%	2.13%	2.15%	2.08%	2.04%	2.12%	2.12%	2.14%	0.05
je,mov	2.09%	2.16%	2.15%	2.18%	2.07%	2.02%	2.05%	2.05%	2.04%	0.07
pop,pop	1.97%	1.35%	1.36%	1.37%	2.48%	2.34%	2.32%	2.25%	2.25%	0.46
mov,test	1.61%	1.58%	1.57%	1.60%	1.64%	1.61%	1.64%	1.65%	1.61%	0.05
callq,test	1.57%	1.59%	1.62%	1.64%	1.55%	1.54%	1.53%	1.54%	1.54%	0.07
xor,mov	1.45%	1.47%	1.42%	1.42%	1.36%	1.46%	1.46%	1.50%	1.50%	0.09
test,jne	1.28%	1.27%	1.30%	1.29%	1.28%	1.27%	1.29%	1.28%	1.25%	0.04
jne,mov	1.22%	1.20%	1.25%	1.21%	1.21%	1.21%	1.23%	1.24%	1.25%	0.04
lea,mov	1.16%	1.03%	1.09%	1.17%	1.22%	1.19%	1.19%	1.18%	1.18%	0.15
mov,jmpq	1.13%	1.09%	1.04%	1.12%	1.15%	1.15%	1.16%	1.16%	1.16%	0.10
push,push	1.12%	0.63%	0.60%	0.63%	0.93%	1.52%	1.53%	1.53%	1.53%	0.61
xor,callq	1.06%	1.02%	1.18%	1.05%	1.02%	1.06%	1.05%	1.06%	1.06%	0.14
cmp,je	1.06%	1.01%	1.08%	1.12%	1.07%	1.04%	1.06%	1.06%	1.05%	0.09
push,mov	0.95%	0.66%	0.71%	0.67%	1.18%	1.13%	1.08%	1.06%	1.06%	0.44
nopl,mov	0.92%	1.00%	0.93%	0.95%	0.88%	0.89%	0.89%	0.88%	0.95%	0.12
mov,add	0.89%	1.11%	1.10%	1.10%	0.77%	0.77%	0.78%	0.77%	0.75%	0.33
test,mov	0.87%	0.86%	0.96%	0.86%	0.86%	0.86%	0.81%	0.84%	0.88%	0.16
mov,lea	0.83%	0.72%	0.71%	0.76%	0.90%	0.89%	0.88%	0.89%	0.89%	0.21
jmpq,nopl	0.82%	0.87%	0.80%	0.82%	0.82%	0.78%	0.78%	0.78%	0.87%	0.10
jmpq,mov	0.75%	0.77%	0.71%	0.78%	0.76%	0.75%	0.77%	0.77%	0.68%	0.12
add,mov	0.69%	0.63%	0.71%	0.68%	0.74%	0.70%	0.69%	0.70%	0.68%	0.14
pop,retq	0.69%	0.52%	0.54%	0.55%	0.87%	0.77%	0.76%	0.74%	0.76%	0.41
mov,je	0.68%	0.64%	0.72%	0.68%	0.68%	0.68%	0.65%	0.67%	0.71%	0.10
sub,mov	0.65%	0.62%	0.61%	0.61%	0.52%	0.72%	0.71%	0.72%	0.72%	0.28
mov,sub	0.65%	0.72%	0.72%	0.69%	0.53%	0.65%	0.62%	0.63%	0.63%	0.26
cmp,jne	0.59%	0.54%	0.55%	0.59%	0.60%	0.59%	0.60%	0.60%	0.65%	0.16
OTHER	44.04%	43.63%	43.29%	43.70%	44.42%	44.30%	44.23%	44.24%	44.47%	0.03

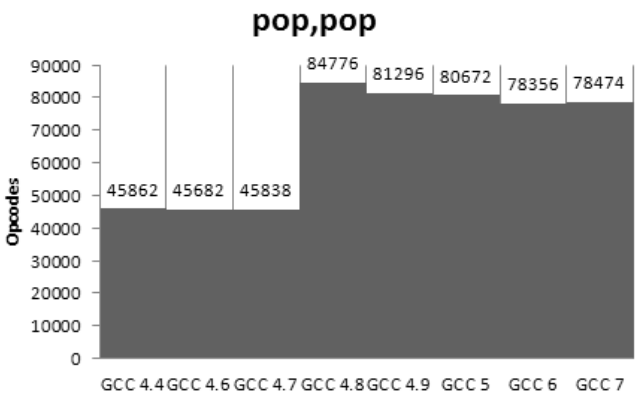


Relative frequencies of opcodes for different GCC versions (2-gram).

GCC versions 2-gram

Opcode	GCC 4.4	GCC 4.6	GCC 4.7	GCC 4.8	GCC 4.9	GCC 5	GCC 6	GCC 7
mov,mov	1.52	1.14	0.89	-0.61	-0.73	-0.66	-0.65	-0.91
mov,callq	0.21	-1.73	-0.87	-0.35	1.11	0.21	0.07	1.35
callq,mov	1.21	1.52	-1.14	0.6	-0.53	-0.53	-0.98	-0.15
mov,xor	-1.02	0	-1.27	-1.16	0.85	0.84	0.85	0.91
test,je	-1.13	-0.38	-0.17	-0.82	-0.92	0.96	0.97	1.48
je,mov	1.54	0.51	1.2	-1.01	-1.25	-0.3	-0.25	-0.44
pop,pop	-1.2	-1.21	-1.2	0.94	0.75	0.72	0.59	0.6
mov,test	-0.93	-1.54	-0.94	0.4	0.4	1.04	1.14	0.43
callq,test	0.22	0.85	1.89	-1.33	-0.38	-0.71	-0.48	-0.05
xor,mov	0.01	-0.9	-0.95	-1.48	0.51	0.58	1.06	1.18
test,jne	-1.4	-0.36	-0.86	-0.33	0.48	1.64	1.06	-0.22
jne,mov	-1.2	-0.11	-1.23	-0.71	0.05	0.87	0.94	1.38
lea,mov	-1.85	-1.18	-0.2	0.74	0.7	0.64	0.52	0.64
mov,jmpq	-0.83	-1.85	-0.62	0.2	0.57	0.82	0.79	0.92
push,push	-1.05	-1.12	-1.07	-0.4	0.9	0.91	0.92	0.92
xor,callq	-1.15	1.96	-0.79	-0.98	0.19	0.17	0.23	0.37
cmp,je	-2.16	-0.03	1.24	0.04	-0.4	0.66	0.43	0.21
push,mov	-1.25	-1.08	-1.26	0.97	0.84	0.64	0.57	0.57
nopl,mov	1.8	-0.22	0.24	-1.18	-0.47	-0.52	-0.76	1.09
mov,add	1.28	1.19	1.14	-0.79	-0.71	-0.61	-0.68	-0.82
test,mov	-0.51	1.99	-0.61	-0.2	0.2	-1.27	-0.37	0.76
mov,lea	-1.23	-1.44	-0.92	0.68	0.76	0.64	0.71	0.79
jmpq,nopl	1.25	-0.78	-0.39	-0.03	-0.63	-0.54	-0.74	1.86
jmpq,mov	0.38	-1.53	0.32	0.37	0.33	0.86	0.89	-1.62
add,mov	-2.04	0.06	-0.71	1.31	0.5	0.29	0.55	0.03
pop,retq	-1.26	-1.16	-1.09	1.26	0.63	0.57	0.44	0.61
mov,je	-1.72	0.89	-0.44	-0.17	0.46	-0.6	0.02	1.56
sub,mov	-0.53	-0.67	-0.72	-1.61	0.85	0.83	0.92	0.94
mov,sub	1.2	0.91	0.44	-2.1	0.11	-0.28	-0.2	-0.07
cmp,jne	-1.39	-1.25	-0.44	0.23	0.28	0.48	0.42	1.66
OTHER	-0.8	-1.46	-1.18	0.14	0.75	0.73	0.73	1.09





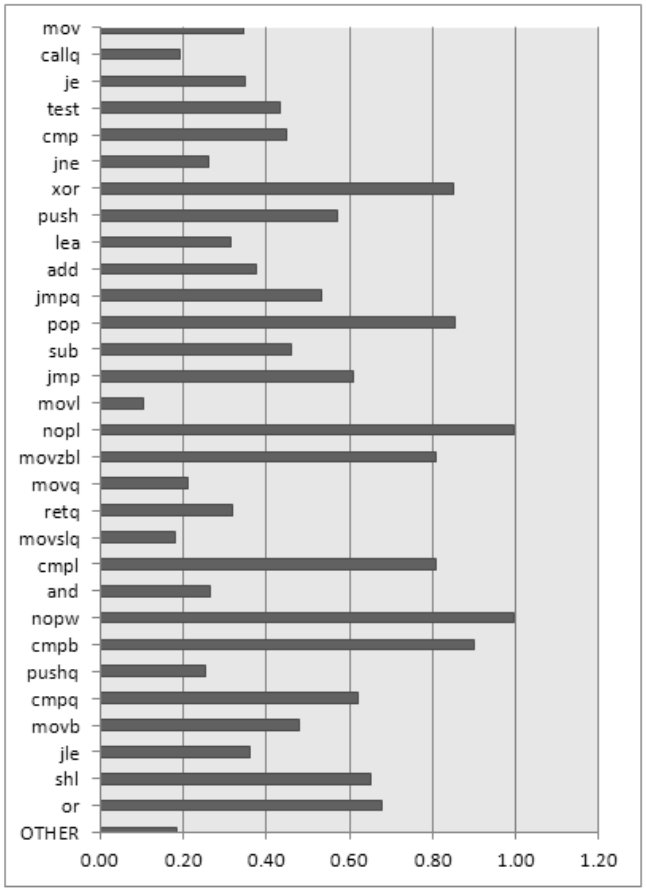
Z-scores and the 2 greatest deviators for different GCC versions (2-gram).

Results

Flags 1-gram

Flags 1-gram

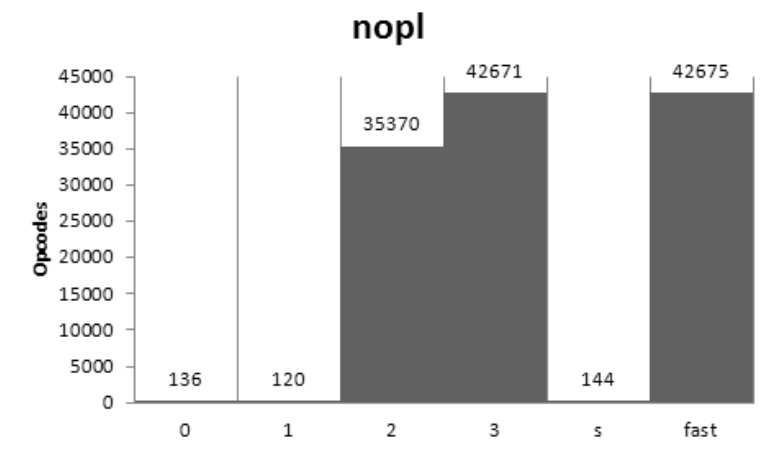
Pearson's chi-squared test (χ2)		668066.8						
Cramér's V		0.115758						
p		0						Differences in
Opcode	Average	0	1	2	3	s	fast	relative frequencies
mov	36.90%	49.92%	37.25%	32.75%	33.06%	32.72%	33.07%	0.34
callq	8.30%	7.56%	9.23%	8.25%	7.97%	9.38%	7.97%	0.19
je	4.90%	3.55%	4.93%	5.00%	5.43%	5.16%	5.44%	0.35
test	4.80%	3.22%	4.44%	5.37%	5.70%	4.56%	5.70%	0.44
cmp	3.40%	2.19%	3.69%	3.53%	3.96%	3.46%	3.96%	0.45
jne	3.30%	2.65%	3.58%	3.44%	3.52%	3.53%	3.52%	0.26
xor	3.10%	0.73%	0.82%	4.36%	4.06%	4.90%	4.06%	0.85
push	3.10%	1.97%	3.61%	3.42%	2.85%	4.60%	2.85%	0.57
lea	3.10%	2.32%	3.40%	3.11%	3.23%	3.37%	3.24%	0.32
add	2.80%	3.13%	2.86%	2.93%	2.80%	1.96%	2.80%	0.37
jmpq	2.70%	1.58%	2.26%	3.31%	3.39%	2.17%	3.38%	0.53
pop	2.50%	0.54%	2.59%	3.40%	2.84%	3.66%	2.84%	0.85
sub	1.70%	1.89%	1.93%	1.75%	1.65%	1.04%	1.65%	0.46
jmp	1.60%	2.31%	2.46%	1.06%	0.96%	2.39%	0.96%	0.61
movl	1.40%	1.47%	1.46%	1.33%	1.34%	1.32%	1.34%	0.10
nopl	1.10%	0.01%	0.01%	2.22%	2.09%	0.01%	2.09%	1.00
movzbl	1.10%	1.65%	1.06%	0.87%	1.07%	0.32%	1.07%	0.81
movq	1.00%	0.89%	1.13%	1.06%	1.06%	1.04%	1.06%	0.21
retq	1.00%	1.17%	1.12%	1.09%	0.82%	1.20%	0.82%	0.32
movslq	0.80%	0.71%	0.87%	0.72%	0.80%	0.79%	0.80%	0.18
cmpl	0.70%	1.13%	1.35%	0.27%	0.26%	1.37%	0.26%	0.81
and	0.70%	0.76%	0.56%	0.62%	0.74%	0.61%	0.74%	0.27
nopw	0.70%	0.01%	0.01%	1.37%	1.18%	0.01%	1.18%	1.00
cmpb	0.60%	0.07%	0.74%	0.67%	0.68%	0.65%	0.68%	0.90
pushq	0.50%	0.50%	0.50%	0.47%	0.41%	0.55%	0.41%	0.26
cmpq	0.50%	0.87%	0.38%	0.33%	0.35%	0.42%	0.35%	0.62
movb	0.50%	0.28%	0.52%	0.48%	0.53%	0.40%	0.53%	0.48
jle	0.40%	0.33%	0.46%	0.47%	0.51%	0.41%	0.51%	0.36
shl	0.40%	0.63%	0.31%	0.27%	0.33%	0.22%	0.33%	0.65
or	0.30%	0.24%	0.26%	0.28%	0.30%	0.76%	0.30%	0.68
OTHER	6.10%	5.72%	6.23%	5.83%	6.09%	7.03%	6.08%	0.19



Relative frequencies of opcodes for different Flags (1-gram).

Flags 1-gram

Opcode	0	1	2	3	s	fast
mov	1.84	-0.42	-0.62	0.11	-1.01	0.11
callq	0.69	-0.43	-0.91	0.97	-1.28	0.97
je	-0.64	-0.62	-0.36	1.27	-0.91	1.27
test	-0.72	-0.73	-0.01	1.22	-0.98	1.22
cmp	-0.94	-0.32	-0.3	1.24	-0.91	1.24
jne	-0.37	-0.5	-0.44	1.24	-1.17	1.24
xor	-1.21	-1.31	0.45	0.87	0.33	0.87
push	-1.89	-0.03	-0.04	0.51	0.95	0.51
lea	-0.63	-0.34	-0.54	1.25	-1.02	1.26
add	1.16	-0.44	-0.19	0.57	-1.66	0.57
jmpq	-0.8	-0.75	0.26	1.16	-1.03	1.15
pop	-1.87	-0.33	0.51	0.74	0.22	0.73
sub	1.13	-0.05	-0.22	0.47	-1.8	0.47
jmp	1.56	0.67	-0.97	-0.74	0.23	-0.75
movl	1.32	-0.48	-0.66	0.6	-1.38	0.6
nopl	-0.91	-0.91	0.69	1.02	-0.91	1.02
movzbl	1.56	-0.27	-0.49	0.31	-1.43	0.32
movq	0.09	-0.43	-0.47	1.12	-1.44	1.12
retq	2.02	-0.35	-0.2	-0.42	-0.63	-0.41
movslq	0.47	-0.26	-0.92	1.02	-1.34	1.02
cmpl	1.22	0.85	-0.98	-0.85	0.61	-0.85
and	1.05	-1.01	-0.59	0.82	-1.08	0.82
nopw	-0.91	-0.91	0.79	0.97	-0.91	0.97
cmpb	-1.83	0.26	0.15	0.86	-0.29	0.86
pushq	1.89	-0.6	-0.64	0.07	-0.8	0.07
cmpq	2.02	-0.48	-0.58	-0.22	-0.53	-0.22
movb	-0.94	-0.1	-0.19	1.17	-1.13	1.19
jle	-0.52	-0.48	-0.24	1.22	-1.2	1.22
shl	1.86	-0.47	-0.58	0.08	-0.98	0.09
or	-0.4	-0.93	-0.68	0.07	1.87	0.07
OTHER	0.72	-0.88	-0.95	1.01	-0.89	0.99



Z-scores and the 2 greatest deviators for different Flags (1-gram).

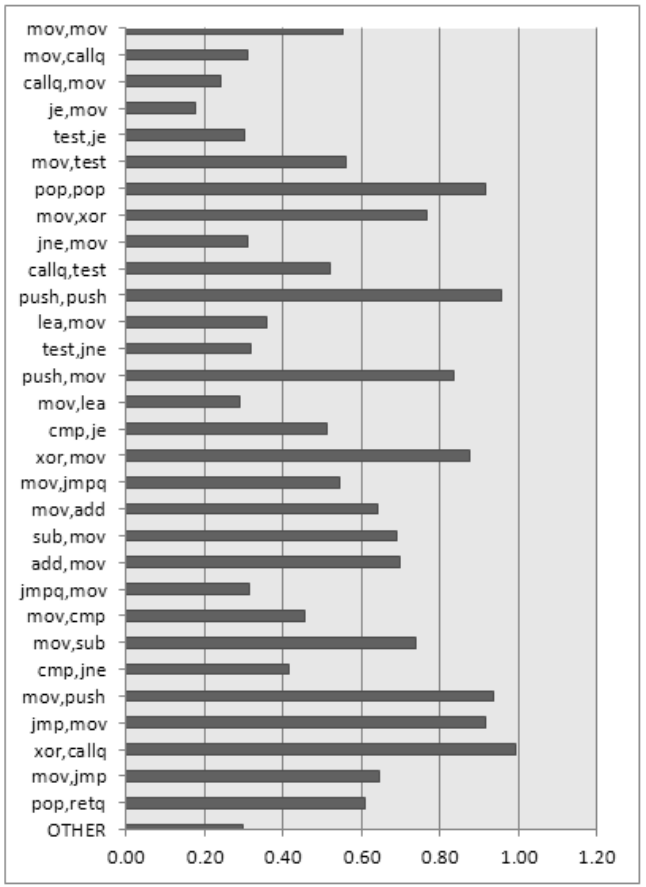
Results

Flags 2-gram

Flags 2-gram

Pearson's chi-squared test (χ2)		570972.1					
Cramér's V		0.13632					
p		0					
Opcode	**Average**	**0**	**1**	**2**	**3**	**s**	**fast**
mov,mov	36.90%	27.90%	18.26%	13.04%	13.22%	12.51%	13.23%
mov,callq	8.30%	6.72%	7.30%	5.17%	5.05%	5.79%	5.05%
callq,mov	4.90%	4.09%	4.71%	3.62%	3.58%	3.78%	3.58%
je,mov	4.80%	2.33%	2.13%	2.14%	2.38%	1.96%	2.37%
test,je	3.40%	1.74%	2.38%	2.29%	2.50%	2.02%	2.50%
mov,test	3.30%	1.39%	1.79%	1.96%	2.23%	0.98%	2.23%
pop,pop	3.10%	0.20%	1.84%	2.29%	2.00%	2.42%	2.00%
mov,xor	3.10%	0.60%	0.63%	2.33%	2.16%	2.58%	2.16%
jne,mov	3.10%	1.90%	1.55%	1.31%	1.40%	1.42%	1.40%
callq,test	2.80%	0.99%	1.30%	1.61%	1.56%	2.06%	1.56%
push,push	2.70%	0.11%	2.49%	1.72%	1.51%	1.98%	1.51%
lea,mov	2.50%	1.93%	1.52%	1.24%	1.26%	1.35%	1.26%
test,jne	1.70%	1.19%	1.32%	1.52%	1.60%	1.09%	1.59%
push,mov	1.60%	1.40%	0.32%	1.30%	1.00%	1.98%	1.00%
mov,lea	1.40%	1.22%	1.30%	0.92%	0.96%	0.99%	0.96%
cmp,je	1.10%	0.60%	0.93%	1.14%	1.23%	0.83%	1.23%
xor,mov	1.10%	0.20%	0.21%	1.43%	1.31%	1.58%	1.31%
mov,jmpq	1.00%	0.70%	0.72%	1.13%	1.24%	0.57%	1.24%
mov,add	1.00%	1.58%	0.72%	0.87%	0.82%	0.57%	0.82%
sub,mov	0.80%	1.57%	0.99%	0.85%	0.76%	0.49%	0.76%
add,mov	0.70%	1.66%	0.82%	0.71%	0.68%	0.51%	0.68%
jmpq,mov	0.70%	0.90%	0.96%	0.78%	0.87%	0.66%	0.87%
mov,cmp	0.70%	1.14%	0.94%	0.62%	0.71%	0.69%	0.71%
mov,sub	0.60%	1.52%	0.39%	0.71%	0.65%	0.57%	0.65%
cmp,jne	0.50%	0.69%	1.09%	0.64%	0.67%	0.81%	0.67%
mov,push	0.50%	0.62%	0.08%	0.88%	0.71%	1.31%	0.71%
jmp,mov	0.50%	1.52%	1.50%	0.13%	0.13%	0.97%	0.13%
xor,callq	0.40%	0.01%	0.02%	1.01%	0.99%	1.09%	0.99%
mov,jmp	0.40%	1.08%	1.04%	0.42%	0.38%	0.76%	0.38%
pop,retq	0.30%	0.33%	0.70%	0.76%	0.60%	0.84%	0.60%
OTHER	6.10%	32.18%	40.04%	45.44%	45.84%	44.86%	45.83%

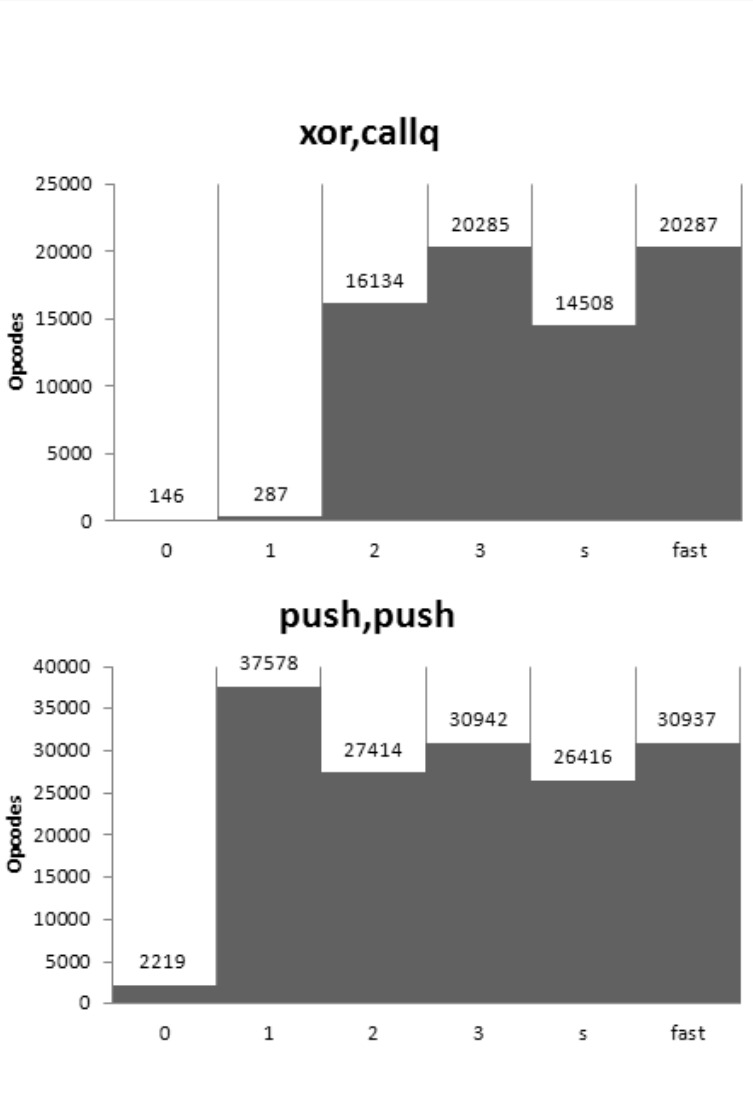
Differences in relative frequencies
0.55
0.31
0.24
0.18
0.30
0.56
0.92
0.77
0.31
0.52
0.96
0.36
0.32
0.84
0.29
0.51
0.88
0.54
0.64
0.69
0.70
0.31
0.46
0.74
0.42
0.94
0.91
0.99
0.64
0.61
0.30



Relative frequencies of opcodes for different Flags (2-gram).

Flags 2-gram

Opcode	0	1	2	3	s	fast
mov,mov	1.95	-0.14	-0.6	-0.17	-0.87	-0.17
mov,callq	1.67	0.33	-0.91	0.02	-1.13	0.02
callq,mov	1.37	0.21	-0.87	0.37	-1.44	0.37
je,mov	0.9	-0.75	-0.56	0.87	-1.34	0.87
test,je	-0.34	-0.39	-0.33	1.19	-1.33	1.19
mov,test	-0.23	-0.39	-0.06	1.11	-1.53	1.11
pop,pop	-1.9	-0.19	0.44	0.76	0.13	0.76
mov,xor	-1.15	-1.35	0.44	0.9	0.26	0.9
jne,mov	1.73	-0.44	-0.78	0.26	-1.03	0.26
callq,test	-1.04	-1.24	-0.11	1.07	0.26	1.07
push,push	-1.93	0.95	0.12	0.41	0.04	0.41
lea,mov	1.88	-0.31	-0.72	0.04	-0.93	0.04
test,jne	0.02	-0.69	-0.07	1.1	-1.45	1.09
push,mov	1.06	-1.83	0.04	0.01	0.72	0.01
mov,lea	1.55	0.21	-0.92	0.21	-1.27	0.22
cmp,je	-0.83	-0.58	0.07	1.19	-1.05	1.2
xor,mov	-1.22	-1.31	0.49	0.85	0.34	0.85
mov,jmpq	-0.3	-0.83	0.15	1.13	-1.28	1.13
mov,add	1.87	-0.64	-0.29	0.03	-1	0.03
sub,mov	1.88	-0.18	-0.34	-0.1	-1.14	-0.1
add,mov	1.96	-0.32	-0.43	-0.16	-0.9	-0.16
jmpq,mov	0.99	-0.12	-0.67	0.7	-1.6	0.71
mov,cmp	1.81	-0.03	-0.87	0.04	-0.99	0.04
mov,sub	1.93	-0.85	-0.28	-0.06	-0.68	-0.06
cmp,jne	0.51	1.39	-1.29	0.21	-1.02	0.21
mov,push	0.11	-1.97	0.28	0.35	0.87	0.35
jmp,mov	1.55	0.82	-0.83	-0.79	0.04	-0.79
xor,callq	-1.26	-1.24	0.45	0.89	0.27	0.89
mov,jmp	1.74	0.64	-0.83	-0.64	-0.27	-0.64
pop,retq	-1.93	-0.15	0.61	0.65	0.17	0.64
OTHER	-0.47	-0.91	-0.14	1.23	-0.95	1.23



Z-scores and the 2 greatest deviators for different Flags (2-gram).

Discussion

- Z-scores can act as weights for machine learning
- Flags will be easy-er to differentiate then GCC versions

	Chi-squared	p	Cramérs V
Dataset (GCC 5)	184522.4	0	**0.055**
Versions 1-gram	116455.3	0	**0.025**
Versions 2-gram	146756.3	0	**0.037**
Flags 1-gram	668066.8	0	**0.116**
Flags 2-gram	570972.1	0	**0.136**

Table – Analysis of matrixes

Cramer's V :

- Indicates strength of relationship between 0 and 1
  - <0.10 indicates a weak relationship between the variables
  - 0.10 - 0.30 indicates a moderate relationship
  - >0.30 indicates a strong relationship

Discussion

- Z-scores can act as weights for machine learning
- Flags will be easy-er to differentiate then GCC versions

	Chi-squared	p	Cramérs V
Dataset (GCC 5)	184522.4	0	**0.055**
Versions 1-gram	116455.3	0	**0.025**
Versions 2-gram	146756.3	0	**0.037**
Flags 1-gram	668066.8	0	**0.116**
Flags 2-gram	570972.1	0	**0.136**

Table – Analysis of matrixes

Cramer's V :

- Indicates strength of relationship between 0 and 1
  - <0.10 indicates a weak relationship between the variables
  - 0.10 - 0.30 indicates a moderate relationship
  - >0.30 indicates a strong relationship
- Enough to train a classifier ?
  - Successful in distinguishing malware
  - Unable to distinguish between hand-written assembly and compiled code
- 2-grams perform better than 1-grams. Confirms related work.
- Improvements to this research : Dataset

Conclusion and Future work

Conclusion

- Differences do occur
- However weak, patterns are visible
- Ground for future research (machine learning)

Future Work

- Create larger dataset
  - Using existing reproducible build or build automation tools
- Train and apply ML classifiers
- System call and library call statistics
- Measure changes on individual applications

FIN

Thank you. Questions?

[References]

Citations in this presentation :

- D. Bilar, “Opcodes as predictor for malware,” vol. 1, 01 2007.
- T. H. Austin, E. Filiol, S. Josse, and M. Stamp, “Exploring hidden markov models for virus analysis : a semantic approach,” in System Sciences (HICSS), 2013 46th Hawaii International Conference on IEEE, 2013, pp. 5039–5048.
- W. Wong and M. Stamp, “Hunting for metamorphic engines,” Journal in Computer Virology , vol. 2, no. 3, pp. 211–229, 2006
- M. Fazlali, P. Khodamoradi, F. Mardukhi, M. Nos- rati, and M. M. Dehshibi, “Metamorphic malware detection using opcode frequency rate and decision tree,” International Journal of Information Security and Privacy (IJISP) , vol. 10, no. 3, pp. 67–86, 2016
- I. Santos, F. Brezo, J. Nieves, Y. K. Penya, B. Sanz, C. Laorden, and P. G. Bringas, “Idea : Opcode- sequence-based malware detection,” in International Symposium on Engineering Secure Software and Sys- tems . Springer, 2010, pp. 35–43
- I. Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas, “Opcode sequences as representation of executables for data-mining-based unknown malware detection,” Information Sciences , vol. 231, pp. 64–82, 2013.
- B. Kang, S. Y. Yerima, S. Sezer, and K. McLaugh- lin, “N-gram opcode analysis for android malware detection,” arXiv preprint arXiv :1612.01445 , 2016.