2/2/2015

The RSA Cryptosystem (a crash course).

// you don't need to understand all the details for awhile.

There are three characters:

Alice: who has a secret number N to send to Bob.

Bob: who needs to know the secret

Eve: who listens to every word Alice & Bob say to each other.

The method

1) Bob prepares a "public key" as follows: — example

- he chooses two primes p, q — $p = 53$, $q = 79$
- he computes the product $m = pq$ — $m = 4187$
- he secretly computes $\varphi = (p-1)(q-1)$ — $\varphi = 4056$
- he randomly chooses a number k with $\gcd(\varphi, k) = 1$. — $k = 101$

Bob announces the numbers m and k to Alice (and doesn't mind if Eve hears them).

2) - Alice has her secret N. (the "plaintext") — $N = 163$
- She scrambles it by computing $N^k \% m$ (the remainder when N^k is divided by m).

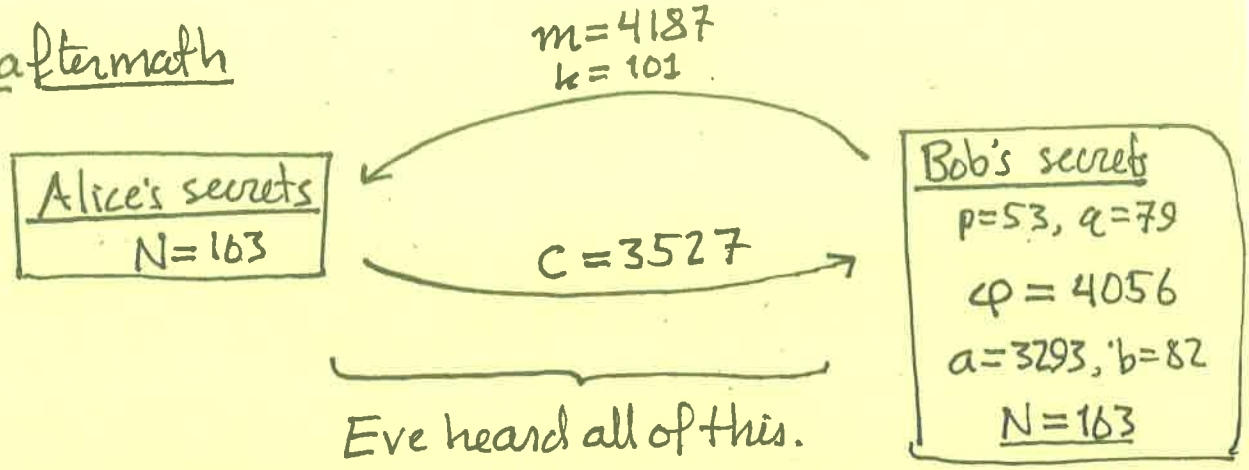
This is the "ciphertext" C. She announces C to Bob (and doesn't mind if Eve hears). — $C = 3527$

3) Bob solves the equation $k \cdot a - \varphi \cdot b = 1$. — $a = 3293$, $b = 82$

Then he unscrambles C by computing $C^a \% m$.

This is guaranteed to equal the secret N. (the plaintext). — $N = 163$

The aftermath



- If Eve manages to factor $m=4187$ into pq ($p=53$, $q=79$), she can decrypt C herself.
- If Eve ever finds a number $N < 4187$ with $N^{101} \% 4187 = 3527$, she'll know this was Alice's secret.

The math we need

1) Telling fast if a number is prime (Bob; to choose p, q).
2) Computing $N^k \% m$ fast, without writing out N^k in full. (both Alice and Bob).
3) Solving $k \cdot a - \varphi \cdot b = 1$ given k and φ (Bob).

(we'll discuss (3) & (2) this semester; possibly (1) as well)

The math we hope Eve doesn't have

- Factoring a large number m into $p \cdot q$.

The elephant in the room

Why does this work?

```
   p = 53
   q = 79
   p*q
=> 4187
```

p = 53, q = 79 } Bob keeps these secret.

p*q => 4187 ← this is m. Bob announces it.

```
   (p-1)*(q-1)
=> 4056
   gcd(101,4056)
=> 1
   euclid2(101,4056)
=> [3293, -82]
   101*3293
=> 332593
   82*4056
=> 332592
```

(p-1)*(q-1) => 4056 ← this is φ. (secret)

gcd(101,4056) => 1 ← Bob does this to make sure 101 is a good choice (if gcd ≠ 1, he just randomly picks another k).

euclid2(101,4056) => [3293, -82] ← these are a and b. Bob uses them to decrypt.

101*3293 => 332593

82*4056 => 332592

Note that $\underbrace{101}_{k} \cdot \underbrace{3293}_{a} - \underbrace{4056}_{\varphi} \cdot 82 = 1.$

```
   pow(163,101)
=>
```

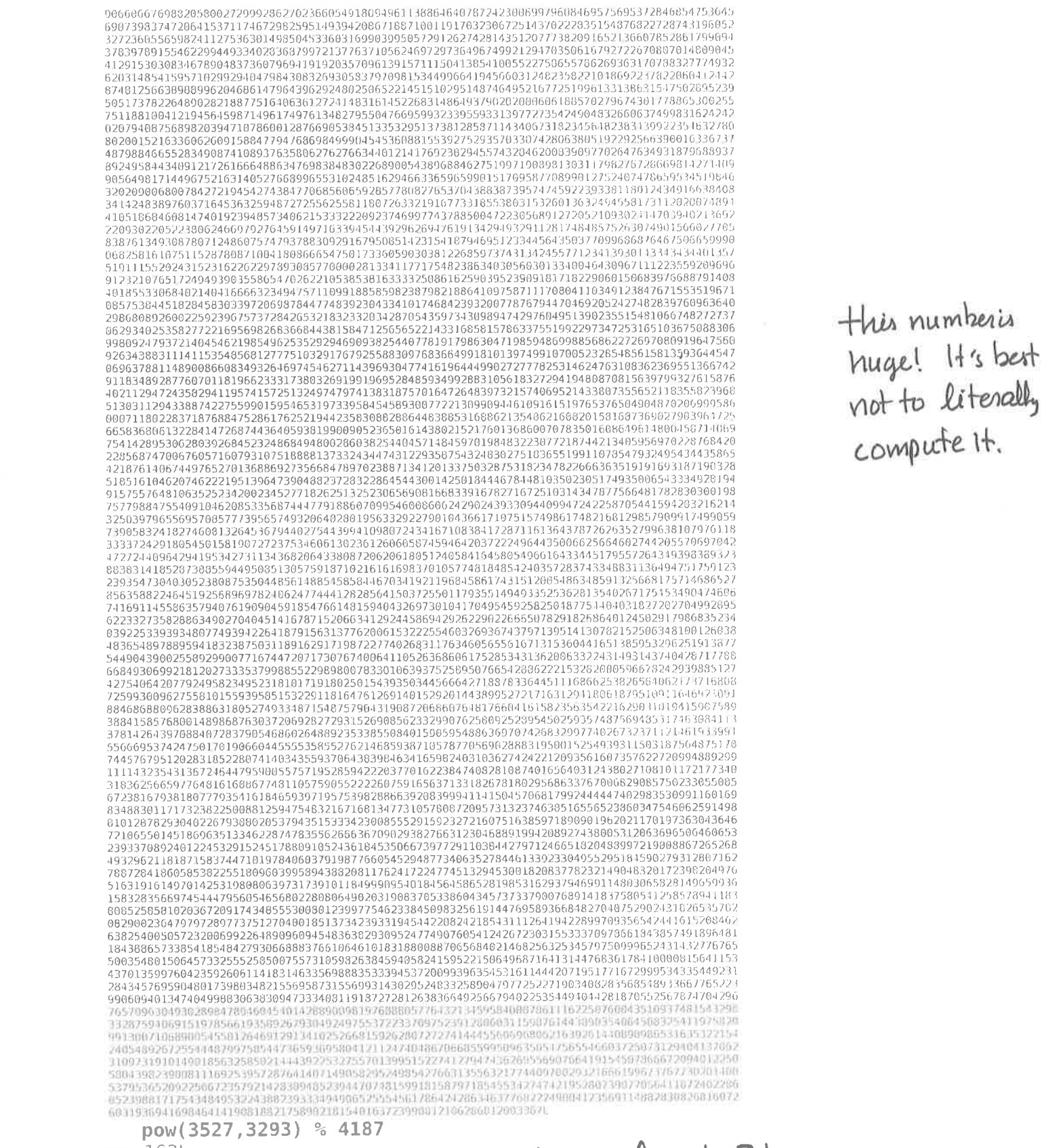
```
269741671775526705545231578863416099096004559124217743
150138872308989997061977903386947641581774348592218586
001307706581921068938052035352544240666272210437068515
959397486008076146947962992635597701097571235886208403
41462163L
```

```
269741671775526705545231578863416099096004559124217743
150138872308989997061977903386947641581774348592218586
001307706581921068938052035352544240666272210437068515
959397486008076146947962992635597701097571235886208403
41462163 % 4187
=> 3527L
```

plaintext N = 163 gives ciphertext C = 3527 ←

```
   pow(3527,3293)
=>
```

```
403811501995312386480954764987826380849182036976612816005949295909328166942850156268754399280689
556234666418036067905188246063504160430497922928824134887216625840280607083515213482141092859081
309483018342679814808410709954086951030848891729958617493642208029644604436919595573021794141229
338257365626792447094662501483795262881828697002342183858658245365905114910224954342891906443039
616419432968634569740140448173809304328870570046294762543343195030020698614957566189966052744181
948960994171277865415538691693567345692539379499769849102953786771749215462469424375017420102076
996443815392893008605456147177375539330523664025686077282688592032845342673387624570866574630700
418437004471555716789656027549976570399962815592208942210498579646157688480096578190725267447907
587801781164812913830913342264610510094598078014501667382386927658731311259653569331646033757366
484765176049028206745928094393376030666872716919999576760218193702200393015253960735224367199959
048806442153375816069644175045999476138126926730277291535159138191297068711105251448150496379004
066728591722933384818312126666534408157621521920120457879813332b451538110559260069789572963296
895504332695630228708212287763244011579004463117889612334653509810077852380249230738026995297614
427233121463939360127693709992235322654745376843161068706189559585864505671170682808838896770139
540989232353926142663845844936003328498689992450149138607050277339362828710081077598004519892219
931713377826905159032340534484616868420461173834911374667874092495963904345693452039006095870127477
389354507153046752676502286246637692498715663918518774627413960417045442936959158136854861169268
293157648232290460884809176836052394394252037172383279903606520117181663011040175759676666694744
436820272087841773692129707280473408872582310300775458295338965217557697649907519536367450289228
544834536041207104362094741129163441662350187534780538793762227538742271067245283810398252948152
027166815679746356767024812431055906392588461880045767785771250974688550415333804781887831917549
865474277482546313927925769983180703716007886759343785638967896797379461689732625483754405097623
173272960624395874116694897755619844882892196791755577746405169215230707983230560542932842231744
760134258588655165593224963526259517366532648657554584789361382304162668457984189544600969020570
699072635672447881498015965680035753620351522961066348000933753130845694380254760243845278284630
191587386548580168066468601831314575062466191138784136765524957760936128862199531876570316313086
747239897112137863693431549524381887249857671523891703073600407839931893022623457554211882716033
416772807322460106607223588450472387246780438705589703428177301644126073247734775435587789380502
548306000934992417500574464714561809556717061654654208683485385612249112932594383828994886542440
616962913774345753755805222552783296359175020615337390137002233866322397237605923364137334369190
842931646835451560925936049006107794399265686409087291724976855071460089179500908444947246594035
810661769287193155498814174121481486495650212389798115010018723969369350473117724261587372215949
003185837242311305260256081412619241134499842536749490556768035405197008124055407983658666616672
552783138969522659445718839973686455684716626266292307206381821477393841661503957016294213054388
```



this number is huge! It's best not to literally compute it.

```
pow(3527,3293) % 4187
=> 163L
```

← from it Bob recovers the plaintext.