MAKING DICTIONARIES

Brute forcing as previously discussed (see here) can still be useful even if inelegant, but one of the biggest problems a lot of people have is getting a decent and useful dictionary. But did you know that you can use a well-known brute-forcing program to make a dictionary. Remember "John the Ripper" (see here), well you can use that to create dictionary files. Take a look..

Whats Needed?
You will of course need "John the Ripper" aka JtR (homepage here) and a starter dictionary. I am going to start very small to show you how it works. Lets assume my dictionary file has one word in it..

```
# cat ./dict.lst
hello
```

Making the Dictionary
Now JtR uses certain rules to help it bruteforce passwords, what we are going to do is ask it to do everything it normally does, except output what it tries to the output. This is done with..

```
./john --wordlist=./dict.lst --rules --stdout
```

That will generate a lot of output, all of which ends in..

```
Hellos
helloed
helling
Helloed
Helling
words: 51  time: 0:00:00:00 100%  w/s: 2550  current: Helling
```

So we can see that from our 1 word, we could output 51 different variations. But wait there is more, this can be so much more fun. JtR allows you to add your own rules. Now I am not going to go through the how that works, partly because it is reasonably complicated and mostly because they have already done so in more then enough details (see here). But as an example, I want to add 4 rules,

1. To prefix all words with 123
2. To replace "o" with "0"
3. To replace "e" with "3"
4. To replace "o" with "0" and "e" with "3" at the same time

So lets do that..

```
# vi ./john.conf
--Go to the [List.Rules:Wordlist] section and add--
#PR: Prefix with 123
>2!?Al^3^2^1
#PR: Replace o with 0
>2!?Aso0
#PR: Replace e with 3
>2!?Ase3
#PR: Replace both
>2!?Aso0se3
```

Now lets run our dictionary creation again..

```
./john --wordlist=./dict.lst --rules --stdout
```

Now we see the output ends in..

```
helloed
helling
Helloed
Helling
words: 55  time: 0:00:00:00 100%  w/s: 2750  current: Helling
```

And in that 55 word list we see..

```
123hello
hell0
h3llo
h3ll0
```

So to get it into a file we just..

```
./john --wordlist=./dict.lst --rules --stdout > ./newdict.lst
```

Final Words
Now that is pretty cool. 55 variations off of one word. With any halfway decent dictionary file you can create a very nice hybrid dictionary and you can customise it any way you want. Take a look at writing your own rules, remember - have fun and learn,