Config Challenge

Please take no more than 3 hours to complete the following:

Every large software project has its share of configuration files to control settings, execution, etc. Let's contemplate a config file format that looks a lot like standard PHP .ini files, but with a few tweaks.

A config file will appear as follows:

```
[common]
basic_size_limit = 26214400
student_size_limit = 52428800

paid_users_size_limit = 2147483648

path = /srv/var/tmp/
path<itscript> = /srv/tmp/

[ftp]
name = "hello there, ftp uploading"
path = /tmp/
path<production> = /srv/var/tmp/
path<staging> = /srv/uploads/
path<ubuntu> = /etc/var/uploads
enabled = no

; This is a comment
[http]
name = "http uploading"
path = /tmp/
path<production> = /srv/var/tmp/
path<staging> = /srv/uploads/; This is another comment
params = array,of,values
```

Where "[group]" denotes the start of a group of related config options, `setting = value` denotes a standard setting name and associated default value, and `setting<override> = value2` denotes the value for the setting if the given override is enabled. If multiple enabled overrides are defined on a setting, the one defined last will have priority.

Assignment

Your task is to write a Ruby function: `def load_config(file_path, overrides=[])` that parses this format and returns an object that can be queried as follows. Note that overrides can be passed either as strings or as symbols: `CONFIG = load_config("/srv/settings.conf", ["ubuntu", :production])`

```
> CONFIG.common.paid_users_size_limit
```

```
# returns 2147483648

> CONFIG.ftp.name
# returns "hello there, ftp uploading"

> CONFIG.http.params
# returns ["array", "of", "values"]

> CONFIG.ftp.lastname
# returns nil

> CONFIG.ftp.enabled
# returns false (permitted bool values are "yes", "no", "true", "false", 1, 0)

> CONFIG.ftp[:path]
# returns "/etc/var/uploads"

> CONFIG.ftp
# returns a symbolized hash:
# {
#    :name => "http uploading",
#    :path => "/etc/var/uploads",
#    :enabled => false
# }
```

We'll be testing using Ruby 2.1.4 unless you tell us otherwise; if you have any features you think are important from a later version, please let us know what to look for!

Design Considerations

1. `load_config()` will be called at boot time, and thus should be as fast as possible. Conf files can get quite lengthy - there can be an arbitrary number of groups and number of settings within each group.
2. CONFIG will be queried throughout the program's execution, so each query should be very fast as well.
3. Certain queries will be made very often (thousands of times), others pretty rarely.
4. If the conf file is not well-formed, it is acceptable to print an error and exit from within `load_config()`. Once the object is returned, however, it is not permissible to exit or crash no matter what the query is. Returning nil is acceptable, however.