John the Ripper and Hashcat - Markov mode comparison - "Take 2"

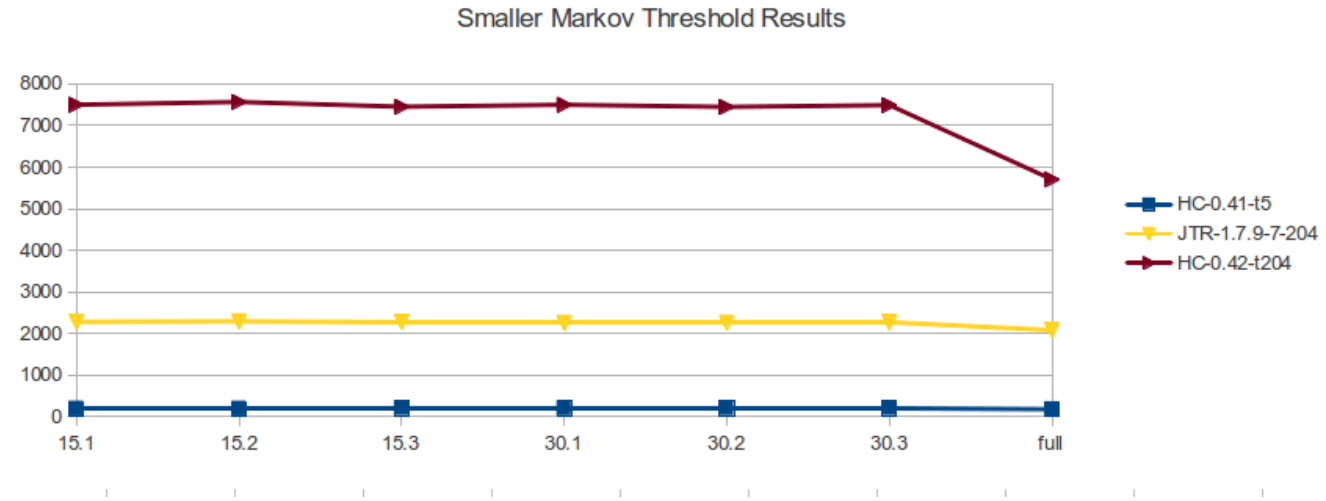
***UPDATE (4-Feb-2013)**: After I first put this up, I got some comments that I had not tested the situation where the passwords used in the training dataset were no longer in the hashlist. This is an extremely valid comment as this is the normal usage of the Markov mode - you use what you have to find more. So I have add in these tests at the bottom of the page. Take a look, the results are still interesting.*

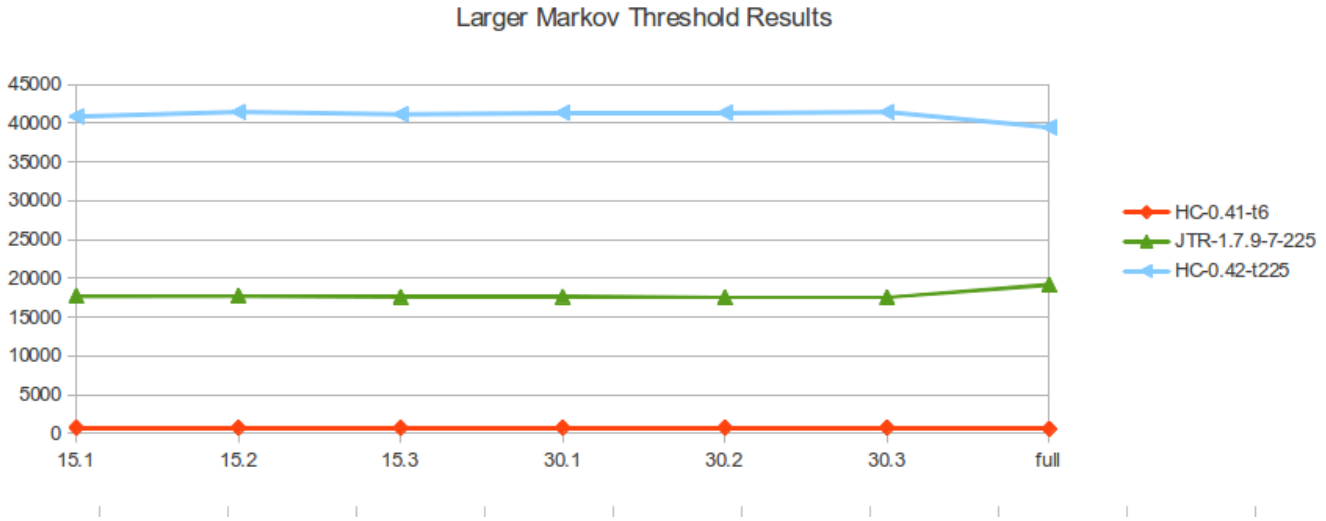
Recently I did a test on how the Markov mode implementation worked between my two favourite cracking tools, John the Ripper (JTR) and Hashcat (see here). At that time, JTR seemed to be the one finding the most passwords and relevant common words. After that, "atom" the brains behind hashcat was kind enough to allow me to run his updated version of the hashcat Markov utility and see how it performed. The results are below, and they are something. I used the same training data and hash list as previous to make the test results as fair as possible.

Test - Various training data sets (15% and 30% of total AND full list of found passwords)
In the table below you can see the original results followed by the results from the updated utility. First you will see that I had to greatly increase the "-t" variable in order to get it to generate the relevant amount of password attempts, but you can also see that it is now almost identical to the threshold value in JTR. Next. looking at the figures, you will see that the increase in found passwords is quite something. This is not an incremental improvement, this is going from a skateboard to a drag racer.

		15-1	15-2	15-3	30-1	30-2	30-3	full
hashcat 0.41 = --pw-max 12 -t 5	**found**	194	194	203	198	201	202	179
hashcat 0.41 = --pw-max 12 -t 6	**found**	745	730	716	742	750	750	581
1.7.9-7 = ./john -markov:204:0:0:12	**found**	2282	2294	2277	2262	2264	2268	2082
1.7.9-7 = ./john -markov:225:0:0:12	**found**	17727	17761	17596	17624	17549	17552	19172
hashcat 0.42 = --pw-max 12 -t 204	**found**	7501	7567	7452	7499	7448	7491	5703
hashcat 0.42 = --pw-max 12 -t 225	**found**	40867	41457	41156	41362	41352	41452	39470

Or to look at the results differently:





Test - Compare Total unique passwords found

Training Set	Cracked	Uniq	New Hashcat Utility Results
JTR-1.7.9-7 -> lvl 204 -> all 15% tests cases combined	6853	2455	
JTR-1.7.9-7 -> lvl 204 -> all 30% tests cases combined	6794	2370	
JTR-1.7.9-7 -> lvl 204 -> all tests cases combined	*15729*	*2808 - (17% of total is unique)*	
JTR-1.7.9-7 -> lvl 225 -> all 15% tests cases combined	53084	18737	
JTR-1.7.9-7 -> lvl 225 -> all 30% tests cases combined	52725	18263	
JTR-1.7.9-7 -> lvl 225 -> all tests cases combined	*124981*	*22633 - (18% of total is unique)*	
Hashcat 0.41 -> t5 > all 15% test cases combined	591	249	22520 cracked / 16324 unique
Hashcat 0.41 -> t5 > all 30% test cases combined	601	215	22438 cracked / 8259 unique
Hashcat 0.41 -> t5 > all test cases combined	*1371*	*326 - (23% of total is unique)*	50661 cracked / 17152 (33% of total) unique
Hashcat 0.41 -> t6 > all 15% test cases combined	2191	925	123480 cracked / 50136 unique
Hashcat 0.41 -> t6 > all 30% test cases combined	2242	834	124166 cracked / 47026 unique
Hashcat 0.41 -> t6 > all test cases combined	*5014*	*1077 - (21% of total is unique)*	287116 cracked / 58509 (20% of total) unique

..as you can see, the improvement is quite something. One thing you can see is that there seems to be more uniqueness in the hashcat runs then JTR. In hashcat.

Now lets do some comparing of the found password sets with one another. For this I created a single file for each tool that had all unique passwords it found in one file. So lets look at the overlap the new hashcat utility had with JTR:

```
# wc -l ./jtr-all.pass ./new-hc-all.pass
22633 ./jtr-all.pass
58509 ./new-hc-all.pass

# grep -xF ./jtr-all.pass -f ./new-hc-all.pass | wc -l
17232
```

..so unlike last time, there is a lot more overlap between the password attempts both tools tried and the passwords they found. Now lets look at the overlap between the old and new hashcat utility passwords:

```
# wc -l ./prev.hc-all.pass ./new-hc-all.pass
  1077 ./prev.hc-all.pass
58509 ./new-hc-all.pass

# grep -xF ./prev.hc-all.pass -f ./new-hc-all.pass | wc -l
249
```

..not a lot of overlap. This is another pointer to the impact that the update has had. Now lets run it through a dictionary analyzer and see what other changes the update made:

```
Top 10 base words
link = 260 (0.44%)
linkedin = 182 (0.31%)
linked = 91 (0.16%)
jack = 29 (0.05%)
mylink = 23 (0.04%)
pass = 18 (0.03%)
july = 18 (0.03%)
mylinkedin = 18 (0.03%)
linkme = 18 (0.03%)
work = 16 (0.03%)

Password length (length ordered)
6 = 9896 (16.91%)
7 = 12567 (21.48%)
8 = 21326 (36.45%)
9 = 8721 (14.91%)
10 = 2831 (4.84%)
11 = 1674 (2.86%)
```

```
12 = 1494 (2.55%)

Password length (count ordered)
8 = 21326 (36.45%)
7 = 12567 (21.48%)
6 = 9896 (16.91%)
9 = 8721 (14.91%)
10 = 2831 (4.84%)
11 = 1674 (2.86%)
12 = 1494 (2.55%)

         |
         |
         |
         |
         |
         |
        ||
        ||
       |||
       ||||
       ||||
       ||||
       ||||
       |||||
       |||||||
|||||||||||||
00000000001111
01234567890123

One to six characters = 9896 (16.91%)
One to eight characters = 43789 (74.84%)
More than eight characters = 14720 (25.16%)
```

Now the changes are even more apparent. We can see that the base words found is a lot more useful, there are multiple instances of link, linked, etc. Last time, it did not even find one. Also there are a lot more passwords found in the lower ranges of the password lengths, we can see a lot more 6-8 character passwords were found, last time there was hardly any in this range.

But in the end, the best way to look at it is "did it help me?". Last time I did the test, I took the full passwords found lists from each and tested them against my working copy of the hash list that I am working on. At that time JTR had no new passwords and Hashcat had 3. Since then I have worked through the list a bit more so I was interested to see how the passwords from this new hashcat utility fared:

```
>Recovered.: 11/862376 hashes, 0/1 salts
```

..it found 11 more passwords I had not yet cracked. Many of my conclusions from the first test remain the same, mainly that JTR and Hashcat do test different passwords, so use both. But I must honestly say that right now, leading with hashcat when you are starting is a good bet, the figures above show that. Hats off to Atom, the update is a great success.

UPDATE (4-Feb-2013):
What I want to test is how the Markov mode implementations work when the training data passwords are no longer in the hashlist. For my first test I started with the original list and did a quick bruteforce attack to get some training data:

```
# ../../hashcat/hashcat-cli32.bin -a 3 -1 ?l?d -m 100 --remove -o ./az19.dic ./unmasked.lst ?1?1?
1?1?1?1?1?1

# wc -l ./az19.dic
25321 ./az19.dic

# wc -l ./orig.unmasked.lst ./unmasked.lst
 2935345 ./orig.unmasked.lst
 2913286 ./unmasked.lst
```

Now lets train hashcat using our found passwords:

```
# ./hashcat-0.42/statsprocessor-0.09/hcstatgen.bin ./az19.hcstat < ./az19.lst
Processed input lines: 25322
Writing stats: ./az19.hcstat
```

Now we make sure we use the correct threshold values for hashcat to make the test fair:

```
# ./hashcat-0.42/statsprocessor-0.09/sp32.bin --pw-max 12 -t 225 ./az19.hcstat | wc -l
2547024178
```

```
# ./hashcat-0.42/statsprocessor-0.09/sp32.bin --pw-max 12 -t 204 ./az19.hcstat | wc -l
301278863
```

Now lets do the test, for the lower threshold of 204 using hashcat:

```
# ./hashcat-0.42/statsprocessor-0.09/sp32.bin --pw-max 12 -t 204 ./az19.hcstat > ../../stream
&

#./hashcat-0.42/hashcat-cli32.bin -m 100 -a 0 -o ./hc-042-t204-az9.pass ./unmasked.lst ../../
stream
Input.Mode: Dict (../../stream)
Index.....: 64/1 (segment), 2244317 (words), 18236266 (bytes)
Recovered.: 39/2913286 hashes, 0/1 salts
Speed/sec.: 35.24M plains, 35.24M words
Progress..: 2244317/2244317 (100.00%)
```

And if we do the same with a threshold of 225?:

```
#./hashcat-0.42/statsprocessor-0.09/sp32.bin --pw-max 12 -t 225 ./az19.hcstat > ../../stream
&

# ./hashcat-0.42/hashcat-cli32.bin -m 100 -a 0 -o ./hc-042-t225-az9.pass ./unmasked.lst ../../
stream
Input.Mode: Dict (../../stream)
Index.....: 591/1 (segment), 2886129 (words), 26443623 (bytes)
Recovered.: 2548/2913286 hashes, 0/1 salts
Speed/sec.: 39.16M plains, 39.16M words
Progress..: 2886129/2886129 (100.00%)
```

Ok, so in this test we got results of "39 found" and "2548 found" respectively.

	BF-training set
hashcat-0.42-t204	`39 found`
hashcat-0.42-t225	`2548 found`

Next, lets use JTR. Start with using the training dataset:

```
# ./calc_stat -p ../../az19.lst stats
```

Now lets do the test with the 2 thresholds:

```
# ./john -pot=./jtr-az19-204.pass -markov:204:0:0:12 ../../unmasked.lst
Loaded 2913286 password hashes with no different salts (Raw SHA-1 [128/128 SSE2 4x])
MKV start (stats=$JOHN/stats, lvl=204 len=12 pwd=318042917)
guesses: 0  time: 0:00:00:39 DONE (Sun Feb  3 18:58:24 2013)  c/s: 23757G  trying: 0qvf - 0

# ./john -pot=./jtr-az19-225.pass -markov:225:0:0:12 ../../unmasked.lst
Loaded 2913286 password hashes with no different salts (Raw SHA-1 [128/128 SSE2 4x])
MKV start (stats=$JOHN/stats, lvl=225 len=12 pwd=2302272063)
guesses: 29  time: 0:00:04:52 DONE (Sun Feb  3 19:09:13 2013)  c/s: 22969G  trying: 0q - 0
```

Lets update our table with those results:

	BF-training set
hashcat-0.42-t204	`39 found`
hashcat-0.42-t225	`2548 found`
jtr-1.7.9-7-t204	`0 found`
jtr-1.7.9-7-t225	`29 found`

But to really see the test properly I want to use one of my original datasets. So I am going to use "15-1", remove all those passwords from the original hashlist and see what results we get then. Lets start by getting the list and dictionary file ready:

```
# wc -l ./markov-train-15-1.txt ./orig.unmasked.lst ./unmasked.lst
  440301 ./markov-train-15-1.txt
 2935345 ./orig.unmasked.lst
 2935345 ./unmasked.lst

# ./hashcat-0.42/hashcat-cli32.bin -a 0 -m 100 --remove -o ./az19.dic ./unmasked.lst ./
markov-train-15-1.txt
Input.Mode: Dict (./markov-train-15-1.txt)
Index.....: 1/1 (segment), 440301 (words), 4613332 (bytes)
Recovered.: 440301/2935345 hashes, 0/1 salts
Speed/sec.: 20.07k plains, 20.07k words
Progress..: 440301/440301 (100.00%)
Running...: 00:00:00:22

# wc -l ./markov-train-15-1.txt ./orig.unmasked.lst ./unmasked.lst ./az19.dic
  440301 ./markov-train-15-1.txt
 2935345 ./orig.unmasked.lst
 2495044 ./unmasked.lst
  440301 ./az19.dic
```

Now lets train hashcat with our dictionary and test the 2 thresholds:

```
# ./hashcat-0.42/statsprocessor-0.09/hcstatgen.bin ./az19.hcstat < ./az19.lst
Processed input lines: 440302
Writing stats: ./az19.hcstat

# ./hashcat-0.42/hashcat-cli32.bin -m 100 -a 0 -o ./hc-042-t204-az9.pass ./unmasked.lst
../../stream
Input.Mode: Dict (../../stream)
Index.....: 66/1 (segment), 2217831 (words), 25569103 (bytes)
Recovered.: 5524/2495044 hashes, 0/1 salts
Speed/sec.: 33.43M plains, 33.43M words
Progress..: 2217831/2217831 (100.00%)

# ./hashcat-0.42/hashcat-cli32.bin -m 100 -a 0 -o ./hc-042-t225-az9.pass ./unmasked.lst
../../stream
Index.....: 608/1 (segment), 446781 (words), 5808153 (bytes)
Recovered.: 30705/2495044 hashes, 0/1 salts
Speed/sec.: - plains, - words
Progress..: 446781/446781 (100.00%)
```

With that, our tracking table looks like this:

	BF-training set	**15-1 training set**
hashcat-0.42-t204	39 found	5524 found
hashcat-0.42-t225	2548 found	30705 found
jtr-1.7.9-7-t204	0 found	
jtr-1.7.9-7-t225	29 found	

Now lets train JTR and do the 2 tests using that tool:

```
# ./calc_stat -p ../../az19.lst stats

# ./john -pot=./jtr-az19-204.pass -markov:204:0:0:12 ../../unmasked.lst
Loaded 2495044 password hashes with no different salts (Raw SHA-1 [128/128 SSE2 4x])
```

```
Remaining 2493240 password hashes with no different salts
MKV start (stats=$JOHN/stats, lvl=204 len=12 pwd=311170284)
guesses: 1804  time: 0:00:00:39 DONE (Sun Feb  3 23:31:02 2013)  c/s: 19899G  trying: ˇ - ˇ
Use the "--show" option to display all of the cracked passwords reliably


# ./john -pot=./jtr-az19-225.pass -markov:225:0:0:12 ../../unmasked.lst
Loaded 2495044 password hashes with no different salts (Raw SHA-1 [128/128 SSE2 4x])
Remaining 2481160 password hashes with no different salts
MKV start (stats=$JOHN/stats, lvl=225 len=12 pwd=2625949753)
guesses: 13884  time: 0:00:05:29 DONE (Sun Feb  3 23:43:18 2013)  c/s: 19853G  trying: ˇ - ˇ
Use the "--show" option to display all of the cracked passwords reliably
```

So at the end of the day, with both tests, in each case the passwords found and used were no longer in the hashlist, the figures are:

	BF-training set	**15-1 training set**
hashcat-0.42-t204	39 found	5524 found
hashcat-0.42-t225	2548 found	30705 found
jtr-1.7.9-7-t204	0 found	1804 found
jtr-1.7.9-7-t225	29 found	13884 found

So, yes, removing the passwords found in the training set from the hashlist does make a difference, but the general trend of the findings still remain:

1. Use both tools
2. Lead with hashcat for now
3. And the improvements Atom made are still obvious via the figures
4. This is not about which tool is "the one". My very first test showed JTR as the one to lead with, and now with Atom putting the work in, that is now hashcat. But in a few months, who knows? Each tool is under active development by very committed people.