FOR /F

tokens and delims

Step by step

The general syntax of `FOR /F` commands, at least the part we are going to analyze, is:

```
FOR /F "tokens=n,m* delims=ccc" %%A IN ('some_command') DO other_command %%A %%B %%C
```

Using an example, we are going to try and find a way to define values for *tokens* and *delims*.

For our example, we are going to find out who is logged on to a computer with a specified IP address (like, say, one found in our firewall logs).
The command we'll use is NBTSTAT.

```
NBTSTAT -A 10.100.0.14
```

will return something like:

```
Local Area Connection:
Node IpAddress: [your own IP address] Scope Id: []

           NetBIOS Remote Machine Name Table

       Name               Type         Status
    ---------------------------------------------
    REMOTE_PC      <00>  UNIQUE      Registered
    MYDOMAIN       <00>  GROUP       Registered
    REMOTE_PC      <20>  UNIQUE      Registered
    MYDOMAIN       <1E>  GROUP       Registered
    REMOTE_PC      <03>  UNIQUE      Registered
    REMOTE_USER    <03>  UNIQUE      Registered

    MAC Address = 01-02-03-44-5A-F1
```

We obviously need the information from the line that contains the string `<03>` but *not* the line with the computer name:

```
    REMOTE_PC      <03>  UNIQUE      Registered
    REMOTE_USER    <03>  UNIQUE      Registered
```

However, since we started with an IP address, in this case there is no way to distinguish between a computer name and a user name.
That's why we'll add another step:

```
NBTSTAT -a REMOTE_PC
```

will return the exact same result.
Note the *lower case* `-a` (`NBTSTAT /?` will show you the syntax in detail).

So if we know the remote PC name we know which line to filter out:

```
~~~~REMOTE_PC      <03>  UNIQUE      Registered~~~~
    REMOTE_USER    <03>  UNIQUE      Registered
```

We'll use PING to convert the IP address to its associated computer name:

```
PING -a 10.100.0.14 -n 1 -w 500
```

will return something like:

```
Pinging REMOTE_PC [10.100.0.14] with 32 bytes of data:

Reply from 10.100.0.14: bytes=32 time<10ms TTL=128

Ping statistics for 10.100.0.14:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms
```

What values for delims and tokens do we need to convert the IP address into its computer name?

Let's have a closer look at the output of the PING command:

- we want the (unknown) second word from the first line (actually, the second line, because the first line is blank)
- that first line contains the (known) IP address enclosed in square brackets [10.100.0.14]
- none of the other lines contain the IP address enclosed in square brackets, nor any other string in square brackets

First let's mark (~~underline, overline and strike-through~~) the boundaries of the requested word REMOTE_PC:

```
Pinging REMOTE_PC [10.100.0.14] with 32 bytes of data:

Reply from 10.100.0.14: bytes=32 time<10ms TTL=128

Ping statistics for 10.100.0.14:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms
```

This makes our choice for the delimiters, delims, quite obvious: a space.
If we mark (~~underline, overline and strike-through~~) all spaces, we can easily see which tokens are available:

```
Pinging REMOTE_PC [10.100.0.14] with 32 bytes of data:

token=1  token=2     token=3      4  5    6   7    8
```

In this case, we're only interested in tokens 2 and 3:

- token 2 is the requested computer name
- token 3 can be used to check if we're dealing with the correct line: it should equal our original IP address enclosed in square brackets

So we're only interested in the tokens 2 and 3:

```
Pinging REMOTE_PC [10.100.0.14] with 32 bytes of data:

         token=2     token=3
```

This leads us to the following command line:

```
FOR /F "tokens=2,3 delims= " %%A IN ('PING -a %1') DO IF "%%B"=="[%1]" SET PC=%%A
```

`%1` is the value of the first command line argument passed to our batch file.
In our case, the IP address to be investigated.

`IF "%%B"=="[%1]"` checks if the third word (token=3) equals the original IP address (%1) enclosed in square brackets ([%1]).
If we were to skip this test, the end result for token 2 would be the equal sign (=) from the last line (just try it).
If the test matches, the second word (token=2) is stored in a variable named PC.

Note that the *first token specified* (token 2) is stored in the variable specified (%%A), and the following token specified (token 3) in the following variable (in this case: %%B).

Our batch file thus far:

```
@ECHO OFF
FOR /F "tokens=2,3 delims= " %%A IN ('PING -a %1') DO IF "%%B"=="[%1]" SET PC=%%A
SET PC
```

The last line, `SET PC`, displays the actual value of the variable PC. I added it for debugging purposes.
(Actually, `SET PC` *will display* all *variables whose names* begin with *"PC".)*

Say we named our batch file FINDUSER.BAT then the command:

```
FINDUSER 127.0.0.1
```

should display your computer name:

```
PC=mycomputer
```

Now that we have the computer name, we can continue with the NBTSTAT command.

Let us mark (~~underline, overline and strike-through~~) the requested substring in NBTSTAT's output:

```
Local Area Connection:
Node IpAddress: [your own IP address] Scope Id: []

           NetBIOS Remote Machine Name Table

       Name               Type         Status
    ---------------------------------------------
    REMOTE_PC      <00>  UNIQUE      Registered
    MYDOMAIN       <00>  GROUP       Registered
    REMOTE_PC      <20>  UNIQUE      Registered
    MYDOMAIN       <1E>  GROUP       Registered
    REMOTE_PC      <03>  UNIQUE      Registered
    REMOTE_USER    <03>  UNIQUE      Registered
    MAC Address = 01-02-03-44-5A-F1
```

The choice for delims will be obvious: a space.

Notes: **(1)** Multiple spaces are still treated as a single delimiter.

(2) A row of characters in the delims definition is interpreted as "the first character *OR* the second character *OR* the third character" etcetera, so you can only use multiple single characters as delimiters in the delims definition, not entire "words".

The token number may be less obvious, since there are several spaces before the first word.
Since leading delimiters (before the first word) are ignored, however, it is still the first word in the line, so we need token 1.

Aside: We can use this feature to strip any number of leading spaces from a string:

```
FOR /F "tokens=*" %%A IN ("    some string") DO ECHO.%%A
```

will return `some string` (without the leading spaces).
And this isn't limited to spaces:

```
FOR /F "tokens=* delims=0" %%A IN ("00000012") DO ECHO.%%A
```

will return `12`

In this particular case, we will filter out the correct line not by checking the value of the second word (<03>) but by using the FIND command:

```
NBTSTAT -a %PC% | FIND "<03>" | FIND /I /V "%PC%"
```

will display only the line containing the user ID:

```
    REMOTE_USER    <03>  UNIQUE      Registered
```

(Remember? %PC% is the value for the remote computer name that we just got using the PING command).

To prevent error messages we need to escape the pipe symbols when we use them within brackets in a FOR /F command.
I will skip the details right now, just remember to place a caret before pipe and redirection characters when used within parentheses of FOR /F commands.

Our batch file now:

```
@ECHO OFF
ECHO IP=%1
FOR /F "tokens=2,3 delims= " %%A IN ('PING -a %1') DO IF "%%B"=="[%1]" SET PC=%%A
SET PC
FOR /F "tokens=1 delims= " %%A IN ('NBTSTAT -a %PC% ^| FIND "<03>" ^| FIND /I /V "%PC%"') DO SET USER=%%A
SET USER
```

Note the use of carets (^) as escape characters for the pipe symbols within the brackets of the second FOR loop!
Also note that no escape characters are necessary when the redirection characters are quoted, as in `FIND "<03>"`.

This batch file could do with some error checking, but as long as we pass it a valid IP address on the command line it should correctly return the IP address, the computer name, and the logged on user.

One could also use another `FOR /F` line, combined with the `NET USER` command, to retrieve the user's full name too, but I will leave that to you.

You got the idea.