Dual_EC_DRBG

or, the story of a not so random backdoor

Martijn Grooten
Virus Bulletin, UK

Martijn Grooten (*Μαρτάιν Γρόουτεν*)

Mathematics

IT Security

Civil liberties

Disclaimer: I am *not* a cryptographer

A backdoor in software

```
...

if ( PORT == 1337 && password == "roodkcab" ) {
  /* support access */
  access_level = "admin";
}

...
```

A weakness in a standard

For Authenticated Encryption in the data plane, AES with 128 bit keys in GCM mode with 128 bit ICV MUST be used. For Integrity checks (when Authenticated Encryption is not in use), HMAC-SHA-256-128 MUST be used. For hashing algorithms, SHA-256 MUST be used. For certificate based signatures, RSA-2048 and SHA-256 MUST be used. For Diffie-Hellman key exchanges, a 2048-bit MODP group MUST be used. Explicitly, Diffie-Hellman Group 14 MUST be used. For pseudo-random generation function, PRF-HMAC-SHA-256 MUST be used.

A weakness in a standard

For Authenticated Encryption in the data plane, AES with 32 bit keys in GCM mode with 32 bit ICV MUST be used. For Integrity checks (when Authenticated Encryption is not in use), HMAC-SHA-128-64 MUST be used. For hashing algorithms, SHA-64 MUST be used. For certificate based signatures, RSA-512 and SHA-64 MUST be used. For Diffie-Hellman key exchanges, a 512-bit MODP group MUST be used. Explicitly, Diffie-Hellman Group 14 MUST be used. For pseudo-random generation function, PRF-HMAC-SHA-64 MUST be used.

A backdoor in a standard?

Introducing: Dual Elliptic Curve Deterministic Random Bit Generator (Dual_EC_DRBG)

NIST SP 800-90A **January 2012**

Acknowledgements

The National Institute of Standards and Technology (NIST) gratefully acknowledges and appreciates contributions by Mike Boyle and Mary Baish from NSA for assistance in the development of this Recommendation. NIST also thanks the many contributions by the public and private sectors.

virus BULLETIN

A bit of history

Public key cryptography (RSA, DH; 1970s)

$$n = p \cdot q$$

public (n) private (p, q)

$$a = x^n \mod p$$ (discrete logarithm problem)

Crypto Wars (1990s)

The crypto wars were won…

…but the battle went on underground

NATIONAL SECURITY AGENCY
UNITED STATES OF AMERICA

GCHQ

virus BULLETIN

Encryption

Encryption is “the process of encoding messages or information in such a way that only authorized parties can read it.” (*Wikipedia)*.

“An encryption scheme usually needs a key-generation algorithm to randomly produce keys” (*Wikipedia*)

virus BULLETIN

Encryption

Encryption *generates* data that is indistinguishable from random.

Encryption needs random data *as input*.

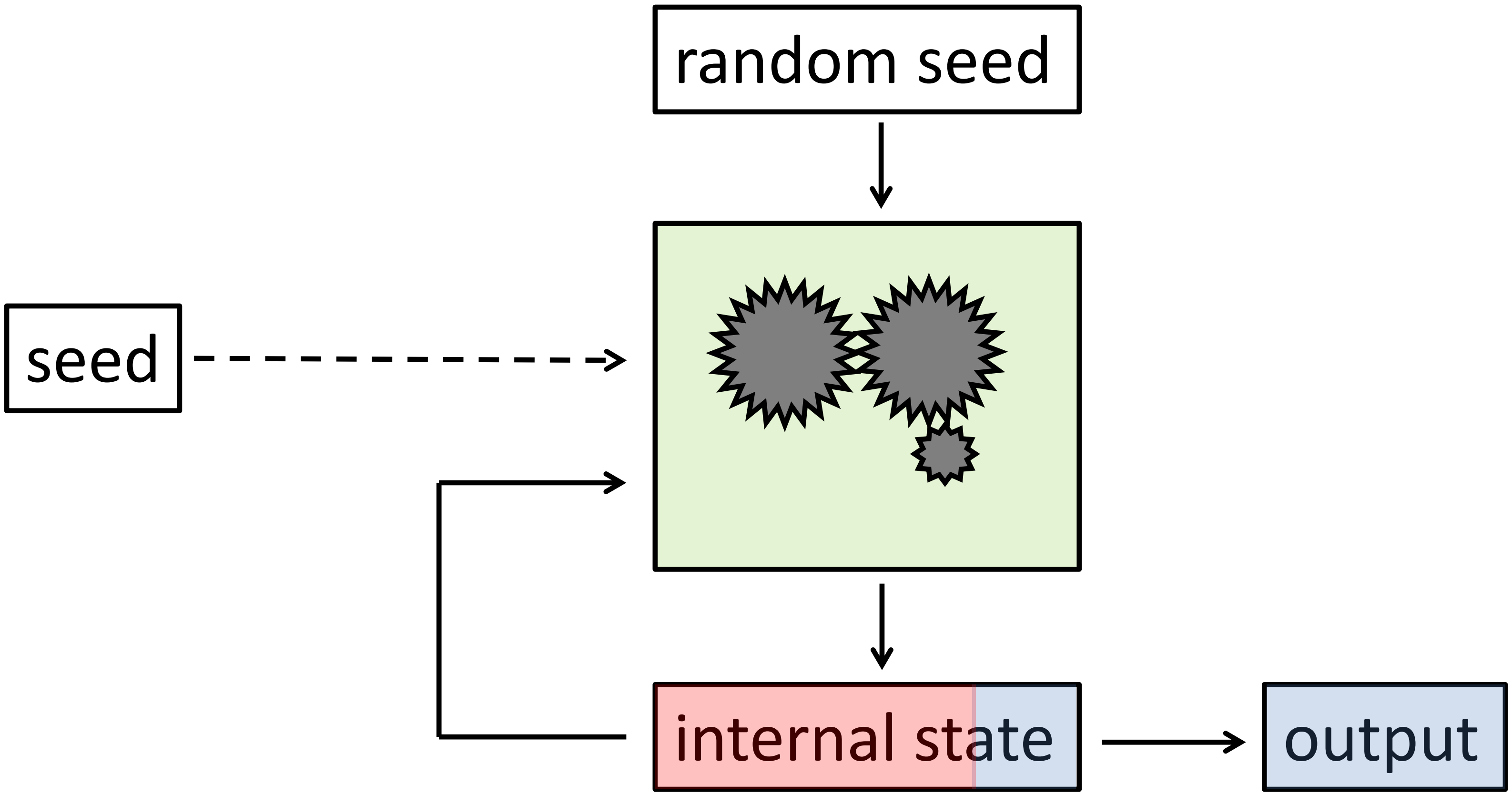
Randomness

Getting good randomness with enough entropy ('surprise') is hard.

It is not impossible though.

But getting *enough* good randomness is without some extra help.

Deterministic random bit generator



Did I say randomness is hard?

Random number generators are a *major weakness* in many cryptography implementation.

Blindly trusting someone else's RNG is a bad idea (as we will see).

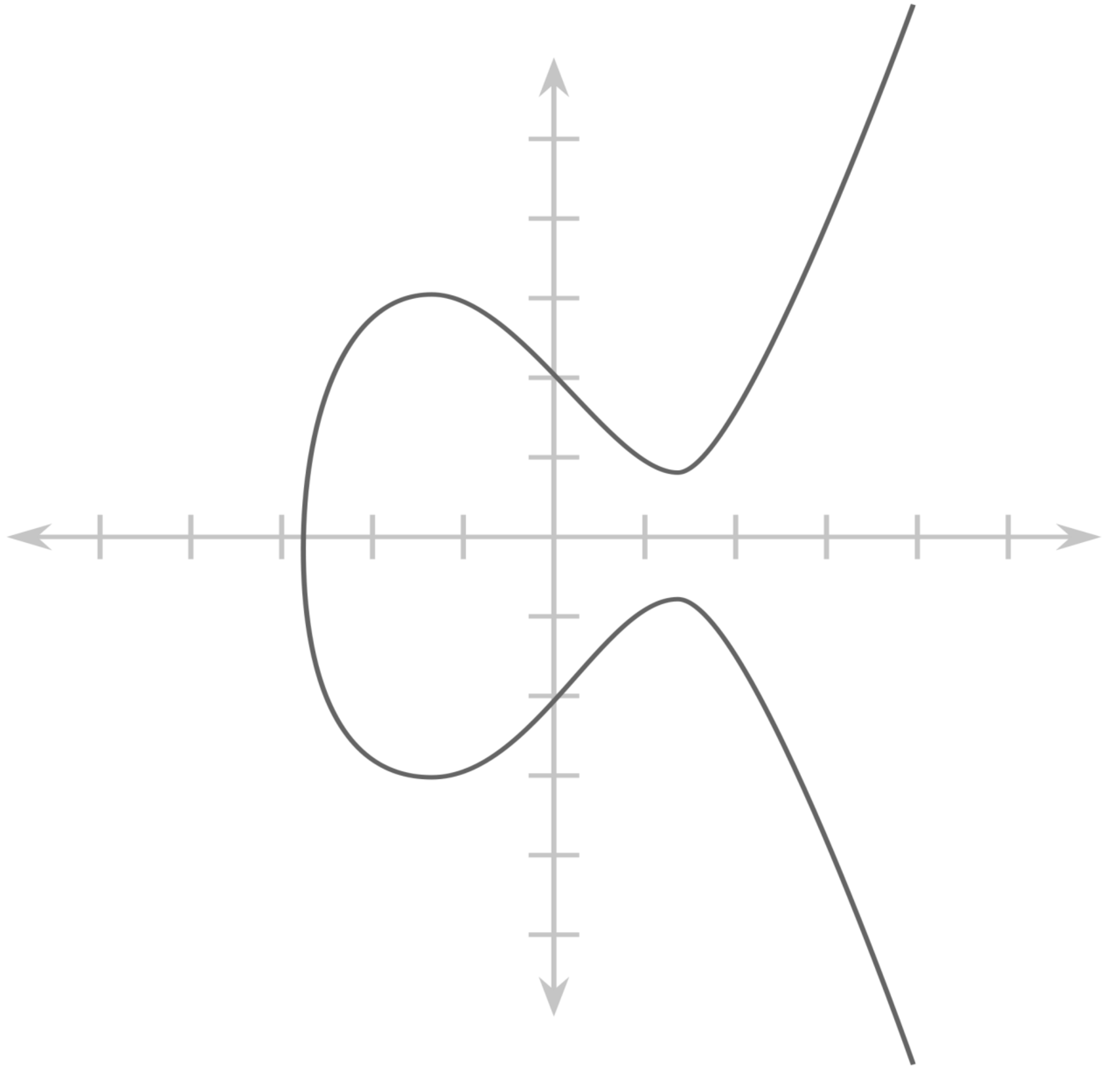
But writing your own is worse.

Elliptic curves

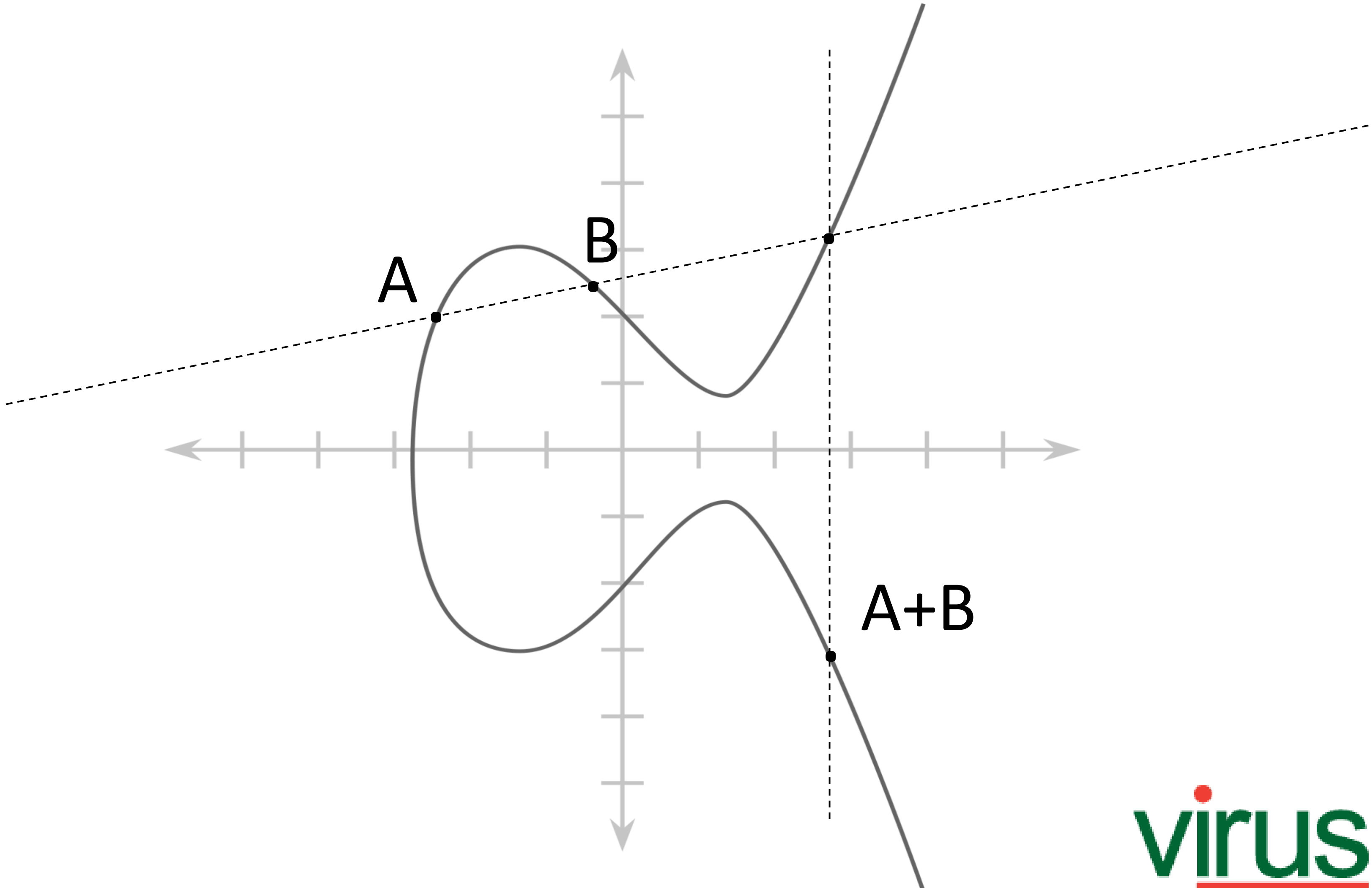
Solution to third degree equation without singularities in the projective plane over a field.

$y = x^3 + a \cdot x + b$ for some given *a* and *b*

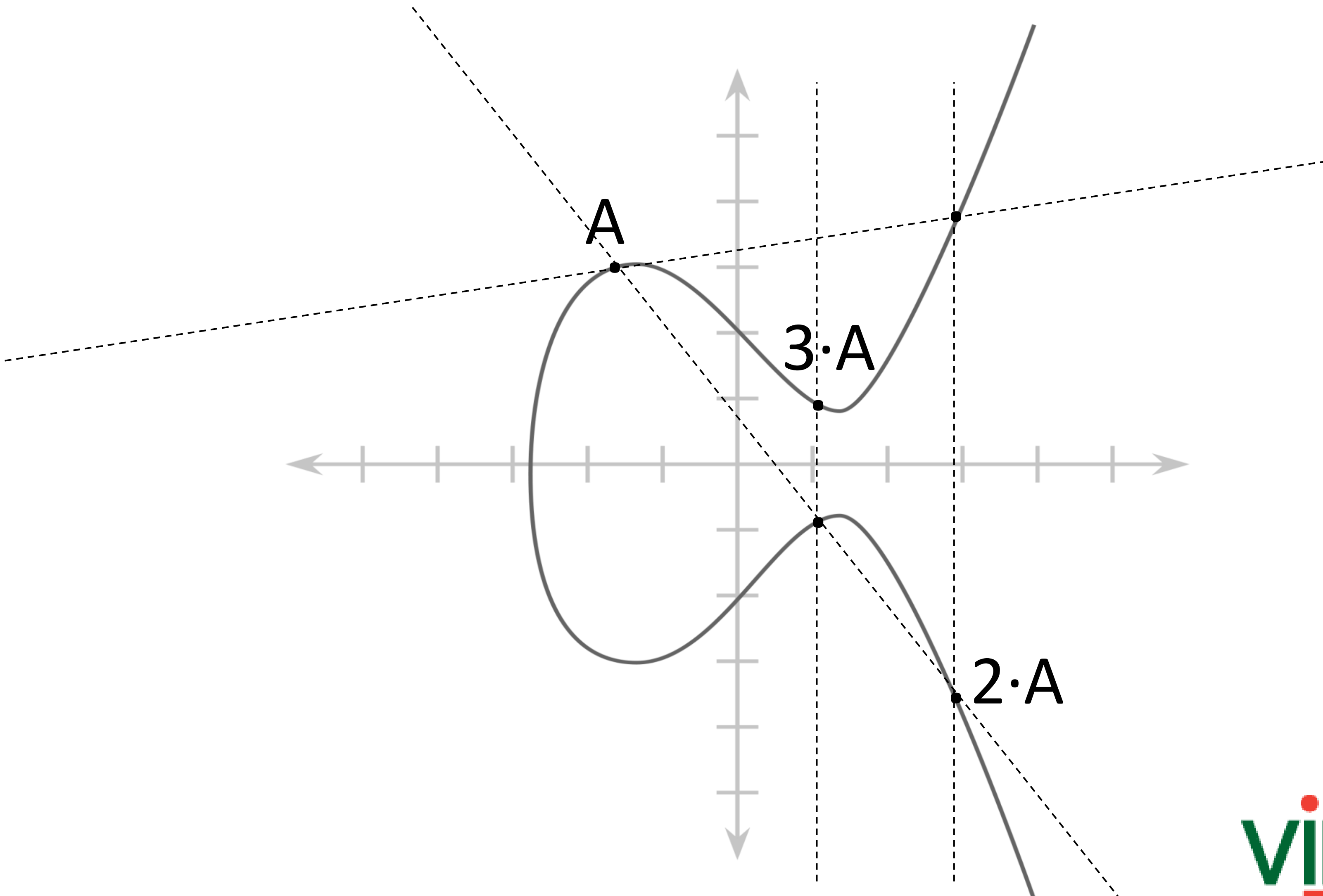
Elliptic curves



Point addition on elliptic curves



Point addition on elliptic curves



Discrete logarithm problem on elliptic curves

Given a point A and a (large) number n, there is – under certain circumstances – a *unique* point B such that $A = n \cdot B$.

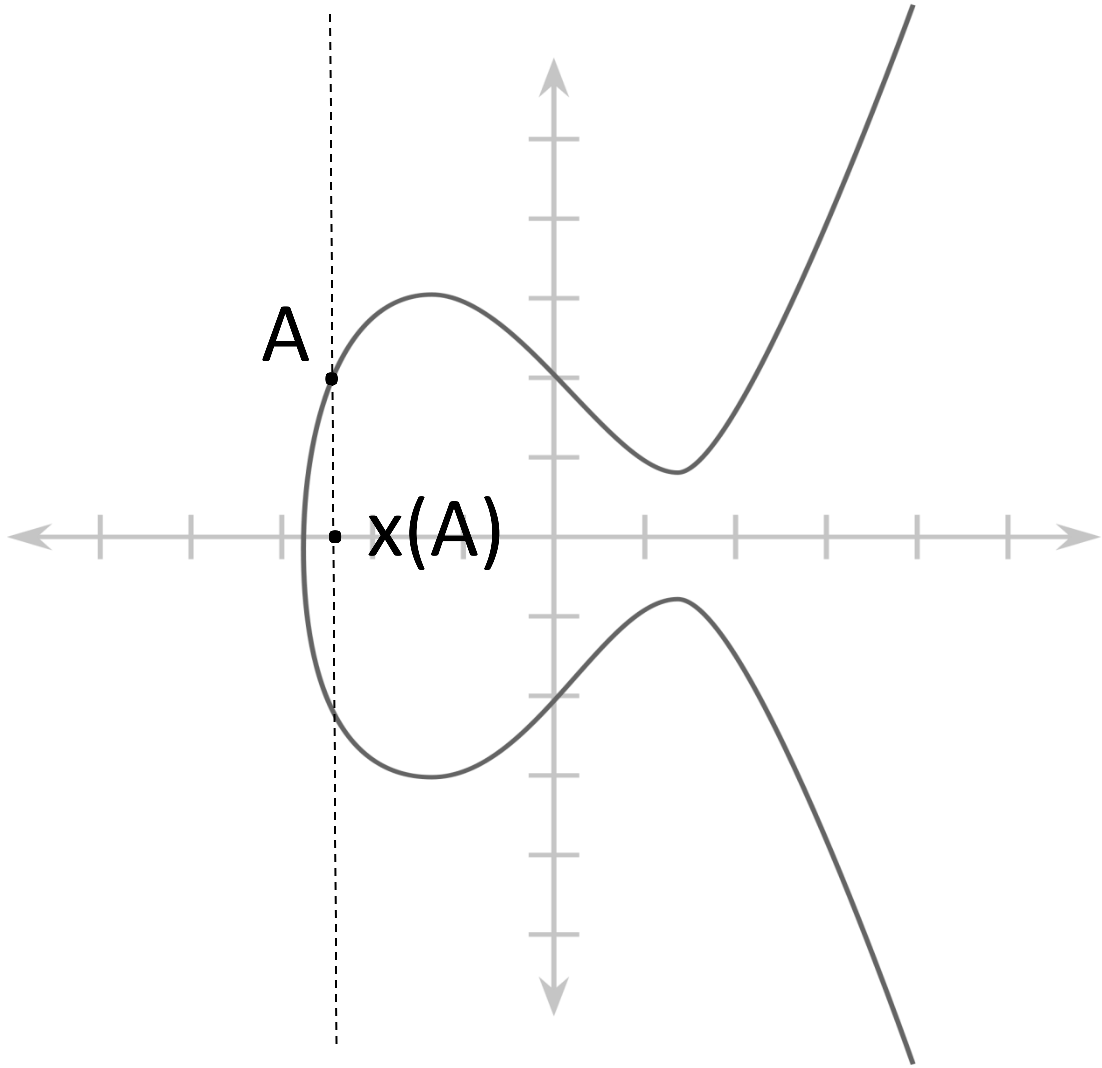
Finding B is fiendishly difficult.

This is very useful for cryptography

… or to backdoor a standard.

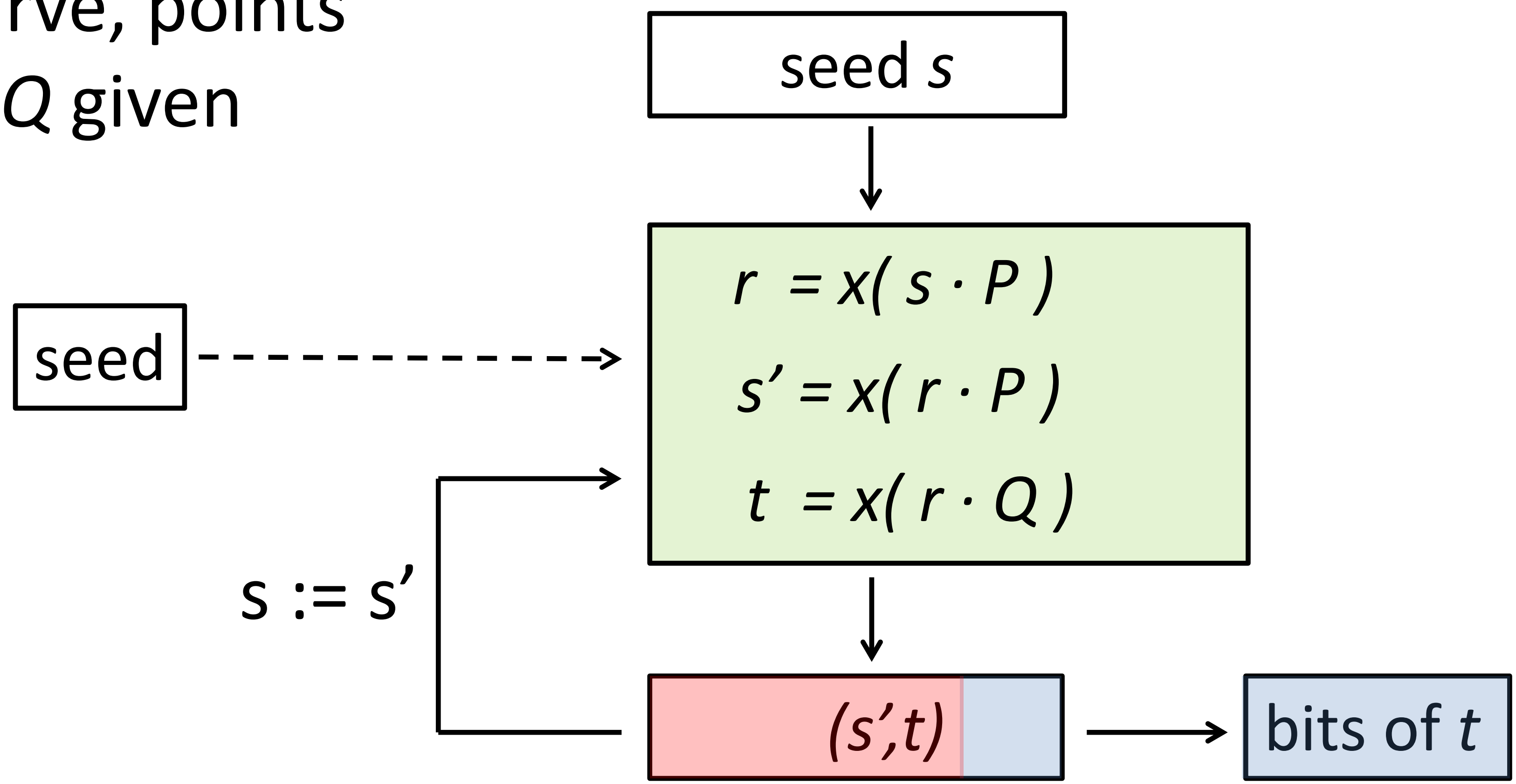
virus BULLETIN

Turning points into numbers



Dual_EC_DRBG

Curve, points
P, *Q* given



Finding good *P and Q* isn't trivial

Using wrong *P* and *Q* could break the algorithm. Thankfully, the NIST standard provides them for us:

"The security of **Dual_EC_DRBG** requires that the points *P* and *Q* be properly generated. To avoid using potentially weak points, the points specified in Appendix A.1 **should** be used."

So how ‘random’ are *P* and *Q*

Fact: given *P* and *Q* the exists a number *e* such that $P = e \cdot Q$.

Remember: *e* is fiendishly hard to find. (Discrete logarithm problem.)

But if you can choose *P* and *Q*, you can do so that you know the number *e*.

Does it matter if you know *e*?

Ferguson, Shumow (2007): it bloody well does.

Knowledge of *e* makes the output of the random number generator *trivial to predict*.

This means Dual_EC_DRBG is **very unsafe** against anyone who knows *e*.

virus BULLETIN

Facts about in Dual_EC_DRBG?

Regardless of the backdoor Dual_EC_DRBG is a rather bad idea (H/T Matthew Green).

Dual_EC_DRBG is (assumed) safe against any adversary who doesn't know the number *e*.

We don't know for sure if anyone knows this number!

(But *I* think the NSA does.)

Why there might not be a backdoor

It is too simple, too clumsy.

$10m allegedly paid to RSA to implement Dual_EC_DRBG in *Bsafe* is not a lot of money.

Why there might be a backdoor

$10m allegedly paid to RSA is a lot of money.

P and *Q* are not explained.

It is widely used despite being a bad idea.

Snowden etc.

Conclusion

Cryptography is *very hard*. This is its biggest weakness.

Keep checking existing standards and implementations. Reject if unsure about certain things.

Thanks

Questions or comments?

Contact:

martijn.grooten@virusbtn.com

@martijn_grooten

www.virusbtn.com www.lapsedordinary.net