Errata for *Gray Hat Python* (updated to 13th printing)

Page 31: In the function `attach()` we removed the `self.run()` line

And in the function `open_process` we flipped the `False` and `pid` parameters so it should now read:

```python
h_process = kernel32.OpenProcess(PROCESS_ALL_ACCESS,False,pid)
```

Page 37: In the function `enumerate_threads()` we dedented the following two lines so as to align with the `while success:` line:

```python
kernel32.CloseHandle(snapshot)
return thread_list
```

And the declaration:

```python
def get_thread_context(self, thread_id ):
```

should now read:

```python
def get_thread_context(self, thread_id=None,h_thread=None)
```

And the `get_thread_context` function should include a check if the `h_thread` parameter isn’t passed to it:

```python
def get_thread_context(self, thread_id=None,h_thread=None):

    context = CONTEXT()
    context.ContextFlags = CONTEXT_FULL | CONTEXT_DEBUG_REGISTERS
    if not h_thread:
        self.open_thread(thread_id)
```

Page 42: In the function `get_debug_event()` the line that reads:

```python
self.context = self.get_thread_context(self.h_thread)
```

should now read:

```python
self.context = self.get_thread_context(h_thread=self.h_thread)
```
And we dedented all the lines from `if exception == EXCEPTION_ACCESS_VIOLATION: to print "Single Stepping."` so they are inside the `if debug_event.dwDebugEventCode == EXCEPTION_DEBUG_EVENT`

And we changed all instances of the `ec` variable to `exception`

And the function declaration `def exception_handler_breakpoint() needs to include the `self` parameter: `def exception_handler_breakpoint(self)`

**Page 44:** In the function `bp_set()`, the line:
`self.breakpoints[address] = address,original_byte)` should now read:
`self.breakpoints[address] = (original_byte)`

**Page 48:** In the function `bp_set_hw()` we dedented all the lines starting at `if available == 0: and ending at kernel32.SetThreadContext( h_thread, byref(context))` so that they are inside the preceding `for thread_id in self.enumerate_threads(): loop`