Resilient Networking

Disclaimer: this lecture has been created with very valuable input from Jussi Kangasharju

Module 2 – Background on Graphs (Winter Term 2020)
Thorsten Strufe
Module Outline

- **Background 1: Graph Analysis**
  - Why bother with theory?
  - Graphs and their representations
  - Important graph metrics
  - On robustness and resilience

- **Background 2: Crypto**
  - Stream ciphers and the OTP
  - Block ciphers and their operation modes
  - Key agreement
  - Asymmetric Crypto
  - Integrity
Some questions...

- How robust is the Internet?
- Why do darknets work?
- What do the existing networks actually look like, and why?
- What would an ideal computer network look like?

Gnutella snapshot, 2000
Graphs

▪ Graph families and models
  ▪ Random graphs
  ▪ Small world graphs
  ▪ Scale-free graphs

▪ Graph theory and real computer networks
  ▪ How are the graph properties reflected in real systems?
    ▪ Users/nodes are represented by vertices in the graph
    ▪ Edges represent connections in overlay / routing table entries

▪ Concept of self-organization (how/why do they evolve?)
  ▪ Network structures emerge from simple rules
  ▪ E.g. also in social networks, www, actors playing together in movies
What is a Graph?

- **Definition of a graph:**

  Graph $G = (V, E)$ consists of two finite sets, set $V$ of vertices (nodes) and set $E$ of edges (arcs, links) for which the following apply:

  1. If $e \in E$, then exists $(v, u) \in V \times V$, such that $v \in e$ and $u \in e$
  2. If $e \in E$ and above $(v, u)$ exists, and further for $(x, y) \in V \times V$ applies $x \in e$ and $y \in e$, then $\{v, u\} = \{x, y\}$

---

**Example graph with 4 vertices and 5 edges**

---

**Side note:**

Edges can have (multiple) “weights” $w : E \rightarrow \mathbb{R}$
Properties of Graphs

- An edge $e \in E$ is directed if the start and end vertices in condition 2 above are identical: $v = x$ and $y = u$

- An edge $e \in E$ is undirected if $v = x$ and $y = u$ as well as $v = y$ and $u = x$ are possible

- A graph $G$ is directed (undirected) if the above property holds for all edges

- Graph $G_1 = (V_1, E_1)$ is a subgraph of $G = (V, E)$, if $V_1 \subseteq V$ and $E_1 \subseteq E$ (such that conditions 1 and 2 are met)
How are Graphs Implemented?

- Adjacency/Incidence Matrix

- Adjacency/Incidence List
  - (Plus specialized others..)

```
1  0  1  0
2  1  0  1
3  0  1  0
```

```
(1,2)  
(2,1),(2,3) 
(3,2)  
1:2    
2:1,3 
3:2    
```

VERY good book is: Sedgewick: Algorithms in C, part 3 (Graph Algorithms)
Some Examples of Computer Networks

- Early Computer Networks Aloha (or WSN, for that matters)
- Network Layers 1,2
Examples: The Internet

- Globally internetworked computers (Layer 3)
Examples: Overlays (Layer 7 (?))

- A **CLIQUE** is a graph that is fully connected \((u,v) \in E\) for all \(u \in V\) and \(v \in V, u \neq v\)

- A (P2P) Overlay \((V_o,E_o)\) (in general) is a subgraph such that \(V_o=V\) and \(E_o \subseteq E\) (edges are selected edges from a CLIQUE graph)

- **Why?** Considering the nodes to be on the Internet, they all can create connections between each other...
Important Graph Metrics

- **Order**: the number of vertices in a graph: $|V|$
- **Size** of the graph is the number of edges $|E|$

- **Distance**: $d(v, u)$ between vertices $v$ and $u$ is the length of the shortest path between $v$ and $u$

- **Diameter**: $d(G)$ of graph $G$ is the maximum of $d(v, u)$ for all $v, u \in V$

- The **density** of a graph is the ratio of the number of edges and the number of possible edges.
Graph Metrics: Vertex Degree

- In graph $G = (V, E)$, the **degree** of vertex $v \in V$ is the total number of edges $(v, u) \in E$ and $(u, v) \in E$
  - Degree is the number of edges incident to a vertex

- For directed graphs, we distinguish between **in-degree** and **out-degree**
  - In-degree is number of edges with the vertex as end-point
  - Out-degree is number of edges going with the vertex as starting point

- The degree of a vertex can be obtained as:
  - Sum of the elements in its row in the incidence matrix
  - Length of its vertex incidence list

- The **degree distribution** is the distribution over all node degrees
  (given as a frequency distribution or (often) complementary cumulative distribution function CCDF (Komplement der Verteilungsfunktion))
Graph Metrics: Degree Distribution (Examples)

Human Protein Interaction [biomedcentral.com]
The Internet (AS-level) [pacm.princeton.edu]
Online Social Network (xing crawl) [strufe10popularity]

Resilient Networks – Winter Term 2020 (KIT/TUD)
Routing and Graph Metrics on Path Length

- **Routing**: Define strategy to find path from $s$ to $d$
  commonly local strategy, based on address/distances,
  usually “greedy”

- **All pairs shortest paths (APSP)**: $d(v, u)$ | all $v,u \in V$

- **Hop Plot**: Distance distribution over all distances $Hist(\text{APSP}(G))$

- **Average CHARACTERISTIC path length (CPL)**: Sum of the distances over all pairs of nodes divided by the number of pairs

- **For defined routings (usually greedy) on directed graphs**: Characteristic Routing Length (CRL): average length of paths found (potentially stochastic...)
Important Graph Metrics: Connectivity

- **Edge connectivity**: is the minimum number of edges that have to be removed to separate the graph into at least two components.

- **Vertex connectivity**: the minimum number of nodes.

- How can we calculate them?
- Which of both is higher?
- In which cases are they the same?
- So where do you attack, naively? ;-) 

- Homework: check maxflow, Menger‘s Theorem
Graph Metrics: Network Clustering

- **Clustering coefficient**: number of edges between neighbors divided by maximum number of edges between them
  - $k$ neighbors: $k(k-1)/2$ possible edges between them

$$C(i) = \frac{2E(N(i))}{d(i)(d(i)-1)}$$

$E(N(i)) =$ number of edges between neighbors of $i$

$d(i) =$ degree of $i$

- What if: a node has only one neighbor? 😊

- Variations exist: *local, average, global CC*

Classes of Graphs

- Regular graphs
- Random graphs
- Graphs with Small-World characteristic
- Scale-free graphs

- ...Graphs with plenty more characteristics
  - (dis-) assortativity
  - Rich-club connectivity
  - ...
Regular Graphs

- Regular graphs have traditionally been used to model networks, they have
  - constant node degree (discrete degree distribution of a single value)
  - potentially different topologies
- However, the model does not reflect real nets well
Random Graphs

- Random graphs are first widely studied graph family
  - Many overlay networks choose neighbors more or less randomly

- Two different generators generally used:
  - Erdös and Renyi
  - Gilbert

- Gilbert’s definition: Graph $G_{n,p}$ (with $n$ nodes) is a graph where the probability of an edge $e = (v, w)$ is $p$

**Construction algorithm:**
- For each possible edge, draw a random number in (0,1)
- If the number is smaller than $p$, then the edge exists
- ($p$ can be function of $n$ or constant)
Basic Properties of Random Graphs

Giant Connected Component

Let \( c > 0 \) be a constant and \( p = c/n \).

If \( c < 1 \) every component of \( G_{n,p} \) has order \( O(\log N) \) with high probability.

If \( c > 1 \) then there is one component of order \( n^*(f(c) + O(1)) \) where \( f(c) > 0 \), with high probability. All other components have order \( O(\log N) \).

- **English**: Giant connected component emerges with high probability when average degree is about 1

Node degree distribution

- If we take a random node, how high is the probability \( P(k) \) that it has degree \( k \)?
- Node degree is Poisson distributed
  - Parameter \( c = \) expected number of occurrences

\[
P(k, c) = \frac{c^k e^{-c}}{k!}
\]

Clustering coefficient

- Clustering coefficient of a random graph is asymptotically equal to \( p \) with high probability
Utility of Random Graphs

- Random Graphs are useful
  - Easy to analyze
  - Good approximation of reality with regards to some properties

- Random Graphs are wrong models of reality
  - Many properties of real-world networks diverge considerably from this random case
Milgram's Small World Experiment
Six Degrees of Separation

- Famous experiment from 1960’s (S. Milgram)

- Send a letter to random people in Kansas and Nebraska and ask people to forward letter to a person in Boston
  - Person identified by name, profession, and city

- **Rule**: Give letter only to people you know by first name and ask them to pass it on according to same rule
  - Note: Some letters reached their goal

- Letter needed six steps on average to reach the destination

- Graph theoretically: Social networks have dense local structure, but (apparently) small diameter
  - Generally referred to as “small world effect”
  - Usually, small number of persons act as “hubs”
Small-World Network Model

- Developed/discovered by Watts and Strogatz (1998)
  - Over 30 years after Milgram’s experiment!

- Watts and Strogatz looked at three networks
  - Film collaboration between actors, US power grid, Neural network of worm *Caenorhabditis elegans* (“C. elegans”)

- Measured characteristics:
  - Clustering coefficient as a measure for ‘regularity’, or ‘locality’ of the network
    - If it is high, short edges exist with high probability
  - The average path length between vertices

- Results:
  - Grid-like networks:
    - High clustering coefficient $\Rightarrow$ high average path length
      (edges are not ‘random’, but rather ‘local’)
  - Most real-world (natural) networks have a high clustering coefficient
    (0.3-0.4), but nevertheless a low average path length
Small-World Graph Generator (W&S)

- Put all $n$ nodes on a ring, number them consecutively from 1 to $n$
- Connect each node with its $k$ clockwise neighbors
- Traverse ring in clockwise order

- For every edge
  - Draw random number $r$
  - If $r < p$, then re-wire edge by selecting a random target node from the set of all nodes (no duplicates)
  - Otherwise keep old edge

- Different values of $p$ give different graphs
  - If $p$ is close to 0, then original structure mostly preserved
  - If $p$ is close to 1, then new graph is random
  - Interesting things happen when $p$ is somewhere in-between
Regular, Small-World, Random

Regular

Small-World

Random

\[ p = 0 \]

\[ p = 1 \]
Utility of Small World Property

- Small world property
  - Explains why short paths exist

- Does not explain, how and why they are found?
Kleinberg’s Small-World Navigability Model

- Small-world model explains why short paths exist

- Missing piece in the puzzle: why can we find these paths?
  - Each node has only local information
  - Even if a shortcut exists, how do people know about it?
  - Milgram’s experiment:
    - Some additional information (profession, address, hobbies etc.) is used to decide which neighbor is “closest” to recipient
    - Results showed that first steps were the largest

- Kleinberg’s Small-World Model
  - Set of points in an \( n \times n \) grid
  - Distance is the number of “steps” separating points
    - \( d(i, j) = |x_i - x_j| + |y_i - y_j| \)
Kleinberg’s Topologies

- Take \( d \)-dimensional grid in which all nodes are connected to all neighbors along each axis
- Additionally connect nodes in higher distance with probability decreasing with distance

iow: the probability that node \( j \) is selected as neighbor for \( i \) is proportional to \( d(i, j)^{-r} \), with clustering exponent \( r \)
Intuition of Navigation in Kleinberg’s Model

- **Simple greedy routing**: nodes only know local links and target position, always use the link that brings message closest to target
  - If $r=2$, expected lookup time is $O(\log^2 n)$
  - If $r\neq 2$, expected lookup time is $O(n^\varepsilon)$, where $\varepsilon$ depends on $r$

- Kleinberg has shown: Number of messages needed is proportional to $O(\log^2 n)$ iff $r=s$ ($s =$ number of dimensions)
  - Idea behind proof: for any $r > s$ there are too few long edges to make paths short
  - For $r < s$ there are too many random edges $\Rightarrow$ too many choices for passing message, greedy may not deterministically converge to destination
    - The message will make a (long) random walk through the network
Problems with Small-World Graphs

Small-world graphs explain why:
- Highly clustered graphs can have short average path lengths ("short cuts")

Small-world graphs do NOT explain why:
- This property emerges in real networks
  - Real networks are practically never ring-like

Further problem with small-world graphs:
- Nearly all nodes have same degree
- Not true for random graphs
- What about real networks?
Real World Measurements: World Wide Web

- Links between documents in the World Wide Web
  - 800 Mio. documents investigated (S. Lawrence, 1999)

- What was expected so far?
  - Number of links per web page: $\langle k \rangle \sim 6$
  - Number of pages in the WWW: $N_{WWW} \sim 10^9$

- Probability “page has 500 links”: $P(k=500) \sim 10^{-99}$
- Number of pages with 500 links: $N(k=500) \sim 10^{-90}$
WWW: result of investigation

\[ P(out(k)) \sim k^{-\gamma_{out}} \]

\[ \gamma_{out} = 2.45 \]

\[ P(in(k)) \sim k^{-\gamma_{in}} \]

\[ \gamma_{in} = 2.1 \]

\[ P(k=500) \sim 10^{-6} \]

\[ N_{WWW} \sim 10^9 \]

\[ \Rightarrow N(k=500) \sim 10^3 \]
Real-World Measurements: The Internet

- Faloutsos et al. study from 99: Internet topology examined in 1998
  - AS-level topology, during 1998 Internet grew by 45%

- Motivation:
  - What does the Internet look like?
  - Are there any topological properties that don’t change over time?
  - How to generate Internet-like graphs for simulations?

- 4 key properties found, each follows a power-law;
- Sort nodes according to their (out)degree
  1. **Out degree** of a node is proportional to its rank to the power of a constant
  2. Number of nodes with same out degree is proportional to the out degree to the power of a constant
  3. Eigenvalues of a graph are proportional to the order to the power of a constant
  4. Total number of pairs of nodes within a distance d is proportional to d to the power of a constant
Conclusion: Power Law Networks

- “Power Law” relationships
  - For the Internet...
  - For Web pages
    - The probability \( P(k) \) that a page has \( k \) links (or \( k \) other pages link to this page) is proportional to the number of links \( k \) to the power of \( y \)

- General ”Power Law” Relationships
  - A certain property \( k \) is – independent of the growth of the system – always proportional to \( k^{-a} \), where \( a \) is a constant (often \( 2 < a < 4 \))

- Power laws very common (”natural”)
  - power law networks exhibit small-world-effect (always?)
  - E.g. WWW: 19 degrees of separation
    - (R. Albert et al., Nature (99); S. Lawrence et al., Nature (99))

- Also termed: scale-free networks
Barabasi-Albert-Model

How do power law networks emerge?

- In a network where new vertices (nodes) are added and new nodes tend to connect to well-connected nodes, the vertex connectivities follow a power-law

Barabasi-Albert-Model: power-law network is constructed with two rules
1. Network grows in time
2. New node has preferences to whom it wants to connect

Preferential connectivity modeled as
- Each new node wants to connect to \( m \) other nodes
- Probability that an existing node \( j \) gets one of the \( m \) connections is proportional to its degree \( d(j) \)

New nodes tend to connect to well-connected nodes

Another way of saying this: “the rich get richer”
Resilience of Scale-Free Networks

- Random failures vs. directed attacks
Resilience of Scale Free Networks

- **Experiment**: take network of 10000 nodes (random and power-law) and remove nodes randomly

- **Random graph**:
  - Take out 5% of nodes: Biggest component 9000 nodes
  - Take out 18% of nodes: No biggest component, all components between 1 and 100 nodes
  - Take out 45% of nodes: Only groups of 1 or 2 survive

- **Power-law graph**:
  - Take out 5% of nodes: Only isolated nodes break off
  - Take out 18% of nodes: Biggest component 8000 nodes
  - Take out 45% of nodes: Large cluster persists, fragments small

- Networks with power law exponent < 3 are very robust against random node failures
  - ONLY true for random failures!
The consequence...

\[ L: \text{"Average Connected Distance"} \]
Summary of Graph Analyses...

- The network structure of a network influences:
  - average necessary number of hops (path length)
  - possibility of greedy, decentralized routing algorithms
  - stability against random failures
  - sensitivity against attacks
  - redundancy of routing table entries (edges)
  - many other properties of the system build onto this network

- Important measures of a network structure are:
  - average path length
  - the degree distribution
  - clustering coefficient
  - Various resilience metrics (with differing foci)
Questions?
Module Outline

- Some background

- Symmetric crypto:
  - Stream ciphers and the OTP
  - Block ciphers and their operation modes

- Key agreement

- Asymmetric Crypto

- Integrity
Terminology: Cryptology (Kryptologie)

- **Cryptology**:  
  - Science concerned with communications in secure and usually secret form  
  - Derived from the Greek  
    - *kryptós* (hidden) and  
    - *lógos* (word)

- Cryptology encompasses:
  - **Cryptography** (*gráphein* = to write): principles and techniques by which information can be concealed in ciphertext and later revealed by legitimate users employing a secret key
  
  - **Cryptanalysis** (*analýein* = to loosen, to untie): recovering information from ciphers without knowledge of the key
Terminology: Cipher (Chiffren)

- **Cipher (Chiffren):**
  - Method of transforming a message (plaintext) to conceal its meaning (and to transform it back)

- Ciphers are one class of cryptographic algorithms (E,D)
- The transformation usually takes the message and a *(secret) key* as input

- Unfortunately: sometimes also used as synonym for the concealed *ciphertext (Chiffrat)!*
Achieving the security goals

- Recall CIA:
  - **Confidentiality**: only authorized access to information
  - **Integrity**: detection of message modification
  - **Availability**: services are live and work correctly

- Where crypto can (trivially) help:
  - **Confidentiality**: Encryption transforms plaintext to conceal it
  - **Integrity**: Append signature that proves legit sender’s knowledge

- Immediate:
  -Hide content or properties (*content, parties, parameters*)
  -Prove a claim (*message/entity authentication, commitments, ZKP*)

- Secondary:
  -Generate (shared) randomness
  - (Enforce) collaboration (*key agreement, secret sharing, threshold crypto*)
The cipher method must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience.

- i.o.w: E, and D will inevitably be discovered at some stage
  - All algorithms should be public
  - security must rely on secrecy of the key only
Confidential Communication

Spaces
- $\mathcal{M}$: plaintext space (e.g. words over an alphabet)
- $\mathcal{C}$: space of ciphertexts
- $\mathcal{K}$: space of keys

Algorithms of a private-key (symmetric) encryption scheme
- $KGen$: generates some (usually random) key $k$
- $E$: encrypts a plaintext $m$ using key $k$ and outputs the ciphertext $c$
- $D$: decrypts a ciphertext $c$ using key $k$ and outputs the plaintext $m$

Correctness for all $k \in \mathcal{K}$, $m \in \mathcal{M}$:
$$\text{Dec}(k, \text{Enc}(k, m)) = m$$
Classifying Encryption Algorithms

- Type of operation
  - *Substitution*: substitute letters by other letters (or symbols)
  - *Transposition*: permute letters according to some scheme

- Number of keys
  - *Symmetric*: secret key
  - *Asymmetric*: „public key“, pair of public and private key

- Processing of plaintext
  - *Stream ciphers*: operate on streams of bits
  - *Block ciphers*: operate on b-bit blocks
Constructing a Stream Cipher

- Idealized OTP:
  - Premise:
    - PRNG is a function $G: \{0,1\}^s \rightarrow \{0,1\}^n$, $n >> s$
    - Deterministic algorithm from seed space to key space (looking random)

- Idea:
  - OTP: $E(k,m): c = m \oplus k$  $D(k,c): m = c \oplus k$ with random $k$
  - In reality: stream of key bits from PRNG (seeded with „$k$“)
Security Definitions

- **Semantic Security**
  - An „efficient“ algorithm cannot find any information in the CT (the CT is polynomially indistinguishable from a CT with PS)

- **Provable Security**
  - Reduction of construction to some mathematical problem which is *assumed* to be hard (then so is breaking the construction)

- **Perfect Secrecy**
  - The ciphertext does not reveal *any* information about the PT
  - Caveat: *Key must be random and as long as the message*
Information Theoretic Security

- Shannon (1949): „CT should not reveal any information about PT“

- Def: A cipher \((E,D)\) over \((\mathcal{K}, \mathcal{M}, \mathcal{C})\) has **perfect secrecy** if

  - \(\forall m_0, m_1 \in \mathcal{M}\) (with \(\text{len}(m_0) = \text{len}(m_1)\))
  - \(\forall c \in \mathcal{C}\) and \(k \leftarrow \mathcal{K}: \mathcal{R}\)

    \[ \Pr[E(k,m_0) = c] = \Pr[E(k,m_1) = c] \]

- *So being an attacker, what do I learn?*
- No CT attack can tell if msg is \(m_0, m_1\) (or any other message)
- \(\rightarrow\) No CT only attacks
Semantic Security

- For $b=0,1$ define experiments $\text{EXP}(0)$ and $\text{EXP}(1)$ as:

  $$\text{Adv}_{\text{SS}}[A,E] := | \Pr[\text{EXP}(1) = 1] - \Pr[\text{EXP}(0) = 1] | \in [0,1]$$

  (i.o.w.: $| \Pr[ b' = b ] - \Pr[ b' \neq b ] |$)

- $E$ is called **semantically secure** if for all eff. $A$, $\text{Adv}_{\text{SS}}[A,E]$ is negligible.

- We define „efficient“ to be PPT, we also talk of „PPT adversaries“
Towards Block Ciphers: Properties of Functions

- A little refresher on functions...

\[ f: X \rightarrow Y \]
\[ X = \{a, b, c\} \quad Y = \{1, 2, 3, 4\} \quad y = f(x) \quad Im(f) = \{1, 2, 4\} \]
Functions, Functions, Functions

- $X = \{1,2,3,\ldots,10\}$  
  $f(x) = x^2 \mod 11$
- $f: X \rightarrow Y$  
  $Y = \{1,3,4,5,9\}$
- $f$ is called "onto" (surjective): $Y = \text{Im}(f)$ or:  
  $\forall y \in Y \exists x \in X: y = f(x)$
- "one-to-one" (injective): $\forall x_1, x_2 \in X: f(x_1) = f(x_2) \Rightarrow x_1 = x_2$
- bijection: $f(x)$ is $1-1$ and $\text{Im}(f) = Y$

For bijection $f$ there is an inverse: $g = f^{-1}: g(y) = x$ ($= f(g(x))$)
Hint: (Trapdoor) One-way Functions

- Finding the inverse $f^{-1}$ is not always „easy“

- **One way functions:**
  - A function $f: X \rightarrow Y$ is called a
  - one-way-function, if $f(x)$ is „easy“ to compute
  - for all $x \in X$, but for “essentially all” elements
  - $y \in \text{Im}(f)$ it is computationally infeasible to find the preimage $x$.

- **Trapdoor one-way functions:**
  - A trapdoor one-way function is a one-way function that, given some additional trapdoor information, is feasible to invert.
(Pseudo Random) Permutations, Involutions

- **Permutations and Involutions:**
  - A permutation $\pi$ is a bijective function from a domain to itself:
  $\pi: X \rightarrow X \quad \text{Im}(f) = X$
  - A permutation $\pi$ with: $\pi = \pi^{-1}$ (or: $\pi(\pi(x)) = x$)
  - is called an *involution*.

- **Pseudo Random Functions (PRF):**
  - $F: K \times X \rightarrow Y$
  - on „domain“ $X$ and „range“ $K$, with „efficient“ algorithm to evaluate $F(k,x)$

- **Pseudo Random Permutation (PRP):**
  - Permutation $E: K \times X \rightarrow X$
  - has efficient deterministic algorithm to evaluate $E(k,x)$ and
  - efficient inversion algorithm $D(k,x) = E^{-1}$
Stream Ciphers and Block Ciphers

- **Goal:**
  - Build a secure PRP for b-bit blocks

- **Examples:**
  - 3DES: \( n = 64, k = 168 \)
  - AES: \( n = 128, k = 128,192,256 \)
The Advanced Encryption Standard

- **1997:** NIST publishes request for proposal
- **1998:** 15 submissions
- **1999:** NIST chooses 5 finalists
- **2000:** NIST chooses Rijndael as AES
- **Key sizes:** 128, 192, 256 bits  
  **Block size:** 128 bits
- **Best known (theoretical) attacks in time** \( \approx 2^{99} \)
AES Substitution-Permutation Network

Not a Feistel network:

\[ \text{input} \oplus S_1 S_2 S_3 \cdots S_{16} \oplus k_1 \oplus S_1 S_2 S_3 \cdots S_{16} \oplus k_2 \oplus \cdots \oplus k_n \oplus \text{output} \]
AES-128 scheme

4x4 input

10 rounds

k_0, k_1, k_2, ..., k_{10}

key

16 bytes

key expansion: 16 bytes → 176 bytes

4x4 output

invertible
Building Block Ciphers (Modes of Operation)

- So far we have seen PRFs and PRPs (3DES, AES)

- **Goal:**
  - Use secure PRPs
  - Build „secure“ encryption of arbitrarily long message
Electronic Code Book Mode (*insecure!*)

- Encrypt each block with the keyed PRP:

```
<table>
<thead>
<tr>
<th>m[0]</th>
<th>m[1]</th>
<th>...</th>
<th>m[L]</th>
</tr>
</thead>
<tbody>
<tr>
<td>F(k)</td>
<td>F(k)</td>
<td>...</td>
<td>F(k)</td>
</tr>
<tr>
<td>c[0]</td>
<td>c[1]</td>
<td>...</td>
<td>c[L]</td>
</tr>
</tbody>
</table>
```

- ECB encryption is *deterministic*
- ⇒ identical PT is encrypted to identical CT:

- *Is this “secure” (how)?*
Randomized Counter Mode R-CTR

- Let $F: K \times \{0,1\}^n \rightarrow \{0,1\}^n$ be a secure PRF.
- $E(k,m)$: choose a random $IV \in \{0,1\}^n$ and do:

  
<table>
<thead>
<tr>
<th>IV</th>
<th>m[0]</th>
<th>m[1]</th>
<th>...</th>
<th>m[L]</th>
</tr>
</thead>
<tbody>
<tr>
<td>F(k,IV)</td>
<td>F(k,IV+1)</td>
<td>...</td>
<td>F(k,IV+L)</td>
<td></td>
</tr>
</tbody>
</table>

  
<table>
<thead>
<tr>
<th>IV</th>
<th>c[0]</th>
<th>c[1]</th>
<th>...</th>
<th>c[L]</th>
</tr>
</thead>
</table>

  
  ciphertext

- Variation: Choose 128 bit IV as: nonce $||$ counter, to avoid repetition

- Remarks:
  - $E$, $D$ can be parallelized and $F(k,IV+i)$ can be precomputed
  - R-CTR allows random access, any block can be decrypted on its own
  - Again: $F$ can be any PRF, no need to invert
Intermediate Summary

- You know the different classes of encryption algorithms
- You understand Kerckhoff’s principle
- You’ve been introduced to the idea of the OTP and stream ciphers
- You know semantic security and the difference to perfect secrecy
- You recall properties of functions
- You can explain what (Trapdoor) One-way functions are
- You can explain AES
- You saw different modes of operation and know their properties
Key Agreement (/ Authentication)

- „Key Distribution“:
  - Secure Channel

How many keys have to be exchanged in a system with N participants?
Interlude: The Dolev - Yao Adversary Model

- Mallory has full control over the communication channel
  - Intercept/eavesdrop on messages (passive)
  - Relay messages
  - Suppress message delivery
  - Replay messages
  - Manipulate messages
  - Exchange messages
  - Forge messages

- But:
  - Mallory *can’t* break (secure) cryptographic primitives!

E.g. reverse a secure hash, find collisions, break AES...
Attacks on Key Agreement

• Man-in-the-middle attack

• Replay attack
KDC: a Very Simple Protocol

- Simple Key Exchange:
  - TTP knows / generates all keys
  - Eve won’t break encryption, but Mallory may actively interfere
  - ...

\[ K_a, K_b \]

\[ \{K\}_{K_b}, \{K\}_{K_a} \]

\[ \{K\}_{K_b} \]
Impersonation / MitM – Step 1
Impersonation / MitM – Step 2

\[ B, D, \{K'_d, K'_b\} \]

\[ ^\text{„A“}, \{K’\}_b \]
Impersonation / MitM - Result

- Hence: Prevent MitM/replay -> authenticated key exchange
- -> Authenticate both parties (*requires trust in KDC*)
- -> ensure „freshness“ of messages
- (and exchange a key...)
Schroeder-Needham Key Exchange

- Impersonation/MitM prevented by explicit addressing (Alice and Bob)
- Replay prevented by Nonce
- If Malory has broken old key, she can impersonate Alice (prevented by timestamps for „freshness“)
Key Agreement

- Don’t exchange keys, calculate them!

- **Goal:**
  - Exchanged information should be public

- Initial idea (due to Merkle, ’74):
  - Alice creates $2^{32}$ puzzles (containing index $P_i$ and key) (O(n))
  - Bob selects random puzzle, „calculates“ index $P_j$ and key (O(n))
  - Bob informs Alice of $P_j$, both know key.

- What is the complexity for Mallory? $O(n^2)$

Can we do better? Polynomial advantage?

Ralph Merkle, Martin Hellman, Whitfield Diffie
Let's generate keys: Diffie-Hellman

- Consider $\mathbb{Z}_p^*$ generated by $g$, and $\varphi(p) = p-1$

- Alice chooses $a \leftarrow \{1,...,(p-1)\}$, Bob chooses $b \leftarrow \{1,...,(p-1)\}$

- Alice can calculate: $(g^b)^a = g^{ab}$ = Bob calculates: $(g^a)^b$

- Computational Diffie Hellmann Problem (CDH (ECDH)):
  - Given $p$, $g$, $g^a$, $g^b$
  - Output $g^{ab}$

Assume absence of MitM
Public key encryption

A public-key encryption system is a triple of algorithms $(G, E, D)$:

- $G()$: randomized alg. outputs a key pair $(pk, sk)$
- $E(pk, m)$: randomized alg. that takes $m \in M$ and outputs $c \in C$
- $D(sk, c)$: det. alg. that takes $c \in C$ and outputs $m \in M$ or ⊥

**Correctness:** $\forall (pk, sk) \text{ output by } G:\n
\forall m \in M: \quad D(sk, E(pk, m)) = m$
RSA – The core idea

- Observation 1:
  - For large primes $p$ and $q$, $n = p \cdot q$ is simple
  - Factoring $n$ to $p$ and $q$ is hard

- Observation 2:
  - Given $p, q$, finding $e, d$, such that $x^{e \cdot d} = x^1$ is simple
  - Extracting the $e$-th root in $\mathbb{Z}_n$ is hard
RSA – Key Generation

- Each participant
  - Chooses two independent, large random primes $p$, $q$
  - Calculates $N = p \cdot q$ and $\varphi(N) = N-p-q+1 = (p-1)(q-1)$
  - Chooses random $e$, with $2 < e < \varphi(N)$, $gcd(e, \varphi(N)) = 1$
  - And calculates $d$ such that $e \cdot d = 1 \mod (\varphi(N))$

- Subsequently:
  - Store $(p,q,d)$ (as secret key sk)
  - Publish $(N,e)$ (as public key pk)
RSA – Encryption and Decryption

**Encryption**
- Given \( pk = (N,e) \):
- \( RSA( pk, m ) : \mathbb{Z}_N^* \to \mathbb{Z}_N^* \); \( c = RSA(e,m) = m^e \) (in \( \mathbb{Z}_N \))

**Decryption**
- Given \( sk = (p,q,d) \):
- \( m = RSA^{-1}( pk, c ) = c^{1/e} = c^d = RSA(d,c) \) (in \( \mathbb{Z}_N \))
Public key encryption using S-TDF (ISO)

- $(G, F, F^{-1})$: secure TDF $X \rightarrow Y$
- $(E_s, D_s)$: symmetric auth. encryption defined over $(K, M, C)$
- $H: X \rightarrow K$: a hash function

**Encryption ($E(\text{pk}, m)$):**

- $x \leftarrow^R X$
- $y \leftarrow F(\text{pk}, x)$
- $k \leftarrow H(x)$
- $c \leftarrow E_s(k, m)$

Output $(y, c)$

**Decryption ($D(\text{sk}, (y, c))$):**

- $x \leftarrow F^{-1}(\text{sk}, y)$
- $k \leftarrow H(x)$
- $m \leftarrow D_s(k, c)$

Output $m$
Intermediate Summary

- You recall the key exchange problem
- You understand the idea of key agreement
- The Diffie-Helman key agreement is easy for you
- You know RSA and you can explain asymmetric and hybrid crypto
**Integrity and Authenticity**

- So far messages can be kept confidential
- *Integrity* of messages not given

\[ c_1 = m \oplus k \]

\[ c_2 = m \oplus k \oplus p \]

**From: Bob**

**E:** $(m \oplus k)$

**From: Eve**

**D:** $(c \oplus k)$

File1 → HDD

File2

File1'
Message Integrity

- **Algorithms:**
  - Tag S: \( M \to T \)  
    \[ M = \{0,1\}^n ; S = \{0,1\}^t \quad \text{with} \quad n \gg t \]
  - Verify V: \( M \times T \to \{\text{yes, no}\} \)

- **Consider your problem:**
  - MDC: Transmission error (bit flips) \( \to \) \( \text{CRC/FEC} \)
  - MAC: Strategic adversary \( \to \) Alice and Bob need a secret: message message tag
Interlude: Collision Resistant Hash Functions

- **Goal:**
  - Map a message of arbitrary length to a (short!) characteristic digest (fingerprint)

- Hash \( H: M \rightarrow S \) with \( M = \{0,1\}^* \) and \( S = \{0,1\}^s \)
  - has an efficient algorithm to evaluate \( H(x) \)
  - is an „onto“ function (surjective, \( Im(H) = S \))
  - maps uniformly to \( S \)
  - creates chaos (slight changes in \( m \) yield large differences in \( s \))
Cryptographic Hash Functions

- Hash functions and security:
  - Compression is irreversible

- We require:
  - Collision resistance
Cryptographic Hash Functions

- We require (ctd.)
  - Pre-image resistance

- 2nd pre-image resistance
The Merkle-Damgard construction

- Given a compression function \( h : \{0,1\}^{2s} \rightarrow \{0,1\}^s \) and
- Input \( m \in \{0,1\}^* \) of length \( L \) and \( PB: = \underbrace{1000...0}_64 \text{ bits} \ll L \)
- Construct \( H \) of \( B = \lceil L/s \rceil \) iterations of \( h \):

- If \( h \) is a fixed length CRHF, then \( H \) is an arbitrary length CRHF

**Proof**: either \( M = M' \), or \( H_{B-i}(m[B-i]) = H_{B-i}(m'[B-i]) \)

- no collision
- collision on \( h \)
Nested MAC

- Let \( F: K \times X \rightarrow X \) be a PRF, define new PRF \( F_{\text{NMAC}}: K^2 \times X^{\leq L} \rightarrow X \)

- Cascade:
  - Why the last encryption with second key?
  - Otherwise: \( \text{cascade}(k, m | | m') = \text{cascade}(\text{cascade}(k, m) | | m') \)
The HMAC (RFC 2104)

- Hashing is fast, but $H(k|\, |m)$ insecure
- Solution: encase message with keys!

\[
\text{HMAC: } S(k,m) = H(k\oplus\text{opad} \, \| \, m[0] \oplus \text{opad} \, \| \, m[1] \oplus \text{opad} \, \| \, m[2] \oplus \text{opad} \, \| \, \text{PB})
\]

(used in TLS, IPsec,...)
Keccak – SHA3

- Specification of two modes:
  - SHA-2 replacement: fixed length output of 224, 256, 384, 512 bits
  - Variable length output: output of arbitrary length
  - Keccak[r,c] with internal permutation Keccak-f[b]

- Construction of two phases
  - Absorb: block-wise input of message
  - Squeeze: output of required bits as hash value
Keccak State – the Sponge

- Keccak[r,c] parameters:
  - bit rate $r$, capacity $c$
  - word length $w = 2^l$ for $l=0,1,...,6$
  - $b = r+c = 5 \times 5 \times w$
  - $c = 25, 50, 100$ (6 for 64-bit systems)
  - $100, 200, 400$
  - $800, 1600$
Permutation in Several Rounds

- Keccak permutes state in $12 + 2l$ rounds
  - 32 bit processor -> Keccak-f[800] -> 22 rounds
  - 64 bit processor -> Keccak-f[1600] -> 24 rounds
  - (Keccak-f[25] -> 12 rounds)

- Five operations for each round:

```
\begin{array}{c}
\text{r} \\
\text{c} \\
\text{s} \quad \text{θ} \quad \text{χ} \quad \pi \quad \text{ρ} \quad \text{ι} \\
\text{s}
\end{array}
```

\[ \text{s} \oplus \text{s} \oplus \text{s} \oplus \text{s} \oplus \text{s} \]
Using SHA3 and SHAKE for Integrity

- Can we implement a secure MAC as:
  - SHA-3(k || m)?
  - How?

- Why?
SHA3 for Confidentiality and Integrity

- Stream ciphers are malleable... Can we achieve authenticated encryption with SHA3?
- How?
Concluding: Security through MACs

- MACs verify integrity of messages
- $S(k, m) ; V(k, m, t)$ → secret key must be used, known to verifier

- MAC hard to forge without secret key, but *integrity purely mutual*:
  - Once key is disclosed, receiver can create arbitrary new tags!
  - $\Rightarrow$ Proof of origin not towards third parties (no non-repudiation!)

- How can we achieve *non-repudiation*?
  - Signature construction from asymmetric crypto
  - $\Rightarrow$ only party in possession of private key can “sign”
  - e.g.: tag$=$ RSA $(pk, h(m)) = h(m)^d \quad (in \ \mathbb{Z}_N)$
Summary of Message Integrity

▪ You can explain the goals and ideas of message integrity

▪ You can explain the Merkle-Damgard construction

▪ You can construct and explain the details of NMAC, and HMAC
Summary

- You know who we are
- You know what to expect from the lecture
- You have seen some trends that are happening
- You have been introduced to Alice, Bob, Eve, and Mallory
- You understand what threats are ... and what this means
- You can tell security goals (CIA!) from security services
- You know how to perform a network security analysis using threat trees ;-)
Papers we want to read:


Questions?