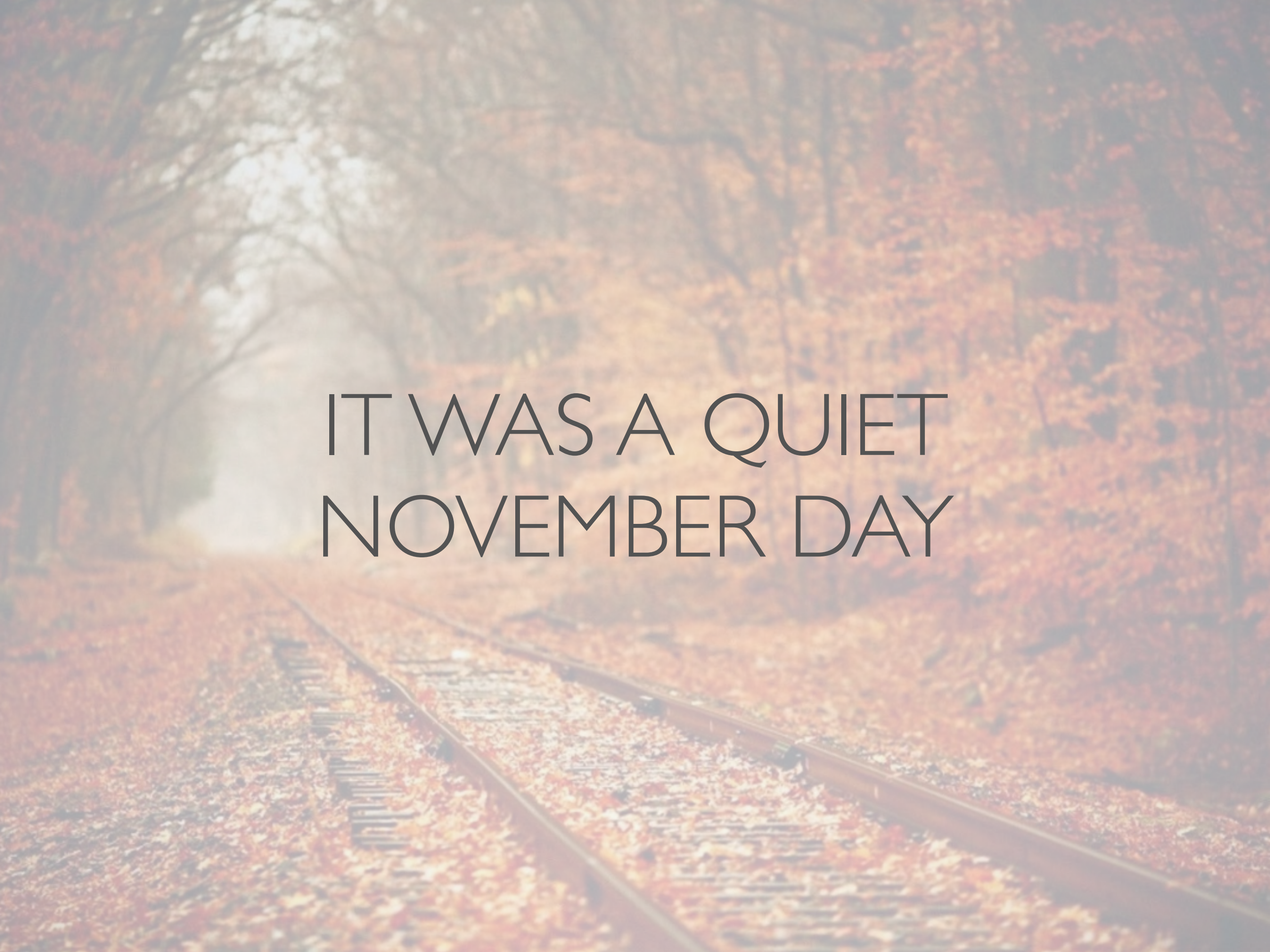
UNPACKING AN ANOMALY

Tales from the Ops Side

VM FARMS

@vmfarms

By Hany Fahim
Founder and CEO
@iHandroid


IT WAS A QUIET
NOVEMBER DAY

“Content is not a valid RSA or DSA SSH Key.”

AN EXCEPTION WAS THROWN

This error would haunt us for a month.

SSH KEY MANAGEMENT

- App in our portal to manage SSH keys.
- Allows one to map users and their SSH keys to servers.
- It also validates that the SSH key is valid and complete.

parse_sshkey()

```
1 try:
2     data_b64 = base64.decodestring(key)
3     int_len = 4
4     str_len = struct.unpack('>I', data_b64[:int_len])[0]
5     if str_len == 7:
6         key_type = data_b64[int_len:int_len + str_len]
7 except Exception:
8     raise ValidationError('Contents is not a valid RSA or DSA SSH Key.')
```

WHY DID IT BREAK?

MEMORY CORRUPTION?

- We thought perhaps it was memory corruption, as it did not make sense to break here.

- Tried to replicate, but failed.

- We thought this was an anomaly and would **never happen again**.

“Content is not a valid RSA or DSA SSH Key.”

6 DAYS LATER

It happened again.

THIS DIDN'T MAKE MUCH SENSE

- Due to the generic exception handler, the real exception was being swallowed up.
- Very little to go on here.
- Decided to add some extra logging and wait for it to happen again.

```
1  import logging
2  logger = logging.getLogger(__name__)
3  ...
4  try:
5      data_b64 = base64.decodestring(key)
6      int_len = 4
7      str_len = struct.unpack('>I', data_b64[:int_len])[0]
8      if str_len == 7:
9          key_type = data_b64[int_len:int_len + str_len]
10 except Exception:
11     message = 'Content is not a valid RSA or DSA SSH Key.'
12     logger.exception(message)
13     raise ValidationError(message)
```

MOAR LOGGING

“Content is not a valid RSA or DSA SSH Key.”

6 DAYS LATER

It was back.

NOW WE KNOW THE ERROR

```
TypeError: Struct() argument 1 must be string, not unicode
```

FROM THIS LINE

```
7 str_len = struct.unpack('>I', data_b64[:int_len])[0]
```

But it still didn't make sense

MORE CONFUSION

- What was more confusing is it should never error at this stage.

- This function is used for both validation and parsing of the SSH Key.

- This error was being thrown ***after*** validation.

- We could not replicate the issue. We tried 1,000,000 times.

DEEPER DIVE

- Looking through the error log, it seemed to error out nearly 100 times.
- Noticed a strange pattern in the logs.
- Approximately 1 in 8 calls were failing.
- We have 8 workers spawned.
- Is it possible a Gunicorn worker got corrupted somehow?

```
1  import logging
2  import os
3  logger = logging.getLogger(__name__)
4  ...
5  try:
6      data_b64 = base64.decodestring(key)
7      int_len = 4
8      str_len = struct.unpack('>I', data_b64[:int_len])[0]
9      if str_len == 7:
10         key_type = data_b64[int_len:int_len + str_len]
11 except Exception:
12     message = 'Content is not a valid RSA or DSA SSH Key.'
13     logger.exception(message, extra={'pid': os.getpid()})
14     raise ValidationError(message)
```

MOAR LOGGING

“Content is not a valid RSA or DSA SSH Key.”

7 DAYS LATER

It was back.

ASSUMPTION CONFIRMED!

- It was the same PID every time.
- A worker was getting corrupted. *Somehow*.
- Now what?

PROBLEM STOPPED.

SELF-HEALING POWERS

- The problem appeared to resolve itself while troubleshooting.
- This is something we never noticed previously.
- Perhaps there is some previous request that causes the corruption?
- Needs moar logging!

```
1 def post_request(worker, req, environ, resp):
2     timestamp = datetime.now()
3
4     worker.log.info('{method} {uri} status:{status} req_no:{req_no}
(CORRUPTION LOGGING)'.format(
5         method=req.method,
6         req_no=worker.nr,
7         uri=req.uri,
8         status=resp.status_code,
9     ))
```

MOAR LOGGING!

“Content is not a valid RSA or DSA SSH Key.”

5 DAYS LATER

It was back.

SAME STORY

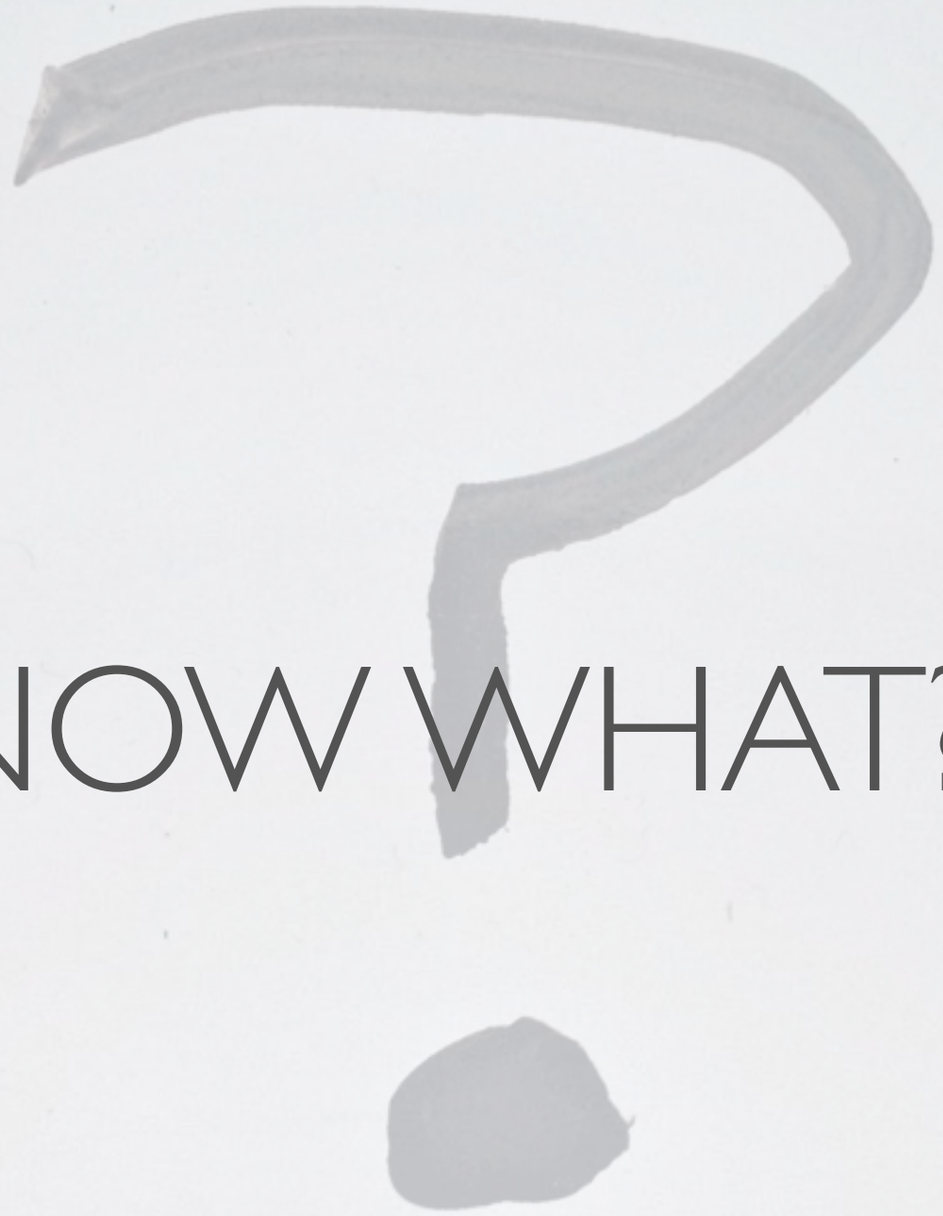
- Same PID.
- Slew of errors during a run, at an impossible stage.
- Stops on its own.
- Looked at our new logging.

GUNICORN LOGGING

- It did not appear to be any particular sequence of requests that caused the worker to break.

- When it did break, ***all*** calls to `parse_sshkey()` would break, for a short period of time, then fix itself ***without*** a restart.

- It did not appear to be related to the number of requests.



NOW WHAT?

WE WERE REACHING THE END

- We were out of ideas for debugging this further.
- Could not replicate this issue.
- Perhaps we should just replace calls to `struct` and abandon ship?

```
SSH_KEY_PREFIX = ['\x00', '\x00', '\x00', '\x07']

int_len = 4

try:
    assert list(data_b64[0:int_len]) == SSH_KEY_PREFIX
except AssertionError:
    message = 'Content is not a valid RSA or DSA SSH Key.'
    logger.exception(message, extra={'pid': os.getpid()})
    raise ValidationError(message)
```

NEW PR. SIMPLIFY.

No more `struct.unpack()`.

However, it was not merged.

“Content is not a valid RSA or DSA SSH Key.”

5 DAYS LATER

It was back.

BUG NEEDS TO BE SQUISHED

- PR hadn't been merged yet, but everyone was keen on resolving.
- Fresh eyes were on the problem.
- Important questions being asked.
- "Why does this ever work?"

```
1  from __future__ import unicode_literals
2  import logging
3  import os
4  logger = logging.getLogger(__name__)
5  ...
6  try:
7      data_b64 = base64.decodestring(key)
8      int_len = 4
9      str_len = struct.unpack('>I', data_b64[:int_len])[0]
10     if str_len == 7:
11         key_type = data_b64[int_len:int_len + str_len]
12 except Exception:
13     message = 'Content is not a valid RSA or DSA SSH Key.'
14     logger.exception(message, extra={'pid': os.getpid()})
15     raise ValidationError(message)
```

IT'S UNICODE!

EVERYTHING WAS UNICODE

- That import was added a few months back.

- It was part of a Django upgrade and an effort to be more Py3 compatible.

- This means…

THIS MEANS:

```
9 str_len = struct.unpack('>I', data_b64[:int_len])[0]
```

IS ACTUALLY:

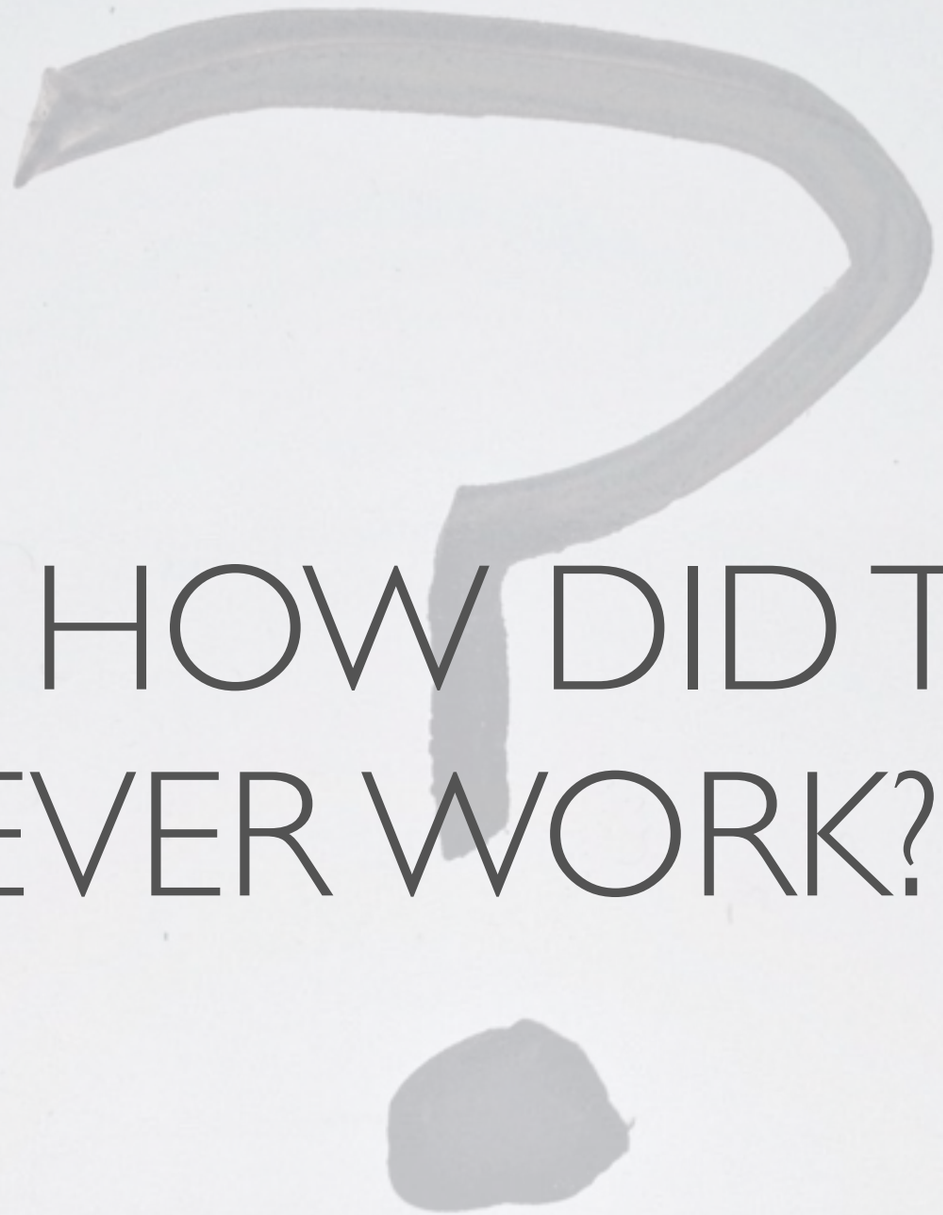
```
9 str_len = struct.unpack(u'>I', data_b64[:int_len])[0]
```

WITH IMPORT

```
>>> from __future__ import unicode_literals
>>> import struct
>>> struct.unpack('>I', '\x00\x00\x00\x07')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Struct() argument 1 must be string, not unicode
```

AND WITHOUT

```
>>> import struct
>>> struct.unpack('>I', '\x00\x00\x00\x07')
(7,)
```



BUT HOW DID THIS EVER WORK?

```
$ ./manage.py shell_plus
...
Python 2.7.6 (default, Nov 26 2013, 13:28:29)
Type "copyright", "credits" or "license" for more information.

IPython 5.0.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.
```

```
In [1]: from __future__ import unicode_literals

In [2]: import struct

In [3]: struct.unpack('>I', '\x00\x00\x00\x07')
Out[3]: (7,)

In [4]:
```

TRIED TO RUN WITHIN DJANGO

```
$ ./manage.py shell --plain
Python 2.7.6 (default, Nov 26 2013, 13:28:29)
[GCC 4.4.6 20120305 (Red Hat 4.4.6-4)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from __future__ import unicode_literals
>>> import struct
>>> struct.unpack('>I', '\x00\x00\x00\x07')
(7,)
>>>
```

WITHOUT IPYTHON

IS DJANGO MONKEY PATCHING `struct`?

No it wasn't…

3 STATEMENTS

1. This ***should*** break, ***every time***.

2. It ***breaks*** consistently outside of Django env.

3. It ***works*** consistently within Django env.

SOMEONE DISPROVED 3

```
>>> import struct
>>> struct.unpack('>I', '\x00\x00\x00\x07')
(7,)
>>> struct.unpack(u'>I', '\x00\x00\x00\x07')
(7,)
```

WHAT IS GOING ON?

IT ONLY FAILS IF THE FIRST CALL PASSES IN UNICODE

```
>>> import struct
>>> struct.unpack(u'>I', '\x00\x00\x00\x07')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Struct() argument 1 must be string, not unicode
>>> struct.unpack('>I', '\x00\x00\x00\x07')
(7,)
>>> struct.unpack(u'>I', '\x00\x00\x00\x07')
(7,)
```

CONFIRMED

WAS THERE SOME SORT OF CACHE?

From https://hg.python.org/cpython/file/3a1db0d2747e/Modules/_struct.c

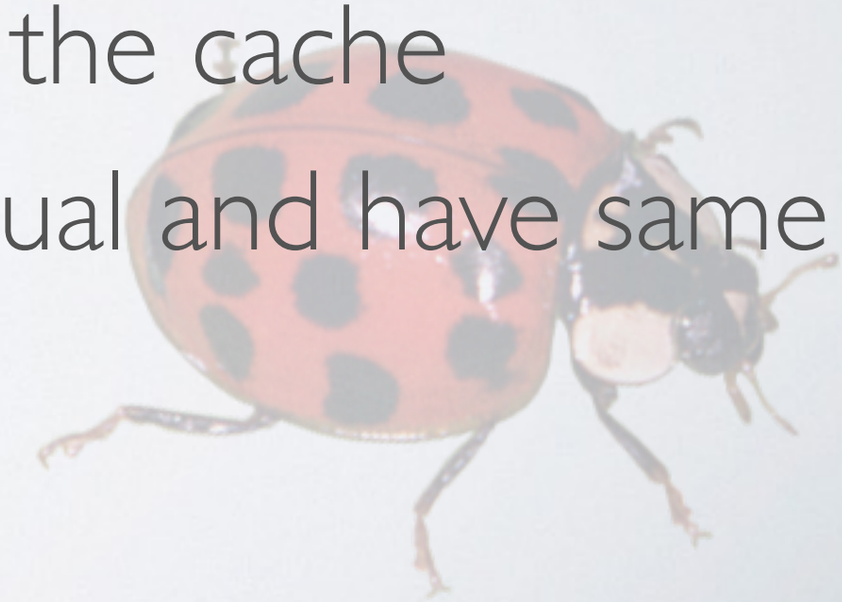
```
1  #define MAXCACHE 100
2  static PyObject *cache = NULL;
3
4  cache_struct(PyObject *fmt)
5  {
6  ...
7      if (s_object != NULL) {
8          if (PyDict_Size(cache) >= MAXCACHE)
9              PyDict_Clear(cache);
10         /* Attempt to cache the result */
11         if (PyDict_SetItem(cache, fmt, s_object) == -1)
12             PyErr_Clear();
13     }
14     return s_object;
15 }
```

YES!

AND THERE'S A BUG REPORT

- From https://bugs.python.org/issue19099

"Struct constructor accepts only `str` and not `unicode`. But `struct.pack()` uses caching and it found `Struct('B')` in the cache (because `u'B'` and `'B'` are equal and have same hash)."



```
$ pwd
/data/virtualenv/vmportal/lib
$ grep -r "'>I'" * | wc -l
77
```

'>I' IS USED AT LEAST 77 TIMES

PUTTING IT ALL TOGETHER

- On Gunicorn boot-up, `Struct('>I')` is called somewhere, and is cached.
- Our code works.
- Then the cache runs out.
- Under the right circumstances, our code will be called next, causing all subsequent calls to fail (and not be cached)…
- Stays broken until this worker processes other code that calls `Struct('>I')`.
- Then everything works again.

```
1  try:
2      data_b64 = base64.decodestring(key)
3      int_len = 4
4      str_len = struct.unpack(str('>I'), data_b64[:int_len])[0]
5      if str_len == 7:
6          key_type = data_b64[int_len:int_len + str_len]
7  except Exception:
8      message = 'Content is not a valid RSA or DSA SSH Key.'
9      logger.exception(message, extra={'pid': os.getpid()})
10     raise ValidationError(message)
```

SOLUTION WAS SIMPLE

Type-cast the argument to `str`

BUG FIX WAS IMPLEMENTED IN 2.7.7

- The very next version (https://hg.python.org/cpython/raw-file/f89216059edf/Misc/NEWS)

```
- Issue #19099: The struct module now
supports Unicode format strings.
```

LESSONS LEARNED

- When things don't make sense: Moar Logging.
- Keep questioning your assumptions. Always try to replicate.
- Be wary of:

```
from __future__ import unicode_literals
```

QUESTIONS?

Psst… we're hiring!

@vmfarms

By Hany Fahim
Founder and CEO
@iHandroid