

Datamining a cloud
that wasn't mean to be public
how I get my free amateur pr0n

Krunch
adrien@kunysz.be

DC4420,
London, UK,
November 2012

A long time ago on an IRC channel far, far away...

<Mr.Orange> Mr.White: [http://\\$SERVICE/p/14Rs/](http://$SERVICE/p/14Rs/)

<Mr.Orange> btw, Mr.White, we have this game in \$COMPANY called Clouderoulette; you take any given \$SERVICE link and increment it by one until you find something interesting

<Krunch> Mr.Orange: with $(10+26*2)^4$ possibilities i think indexing everything is feasible within hours or days

That's 14 776 336 combinations.

Wait, what?

How \$SERVICE works:

- ▶ user uploads file
- ▶ service serves file publicly with URL of the form
`http://$SERVICE/p/$BLOB/`

Observations:

- ▶ the blob is short (1-4 characters)
- ▶ the blob is actually a counter, not a hash or random value
- ▶ but it looks random if you don't know what "random" means

What Would \$DEITY Do?

20:50 <Krunch> Mr.Orange: \$SERVICE indexing now ongoing
with a hundred threads performing HEADs

20:50 <Krunch> User-Agent: Mr.Orange made me do it

23:22 <Krunch> still indexing but i think i won't need
bittorrent for a while

Technicalities

- ▶ count using `Math::BaseCalc([0..9, a-z, A-Z])`
- ▶ find the last value
- ▶ split the addresses space
- ▶ run HEADs in worker processes to retrieve each chunk
- ▶ write output in an `awk(1)`able format

Examples:

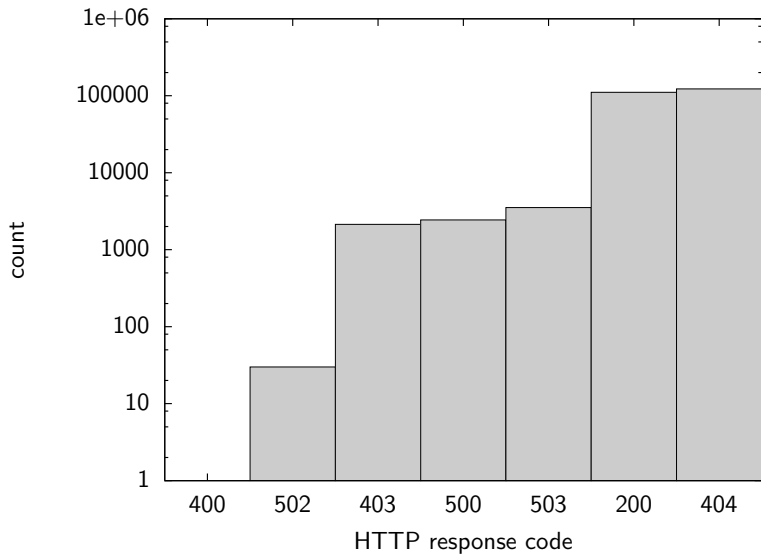
```
# path atime ret mtime      H/M sz      name
8 1310673767 200 1267080649 HIT 1969081 03 - Buy Me a Pony.ogg
fwc 1310682783 200 1298148693 MISS 6553936 anal cunt side 1 of
```

The URL for that audio file being `http://$SERVICE/p/8`

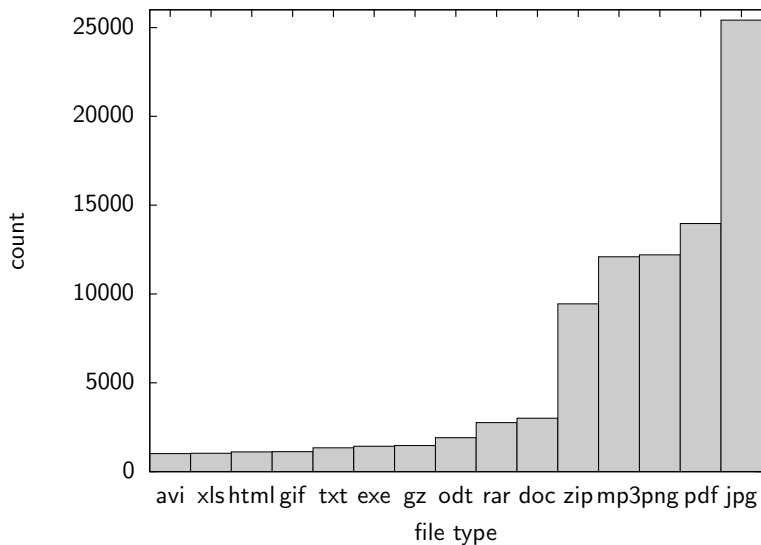
Lies, damned lies, and statistics

- ▶ indexed from 0 to 1NuC
- ▶ 241 993 URLs
- ▶ 111 070 actual files

HTTP response codes

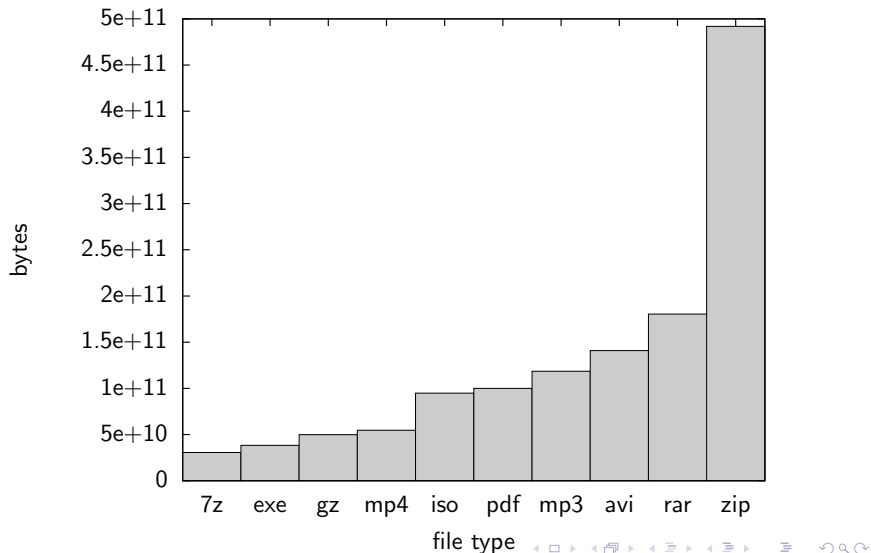


File types count



File types size

($5 * 10^{11}$ is about 500 gigs)



Finding 1: next file name is predictable

You can almost tail -f the upload logs.

Finding 2: rule 34 applies

<Krunch> 8L6 is german dinosaur porn

Finding 3: HIT/MISS header is not updated by HEAD requests

You can tell when someone accesses any given file.

Finding 4: people upload files in batch

```
il 1310680381 200 1271597102 HIT 169279 FICHES-RECETTES-MAROC.pd
im 1310680390 200 1271597104 HIT 164789 FICHES_RECETTES_THAILAND
in 1310680401 200 1271597110 HIT 170502 FICHES-RECETTES-VIETNAM.
io 1310680412 200 1271596925 HIT 24040656 V160410_13.010001.AVI
```

- ▶ upload some files that can identify you
- ▶ upload your anonymous sextape
- ▶ better wait at least a few hours between the two

Finding 5: \$SERVICE doesn't serve that much porn

- ▶ only 358 files that have a promising name
- ▶ kind of disappointing
- ▶ but if you like amateur, you'll still find stuff

Happy ending

- ▶ \$SERVICE people eventually realised there was a problem
- ▶ \$BLOB is now a long-ish random string
- ▶ insider knowledge suggests they were concerned about poor anonymisation (finding 4: people upload in batch)

Questions?

- ▶ `adrien@kunysz.be`
- ▶ Krunch on Freenode