Abstractions & 3G Evolution

G54ACC – IP and Up

Lecture 10

Recap

- Addressing, encapsulation, routing
- Multiplexing, UDP, TCP, reliability
- Address & name translation, layer violation
- XMPP, HTTP, pipelining, P2P, active networks
- IP network management
- Sockets, multiplexing (again), endianness
- Security, OpenID, OAuth, network security

Contents

- System Design
- Common Techniques
- 3G Data Network Evolution

Contents

- System Design
  - Goals
  - Resources
  - Constraints
- Common Techniques
- 3G Data Network Evolution

System Design

- The art and science of extracting what you can from what you have
- Specifically,
  - Solve the problem,
  - Using your resources,
  - Subject to any constraints

Goals and Resources

- Goals are often not well-formed
  - Scalability, modularity, extensibility, elegance
- Resources are a multiplicity of
  - Computation
  - Storage
  - Communications
  - (People?)

System Constraints

- Design
  - Bottleneck resource
    - The most constrained element
  - Scaling
    - To infinity and beyond!
- Resources
  - Latency
    - *c*, the speed of light
  - Throughput, capacity
  - Money

Social Constraints

- Standards compliance
  - Legal implications
- Market conditions
  - (Economic) sustainability
- Labour availability
  - Otherwise nothing gets built
- Future proofing
  - Times change

Contents

- System Design
- Common Techniques
  - Layering & Tunneling; Extensibility
  - Hierarchy; Binding & Indirection
  - State management; Randomization
  - Damping & Hysteresis; Operating timescale
- 3G Data Network Evolution

Layering & Tunneling

- Control complexity by restricting component interactions
  - Simplicity at the cost of performance
  - Often benefit from (restricted) layer violation
- Need not always be strictly layered
  - Protocol heaps [HOTNETS'02]
  - Cycles due to tunneling
- E.g., IP and OSI stacks; IP-in-IP

Extensibility

- “Prediction is very difficult, especially about the future” (Niels Bohr)
- You don’t know how things will be used
  - Or in what circumstance
- Allow for future growth & upgrade
  - “Ships in the night” vs. integrated operation
  - Tunneling as a workaround
- E.g., Version fields, negotiation, explicit state

Hierarchy

- Recursive decomposition into smaller pieces
  - Each typically depends on parent
- No single point of control
  - Though potential single point of failure
- Inter-sibling communication can be expensive
  - May choose to introduce shortcuts (complexity)
- Structure may become too rigid
  - Need to adapt to system dynamics
  - Cf. Pooling, P2P
- E.g., CIDR, IANA

Binding & Indirection

- Need to translate from abstraction to instance
  - Computation *vs*. lookup
  - Distributed *vs*. centralised control
- If well-known, translate (bind) automatically
  - Choice of when: late *vs*. early binding
- E.g., DNS, service lookup
- Combine with multiplexing: *virtualization*

State Management

- Soft state
  - Requires refreshing else deleted on timer
  - Increased bandwidth for automatic clean up after failure
- Explicit state
  - Easier to interpret, manage, error-protect
- E.g., Link state *vs*. distance vector routing

Randomization

- A powerful tool
  - Tie-breaking
  - Security
- Used up and down the stack
- E.g., Ethernet collisions, nonces, designated router election

Damping & Hysteresis

- (Reducing amplitude *vs*. adding memory)
- System state changes based on thresholds so may fluctuate if near a threshold
- Such fluctuations, if rapid, are usually bad
  - Bandwidth overheads
  - Reduced performance
  - Increased likelihood of inconsistency
- Make thresholds *state-dependent*
- E.g., TCP congestion control/RTT estimation; but BGP flap damping is problematic

Operating Timescales

- Not all operations need be equally fast
- Not all operations have same granularity
  - E.g., Once per packet *vs*. once per communication
- Data *vs*. control *vs*. management
- E.g., Connection setup, QoS for packets *vs*. flows *vs*. aggregates

Pipelining

- A complex task can be subdivided
  - Into many independent tasks: fully parallelizable
  - Into many dependent tasks: not much help
- Many *serially dependent* subtasks
  - Each depends only on predecessor
  - Typically improves throughput
  - … while slightly increasing latency
- E.g., compare HTTP/1.1 vs. HTTP/1.0

Batching & Locality

- Grouping tasks to *amortize* overheads
  - Assumes (overhead for N) < N * (overhead for 1)
  - Increased latency accumulating batch *vs*. higher throughput
- Recently used data probably used again, soon
  - Basis for use of caching
  - Increase overheads for statistical benefits

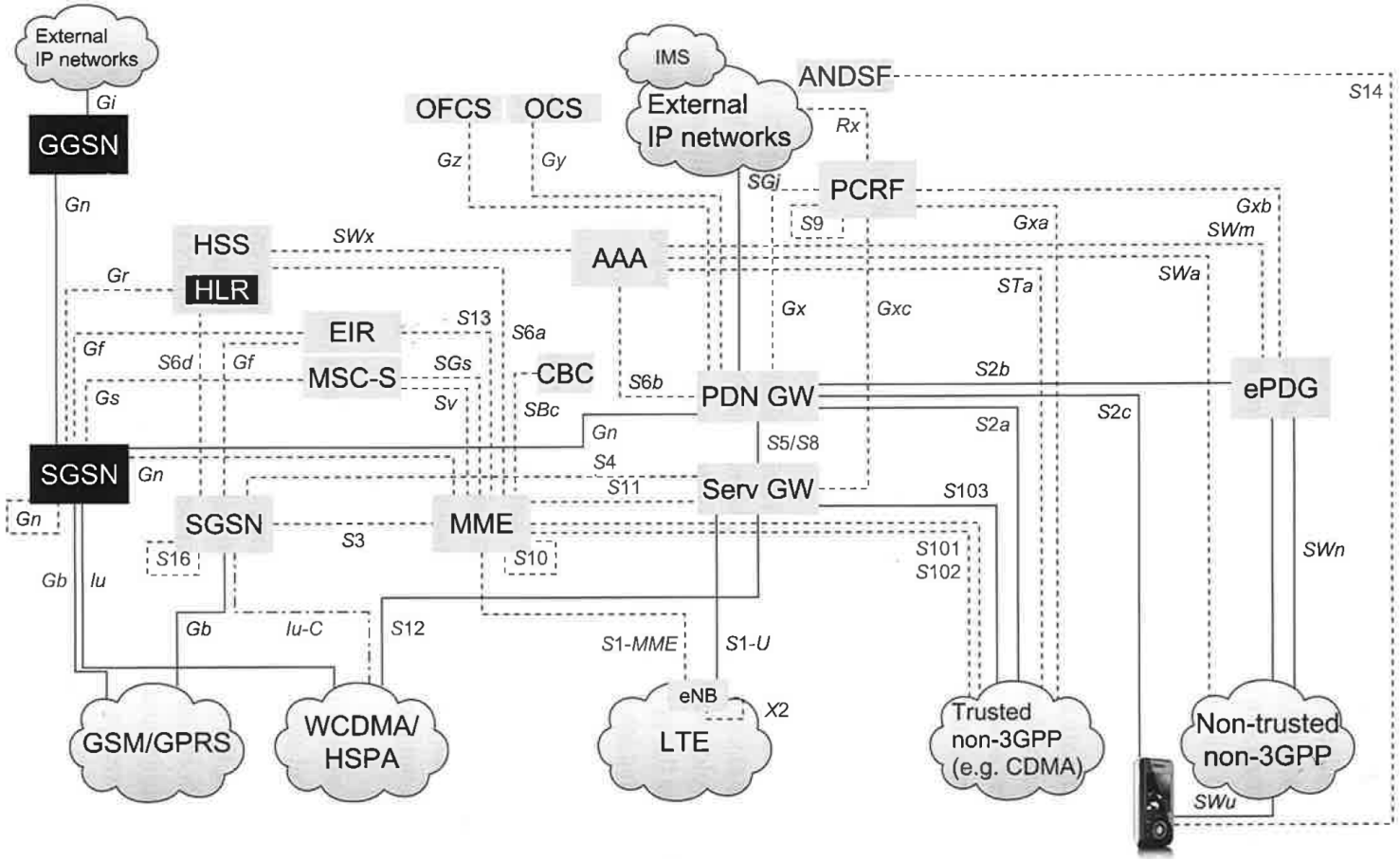
Contents

- System Design
- Common Techniques
- 3G Data Network Evolution
  - Basic LTE operation
  - A different style
  - Some interesting points

EPC Data Networks

- Vision is to provide continuous IP connectivity
- …on resource constrained devices
  - Single radio, battery powered
- …while mobile
  - Geographically, at speed
  - Logically, between network types (2g, 3g, WLAN)
- …with detailed control/management facility
  - Policy
  - Charging
  - Lawful intercept

EPC Data Networks



From Olsson *et al*., Elsevier AP 2009, "SAE and the Evolved Packet Core: Driving the Mobile Broadband Revolution"

Basic LTE Operation

- Tx
  - UE registers with MME
  - Obtains configuration (address, Serv GW, *&c*)
  - Network signals bearers (tunnels) to be set up
- Rx
  - MME pages Serv GWs for UE
  - UE responds, setting up bearers via Serv GW – PDN GW

A Different Style

- It's the network, stupid!
  - Not the stupid network
  - Cf. end-to-end argument and the "dumb network"
  - *Dramatically* more complex accounting required
- E.g.,
  - PCRF, OFCS, OCS
    - Apply complex connectivity and charging policies
  - EIR
    - Deny connectivity to stolen devices
  - ETWS
    - Earthquake and Tsunami Warning Service
    - (to be replaced by Public Warning Service, to benefit US/EU)

Signal Measurement

- eNodeBs (base stations, eNBs) may overlap
  - Which should I pick? How?
- eNB creates idle slots to enable UEs to measure signal strength
  - Only rely on a single radio
  - Can't do simultaneous Rx and Tx
  - Can't be simultaneously on different RANs
- UE switches as desired

Idle Mode

- UE can go into idle mode to save power
  - Less background transmission to network
  - But what about tracking location, e.g., to deliver incoming packets?
- Requires support for paging by MME
  - At request of Serv GW, while it buffers packets
- Periodic updates support vanished terminals
  - Else might keep paging forever
- Trade-off paging overheads *vs*. UE battery

Tracking Areas

- To where should the page be sent?
  - Each UE gets a list of eNBs defining TA
  - UE notifies network if it changes TA
- What about, e.g., trains though?
  - Massive synchronized movement between TAs
  - E.g., Bearer setup/teardown
- Solve by giving out different TA lists
  - Spreads the load
- Further optimize using movement prediction

Pooling

- How to dimension capacity?
  - 2G/3G used hierarchical core
  - UE in a cell connects to 1 base station, &c.
- Region in EPC has a *pool* of weighted MMEs
  - Enables eNBs to distribute load across MMEs
- More efficient use of statistical multiplexing; compare:
  - Peak + 20% everywhere, e.g., region of 1M users
  - Pool of N+1 nodes where each node handles X users

Summary

- System design is hard: an art and a science
- There are many *many* details and interactions involved in any large-scale (networked) system
  - All networked systems are thus interesting at some level
- The Internet and related technologies are good widely deployed, evolved, examples
  - Demonstrate a large toolkit of useful techniques and abstractions
  - Occasionally (partly) by accident rather than design