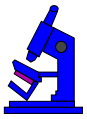
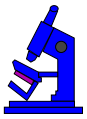


Chapter 2

Lecture 2 - Preliminaries

Last session - movie night!

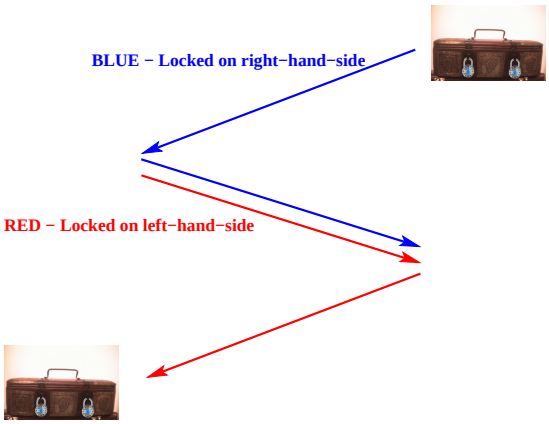
- ✔ Introduction, setting context
- ✔ Definitions
- ✔ Cæsar cipher, Enigma, Secure shell
- ✔ Insecurity

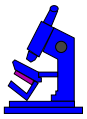


This session

- Finish context
- Math preliminaries
  - XOR
  - Logarithms
  - Fields and groups

Diagram for lock protocol





Safety/control software

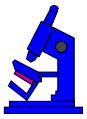
A naive approach to security might involve attempting to ensure that all programs that run on a computer are *safe*, and that all users of computer systems are *trustworthy*.

Checking even one program is a non-trivial task.

The computer *operating system* normally provides some level of software and hardware security for computer systems, combined with some level of user authorization.

Safety/control software

- **User authorization** means passwords!
- Systems have grown in complexity over the years.
- An article shows the changes in the UNIX mechanism



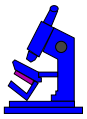
Hardware security

Hardware security in operating systems has been studied in CS2106 (Operating Systems) and other courses. The Kernel/Supervisor bit, processor ring0, memory protection/mapping hardware and so on are all examples of hardware security systems intended to co-operate with the OS to enhance system security.

Software security in operating systems takes many forms. The forms range from ad-hoc changes to operating systems to fix security loopholes as they are found, through to operating systems built from the ground up to be secure.

Example: network security

- TCP wrappers:
  - Attacks through poorly controlled TCP or UDP ports.
  - Wrapper provides single point of control
  - Default installation disables *all* access
  - Re-enable on a case-by-case basis.



OS security

NSA have a security-enhanced Linux system:

> *This version of Linux has a strong, flexible mandatory access control architecture incorporated into the major subsystems of the kernel. The system provides a mechanism to enforce the separation of information based on confidentiality and integrity requirements.*

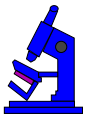
You can read about SELinux at

http://www.nsa.gov/selinux/index.html

OS security

- Java virtual machine has built-in security model
- Microsoft point out that the Linux security model is weak...

> *Every member of the Windows NT family since Windows NT 3.5 has been evaluated at either a C2 level under the U.S. Government's evaluation process or at a C2-equivalent level under the British Government's ITSEC process. In contrast, no Linux products are listed on the U.S. Government's evaluated product list.*



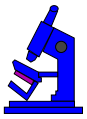
Topic: Assurance

How can we convince ourselves (or our employer) that the computer system is to be trusted?

Building assurance is best done by adopting formal methods to confirm, specify and verify the behaviour of systems.

ITSEC and CC

- UK, Germany, France, Netherlands produced Information Technology Security Evaluation Criteria (ITSEC).
- IT Security Evaluation Manual (ITSEM) specifies methodology for evaluation.
- Common Criteria for Information Technology Security Evaluation is ITSEC, CTCPEC (Canadian Criteria) and US Federal Criteria
- Accepted by the ISO (ISO15408).



ITSEC

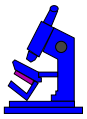
In an article, elements of the first certification of a smart-card system under the European ITSEC level 6 certification are outlined.

This process involved verification of the specification with independent systems, and a formal process for the implementation, deriving it from the specification using the *refinement* process.

Weekly mystery...

During the week, some students have been trying to find out....

The first student who can tell me the IP address of my HOME (that is the IP address allocated to the router in my living room by Starhub, my cable Internet provider) will win SING$2,000.00 OR a cup of coffee, whichever is the lesser.

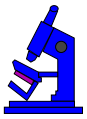


And the winner is?

No-one at 4:56 on Wednesday anyway...

Math preliminaries

The teacher pretended that algebra was a perfectly natural affair, to be taken for granted, whereas I didn't even know what numbers were. Mathematics classes became sheer terror and torture to me. I was so intimidated by my incomprehension that I did not dare to ask any questions. [Carl Jung]

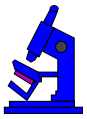


Exclusive-Or

- Exclusive-Or comes up constantly in cryptography.
- Same as *addition mod 2*

Exclusive-Or

- Also as **`xor`** or a plus sign in a circle, $\oplus$.
- The expression $a \oplus b$ means either a or b *but not both*.
- Ordinary *inclusive-or* in mathematics means either one or the other *or both*.
- The exclusive-or function in C / C++ / Java for bit strings as a hat character: **^**.

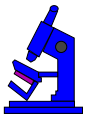


Exclusive-Or for 1-bit

Exclusive-Or		
a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive-Or for multiple bits

Message	A	B	C
m	0 1 0 0 0 0 0 1	0 1 0 0 0 0 1 0	0 1 0 0 0 0 1 1 . . .
Key= k	0 0 0 1 0 0 1 1	0 1 1 0 0 1 0 1	0 0 1 1 1 0 0 1 . . .
$K(m) = m \oplus k$	0 1 0 1 0 0 1 0	0 0 1 0 0 1 1 1	0 1 1 1 1 0 1 0 . . .
$K(m)$	R	'	z



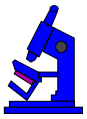
Exclusive-Or

$K(m)$	R	'	z
	0 1 0 1 0 0 1 0	0 0 1 0 0 1 1 1	0 1 1 1 1 0 1 0 . . .
Key= k	0 0 0 1 0 0 1 1	0 1 1 0 0 1 0 1	0 0 1 1 1 0 0 1 . . .
$m = K(m) \oplus k$	0 1 0 0 0 0 0 1	0 1 0 0 0 0 1 0	0 1 0 0 0 0 1 1 . . .
Message	A	B	C

If the bit-stream is random, and not known to an eavesdropper, then this is the most secure system. It is known as a one-time-pad.

Properties of XOR

$a \oplus a = 0$
$a \oplus 0 = a$
$a \oplus 1 = \sim a$, where $\sim$ is bit complement.
$a \oplus b = b \oplus a$ (commutativity)
$a \oplus (b \oplus c) = (a \oplus b) \oplus c$ (associativity)
$a \oplus a \oplus a = a$
if $a \oplus b = c$, then $c \oplus b = a$ and $c \oplus a = b$.



Reminder

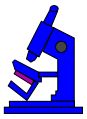
Exchange the values in two variables **a** and **b**

```
temp = a;
a = b;
b = temp;
```


Exchange using XOR

```
a = a xor b;
b = a xor b;
a = a xor b;
```

$$
\begin{aligned}
a' &= a \oplus b \\
b' &= (a \oplus b) \oplus b = a \\
a'' &= (a \oplus b) \oplus a = b
\end{aligned}
$$

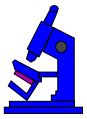


Logarithms

- $y = \log_b x$ is the same as $b^y = x$
- $b^{(\log_b x)} = x$
- Logarithm is inverse of exponential.

Logarithms

- Use logs base 2 in cryptography.
- $y = \log_2 x$ is the same as $2^y = x$
- $2^{10} = 1024$ is the same as $\log_2 1024 = 10$.
- $2^y > 0$ for all y, and
- $\log_2 x$ is not defined for $x \leq 0$.

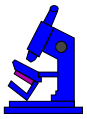



Properties of logs

$\log_2(ab) = \log_2 a + \log_2 b$, for all $a, b > 0$
$\log_2(a/b) = \log_2 a - \log_2 b$, for all $a, b > 0$
$\log_2(1/a) = \log_2(a^{-1}) = -\log_2 a$, for all $a > 0$
$\log_2(a^r) = r \log_2 a$, for all $a > 0, r$
$\log_2(a + b) =$ (Oops! No simple formula for this.)


Examples

Logarithms base 2	
$x = 2^y = 2^{\log_2 x}$	$y = \log_2 x$
$1,073,741,824$	30
$1,048,576$	20
$1,024$	10
8	3
4	2
2	1
1	0



Examples

Logarithms base 2	
$x = 2^y = 2^{\log_2 x}$	$y = \log_2 x$
1	0
$1/2$	-1
$1/4$	-2
$1/8$	-3
$1/1,024$	-10
0	$-\infty$
< 0	undefi ned

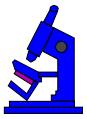
Natural logs

- A log base 2 is just a fixed constant times a natural log:

$$\log_2 x = \log_e x / \log_e 2, \text{ (mathematics)}$$

$$= \texttt{Math.log(x)/Math.log(2.0);} \text{ (Java).}$$

- The magic constant is:
  - $\log_e 2 = 0.69314\ 71805\ 59945\ 30941\ 72321$, or
  - $1/\log_e 2 = 1.44269\ 50408\ 88963\ 40735\ 99246.$



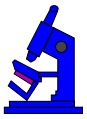
log_2= Bits to represent

Thus $\log_2 10000 = 13.28771238$, so it takes 14 bits to represent 10000 in binary. (In fact, $10000_{10} = 10011100010000_2$.) Exact powers of 2 are a special case: $\log_2 1024 = 10$, but it takes 11 bits to represent 1024 in binary, as 10000000000_2.

Similarly, $\log_{10}(x)$ gives the number of decimal digits needed to represent x.

Steps towards finite fields

- Groups, (skip over rings), fields, finite fields...
- Interested in closed algebraic systems/structures
- Compare +1 with +1 modulo 256



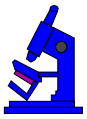
Groups

- A *group* is
  - a set of *group elements* with
  - a *binary operation* $\bullet$

If one denotes the group operation by $\bullet$, then the above says that for any group elements a and b, $a \bullet b$ is defined and is also a group element (i.e. it is closed)

Groups

- Groups
  - are *associative*, meaning that $a \bullet (b \bullet c) = (a \bullet b) \bullet c$
  - have an *identity element* e satisfying $a \bullet e = e \bullet a = a$ for any group element a.
  - have an *inverse* a' any element a satisfying $a \bullet a' = a' \bullet a = e$.



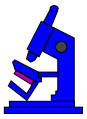
Groups

- If $a \bullet b = b \bullet a$ for all group elements a and b, the group is *commutative*[1].
- Otherwise it is *non-commutative*. Notice that even in a non-commutative group, $a \bullet b = b \bullet a$ might sometimes be true — for example if a or b is the identity.
- A group with only finitely many elements is called *finite*; otherwise it is *infinite*.

[1] or *abelian*.

Examples

- The *integers* (all whole numbers, including 0 and negative numbers) form a group using addition. The identity is 0 and the inverse of a is $-a$.
  - This is an infinite commutative group.
- The *positive rationals* (all positive fractions, including all positive integers) form a group if ordinary multiplication is the operation. The identity is 1 and the inverse of r is $1/r = r^{-1}$.
  - This is another infinite commutative group.

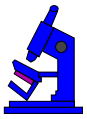


Examples

- The *integers mod n* form a group for any integer $n > 0$. This group is often denoted Z_n. Here the elements are 0, 1, 2, $\ldots$, $n-1$ and the operation is addition followed by remainder on division by n. The identity is 0 and the inverse of a is $n-a$ (except for 0 which is its own inverse).
  - This is a finite commutative group.

Integers modulo 10

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	0
2	2	3	4	5	6	7	8	9	0	1
3	3	4	5	6	7	8	9	0	1	2
4	4	5	6	7	8	9	0	1	2	3
5	5	6	7	8	9	0	1	2	3	4
6	6	7	8	9	0	1	2	3	4	5
7	7	8	9	0	1	2	3	4	5	6
8	8	9	0	1	2	3	4	5	6	7
9	9	0	1	2	3	4	5	6	7	8



Fields

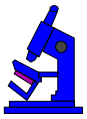
- A field has two operations traditionally called $+$, $*$
  - $+$, with elements of the field forming a commutative group. Identity is 0 and inverse of a is $-a$.
  - $*$, with elements of the field except 0 forming another commutative group, identity denoted by 1 and inverse of a denoted by a^{-1}.

Fields

- There is also the *distributive identity*, linking $+$ and $*$:

$$a * (b + c) = (a * b) + (a * c)$$

- Exclude *divisors of zero*, that is, non-zero elements whose product is zero.
- Equivalent to the following cancellation property: if c is not zero and $a * c = b * c$, then $a = b$.

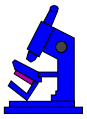


Examples

- The *rational numbers* (fractions) Q, or the *real numbers* $\mathbb{R}$, or the *complex numbers* $\mathbb{C}$, using ordinary addition and multiplication (extended in the last case to the complex numbers).
  - These are all infinite fields.

Finite field: integers mod p

- The *integers mod p*, denoted Z_p, where p is a prime number (2, 3, 5, 7, 11, 13, 17, 19, 23, 29, ...).
  - A group using $+$.
  - Elements without 0 form a group under $*$.
  - The identity is clearly 1, but
  - the inverse of a non-zero element a is not obvious.

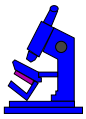


Modular arithmetic: $+$ in Z_7

$+$	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

Modular arithmetic: $*$ in Z_7

$*$	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

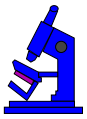


Modular arithmetic: $+, *$ inverses

a	$-a$	a^{-1}
0	0	-
1	6	1
2	5	4
3	4	5
4	3	2
5	2	3
6	1	6

Integers mod p inverse

- Inverse must be x satisfying $(x * a) \bmod p = 1$.
- Find x using the *extended Euclidean algorithm*:
  - p is prime and a is non-zero, the greatest common divisor of p and a is 1.
  - The extended Euclidean algorithm gives x and y satisfying $x * a + y * p = 1$, or $x * a = 1 - y * p$,
  - and x is the inverse of a.



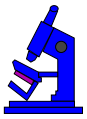
Properties of finite fields

1. Fields are unique up to renaming of their elements, meaning that one can always use a different set of symbols to represent the elements of the field, but it will still be essentially the same.

2. Their size must be p^n where p is a prime and n a positive integer.

Modular arithmetic: $+$ in Z_8

$+$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

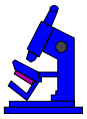


Modular arithmetic: $*$ in Z_8

$*$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

Modular arithmetic: $+, *$ inverses

a	$-a$	a^{-1}
0	0	-
1	7	1
2	6	-
3	5	3
4	4	-
5	3	5
6	2	-
7	1	7



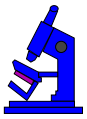
Modular arithmetic in Z_8

- Note that in Z_8, modular arithmetic using ordinary addition and multiplication does not form a field.
- Why not?

  Well - because there are not multiplicative inverses....

But ... but ...

- $8 = 2^3$, so we can have a field of size 8 can't we?
- Yes - but the operations are not *simple* arithmetic $+$ and $*$.
- Instead we use *polynomial* arithmetic (to be explained...)

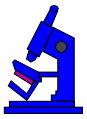


Another field: $GF(2^n)$

- A finite field with p^n elements for any integer $n > 1$, denoted $GF(p^n)$.
- Useful in cryptography with $p = 2$, that is, with 2^n elements for $n > 1$.
- The case $2^8 = 256$ is used, for example, in the new U.S. Advanced Encryption Standard (AES).
- Arithmetic different...

Polynomial arithmetic: $3x^5 + x^2 + x + 1$.

- ✔ A polynomial of degree n is $\sum_{i=0}^{n} a_i x^i$.
- ✔ Consider the set where the coefficients a_i belong to Z_p.
- ✔ Where $p = 2$, the coefficients are either 0 or 1.
- ✔ In this case, polynomial addition is **`xor`** over the coefficients.



Polynomial + example

Consider the addition of 3 and 6:

	$3 = 011$	$x + 1$
$\oplus$	$6 = 110$	$x^2 + x$
	$5 = 101$	$x^2 + 1$

Coefficients may be only 0 or 1, and we do the arithmetic modulo 2.

For multiplication, if it results in too large a polynomial, then we reduce it, by dividing it by an irreducible polynomial of degree n, and take the residue.

Multiplication of 7 and 7:

	$7 =$	111	$x^2 + x + 1$
$*$	$7 =$	111	$*\ x^2 + x + 1$
		111	$x^2 + x + 1$
	$\oplus$	1110	$x^3 + x^2 + x$
	$\oplus$	11100	$x^4 + x^3 + x^2$
		10101	$x^4 + x^2 + 1$

The multiplication has resulted in a polynomial that is too large, so we reduce it

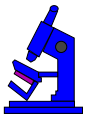
Reduction

$$
\begin{array}{r|l}
 & 10 \\
\hline
1011 & 10101 \\
 & \underline{1011} \\
 & 11
\end{array}
\qquad\qquad
\begin{array}{r|l}
 & x \\
\hline
x^3+x+1 & x^4 + x^2 + 1 \\
 & \underline{x^4 + x^2 + x} \\
 & x + 1
\end{array}
$$

Finally, we have that in $\mathrm{GF}(2^3)$, $7 * 7 = 3$.

Addition in : $GF(2^3)$

$+$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	2	3	4	5	6	7
1	1	0	3	2	5	4	7	6
2	2	3	0	1	6	7	4	5
3	3	2	1	0	7	6	5	4
4	4	5	6	7	0	1	2	3
5	5	4	7	6	1	0	3	2
6	6	7	4	5	2	3	0	1
7	7	6	5	4	3	2	1	0



Multiplication in $GF(2^3)$

$*$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	3	1	7	5
3	0	3	6	5	7	4	1	2
4	0	4	3	7	6	2	5	1
5	0	5	1	4	2	7	3	6
6	0	6	7	1	5	3	2	4
7	0	7	5	2	1	6	4	3

Inverses in $GF(2^3)$

a	$-a$	a^{-1}
0	0	-
1	1	1
2	2	5
3	3	6
4	4	7
5	5	2
6	6	3
7	7	4