Advanced Internet Technology - summary

Table of Contents

0-introduction

- Ad hoc 和 mesh networks 是有区别的，http://superuser.com/questions/481145/what-is-the-difference-between-ad-hoc-and-mesh-network-also-with-p2p
  大致意思，一个是点到点，一个是连接的。
- 网络的 reliability 包括：availability, fault tolerance, connectivity
- Napster: directory at central server
- Gnutella: Ping/Pong (keep-alive, for determining active peers)

Query/QueryHit (content search, for locating files)

- Pong and queryhit 是 reverse routed 到原节点。之后 provider 没被墙的话二者直接联系。
- Gnutella 是 power law

1.2-small-1 world

- Cluster coefficient:

$$C(v)=\frac{e(v)}{\deg(v)(\deg(v)-1)/2}$$

- distance d(v,w) is defined as the minimal path length of any path between v and w
- **random graph**

  Erdös-Renyi Random Graphs: random chosen with exactly n vertices and m edges

  Gilbert Random Graphs: With probability p, add an edge (v,w) between any pair of vertices v, w(cluster coefficient 渐进于 p)

  Degree of a certain vertex 是 Poisson distribution

  Theorem:

$$\text{if } p>\frac{\ln n}{n}\text{, a graph } g_{n,p} \text{ will almost sure be connected}$$

  (renyi 也有一条类似定理，就是改成 node degree >O(log n))
- diameter of grid-like is O(√ n)
- A small world network is a network with a **dense local structure** and a **diameter** comparable to a random graph with same numbers of nodes and edges
- **Small world:**

  Watts-Strogatz Model:

  0- Build a ring of n vertices and connect each vertex with its **k clockwise** neighbors on the ring

  1- Draw a random number between 0 and 1 for **each edge**

  2- Rewire each edge with probability p: if the edge's random number is **smaller** than p, **keep the source** vertex of the edge fixed, and choose a new target vertex **uniformly** at random from all other vertices

  P=0,regular, cluster coefficient =3/4 for large k.

  P=1, random, diameter O(log n)

Decentralized routing algorithm by Kleinberg(既是 local 寻路算法，不用知道之前的情况，类似 markov，也是 small world 生成 model)

就是 d-dimensinal space，grid like。$d_M(v,w)$ Manhattan 距离。

Consider a 2-dimensional grid

Add random edges between vertices v and w with a probability of

$P(v,w)\sim d_M(v,w)^{-\alpha}$

The routing algorithm will find 'short' paths, if and only if $\alpha = d$

'Short' paths: length of *O(log n)* from any source to any target vertex

- Power law: $P(k) \sim k^{-\gamma}$ (usually with $2 < \gamma \leqslant 3$)

Barabási-Albert Model

$$\Pi(v) = \frac{\deg(v)}{\sum_{w \in V} \deg(w)}$$

1. Start with a small network (e.g., **10 vertices, 20 edges, at random**)
2. Every time step, add a new vertex x. Add m edges from x to the vertices v that are already there, where the target of the edges is drawn with the probability given by the preferential attachment

Copy model:

0- Each time step randomly copy one of the existing nodes keeping all its links
1- Connect the original node with the copy
2- Randomly remove edges from **both** nodes with a very **small** probability, and for each removed edge randomly draw **new** target nodes

- Under which circumstances is a network both power-law and small world network?

  Node degree: power law distributed, high clustering coefficient, low average path length(注意，aveg path length 就是最短 path 的平均)

1.3-DHT-1

- DHT 节点可以 overlapping of parts，提供 redundancy
- Chord:

  Key associated with data item　　E.g. key = sha-1(data-item-descriptor)

  ID associated with host　　E.g. id = sha-1 (IP address ⊕ port)

  Routing algorithm（就跳到管 k 的前一位，不用到达 successor(k)）:

  Each node n forwards query for key k clockwise

  To farthest finger preceding k

  Until n = predecessor(k) and successor(n) = successor(k)

  Return successor(n) to source of query

  Chord **soft-state approach**:

  Nodes delete (key, value) pairs after timeout (30sec to some minutes)

  Applications need to refresh (key, value) pairs periodically

Last step 可能遭遇坏 node，这时保留好 n 个最邻近的 successor list，然后 successor[0] fails → use successor[1]

Node join 的算法：

a) New node picks ID
b) Contact existing node
c) Construct finger table via standard routing/lookup()
d) Retrieve (key, value) pairs from successor
    Add immediate successor from finger table
    Request successor list from successor（一个一般就够了，因为它也有 list）

Node join 我们还得 update 别人的错误指向：
While p's 2^i-finger points to a node beyond n
  change p's 2^i-finger to n
  Set p to predecessor of p and repeat
Continue with 2^{i+1}
（node 到自己的 predecessor 是提供的，这个更新 $O(\log^2 N)$,状态和查询都是 logN）

1.3-DHT-2

Pastry: (key, value) pairs managed by numerically closest node(可左可右)
b=2, base=4,l=128
routing:

1. Routing table: Long-distance links to other nodes
2. Leaf set: Numerically close nodes
3. Neighborhood set: Close nodes based on proximity metric (e.g., latency)

注意到上面 routing table：in practice, a node is chosen that is close to the present node, according to the proximity metric（topologically）(might be empty if corresponding node unknown)

Routing algorithm:

1. Is K in Leaf Set, route packet directly to that node;
2. If not, determine common prefix (N, K)
3. Search entry T in routing table with prefix (T, K) > prefix (N, K), and route packet to T
4. If not possible, search node T with longest prefix (T, K) out of ***merged set*** of routing table, leaf set, and neighborhood set and route to T (As shown in Pastry paper, this does not happen often)

New node arrival:

1. Hash and get ID
2. Send Join 12333
3. Copy neighbor-set
4. Copy each row of entries
5. Copy leaf-set of old destination node
6. Take node-ID just visited and modify table
7. transmits a copy of its resulting state to each of the nodes found in its neighborhood set, leaf set, and routing table and they update

如果 entry in routing table corrupted：问同一行，还不行，问下一行。
Hops and storage are O(log N) 具体来说 hop is O(log $_2{}^b$ N)

1.3-DHT-3

CAN:

- Search : $O\left(\frac{D}{4}N^{\frac{1}{D}}\right)$
  令 D=log(N)，则为 log(N), p.s. $N^{(1/\log N)}=e$
  Node state：O(D)

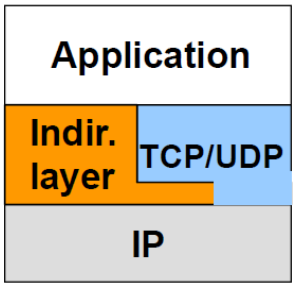
- Two zones are neighbors if d-1 dimensions overlap
- Expected number of neighbors: O(2·D)
- Routing: The neighbor with the shortest distance from the target is the next hop(比较的是邻居 node 坐标和查找 key 的坐标，最后找到 node)
- Leave or fail: 不能融合时 neighbor with smallest number of key gets both regions to manage (no merging!)
- More concurrent coordinate systems:
  1. Mapping of nodes onto r **different** coordinate systems via different hash functions
  2. **Keys use same** hash function for all realities
  3. **All realities are checked** in each routing step
- More dimension shorter path, but more realities more redundancy
- Pastry is better than Chord because of proximity support(就是如 latency 这种揭示 locality)
-

	Chord	Pastry	CAN
States per node	*O(log N)*	*O(log N)*	*O(D)*
Path length (Routing)	*O(log N)*	*O(log N)*	$O(\frac{D}{4}N^{\frac{1}{D}})$
Join of node	*O(log² N)*	*O(log N)*	$O(DN^{\frac{1}{D}})$
Leave of node	*O(log² N)*	*?*	*?*

1.4-i3-1

- Internet Indirect Infrastructure (i3) 为了简化 point-to-point communication abstraction to provide services like multicast, anycast, and mobility，在 DHT 上层。而 p2p application 在 i3 上面

- I3 在 ip 上面，是层 indirection layer



- Unicast:1 to 1
  Multicast: 1 to many
  Broadcast: 1 to all
  Anycast: 1 to 1 of many
- Properties of the indirection service
  1. Implementation via DHTs
  2. **Best-effort-type** services
  3. Triggers have to be refreshed periodically **(soft state)**
  4. Reliability-, flow-, congestion- control etc. realized **in end systems**
- Anycast: with longest prefix match, selection of postfix enables different strategy
- 到 end 是去掉 stack 中第一个，到 trigger 是换 stack（i3 destination address 有两种:address of computer; address of an ID）
- Search for nodes locate closely to current node: Choose **m** IDs randomly and determine RTT

Separation of data and control plane

	Host	Infrastructure
Internet & Infrastructure overlays		Data plane Control plane
P2P & End-host overlays	Data plane Control plane	
i3	Control plane	Data plane

2.1-cloud computing-1

- Properties:
  1. No additional charge for scaling
  2. Illusion of infinite scalability

- Properties:
  1. Resource pooling
  2. Broad network access
  3. On-demand self-service
  4. Rapid elasticity

5. Measured service
6. “no-need-to-know”
7. “flexibility and elasticity”
8. “pay as much as used and needed”
9. “always on!any where and any place”

- (IaaS)Infrastructure(as a Service);
  (PaaS)Platform;
  (SaaS)Software

- Cassandra: Distributed Database Management Systems
- Scale up: one bigger
  Scale out: more nodes

- CAP Theorem

1. Consistency: all nodes have same data at any time
2. Availability: the system allows operations all the time
3. Partition-tolerance: the system continues to work in spite of network partitions

- Cassandra: Eventual (weak) consistency, Availability, Partition-tolerance

- Cassandra data model: Not Only SQL; no schemas; support get(key), put(key,value)

- Different Replication Policies:

1. Rack Unaware: replicate data at N-1 successive nodes after the responsible node
2. Rack Aware: uses a coordinator which tells nodes the range they are replicas for
3. Datacenter Aware: similar to Rack Aware but coordinator is chosen at datacenter level instead of rack level

- Cassandra uses a **Ring-based DHT but without routing**
- Write operation:
  1. Log it in the disk commit log
  2. Modify the appropriate memtables(in memory)
  3. Later, when memtable is full or old, flush to disk

- Bloom filter false positive:

number of items to be stored
k : number of hash functions
m : size of bit map and range of hash functions

$$\left(1-\left(1-\frac{1}{m}\right)^{kn}\right)^{k} \approx \left(1-e^{-\frac{kn}{m}}\right)^{k}$$

- Optimal k:

$$k = \ln 2 \cdot \frac{m}{n}$$

- Read-repair makes read slower than write

2.3-MapReduce

- Amdahl's Law:

$$S = \frac{1}{(1-p) + \frac{p}{n}}$$

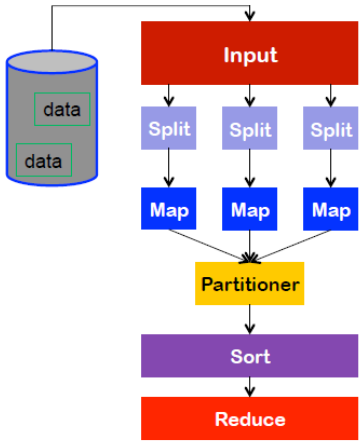
- Two types of parallelism in Cloud

Request-Level Parallelism (RLP)(举例 google searching，用 replicas)

Concurrent processing of multiple requests

Data-Level Parallelism (DLP)（MapReduce）

Concurrent processing of multiple data



Master/slave: job tracker, task tracker, pull strategy

Map task scheduling: locality

Reduce task scheduling: not locality, but the most urgent

- speculative execution: Only one copy of a straggler(average - 20%) will be run speculatively

3-SDN-1

- SDN is concept of separation of control- and data plane
- OpenFlow is communication interface(protocol) between control- and data plane of an SDN architecture
- OpenFlow handling rules:
  Pattern: match packet header bits
  Actions: drop, forward, modify, send to controller
  Priority: disambiguate overlapping patterns
  Counters: #bytes and #packets

- Network OS, distributed, get information from forwarding elements through OpenFlow
  Control program, not distributed, output configuration to OS

- Middleboxes
  1. High Capital and Operating Expenses
  2. Time Consuming and Error-Prone
  3. Physical and Overload Failures

  Carrier networks are complex
  Launching new services takes long

Category	SDN	NFV
Reason for Being	Separation of control and data, centralization of control and programmability of network	Relocation of network functions from dedicated appliances to generic servers
Target Location	Campus, data center / cloud	Service provider network
Target Devices	Commodity servers and switches	Commodity servers and switches
Initial Applications	Cloud orchestration and networking	Routers, firewalls, gateways, CDN, WAN accelerators, SLA assurance
New Protocols	OpenFlow	None yet
Formalization	Open Networking Forum (ONF)	ETSI NFV Working Group

- REFLEXES: in-network control, proactive reaction, deal with problems of latency and variability
- Reflexes = precomputed reactions; 横向纵向都变短了。Update from original controller
- OSI

Physical (Layer 1)
Data Link (Layer 2)
Network (Layer 3)
Transport (Layer 4)
Session (Layer 5)
Presentation (Layer 6)
Application (Layer 7)

- IP vs MAC

IP address or Internet Protocol address is the address assigned to your mobile, printer or computer by the network that uses Internet protocol for communication.(network layer3)
MAC address is your machine address. This address will never change. It is the unique machine address given to your device.（data link layer2）

4.1-sensor network

- QoS: quality of service
- MANETs vs WSN

MANETS are usually “closed” to humans, in the sense that most nodes in the network are devices used by human beings (e.g., laptop computers, PDAs, mobile radio terminals, etc). On the other side, sensor networks do not focus on human interaction, but on the interaction with the environment. Nature of changing topology is quite different in WSNs and MANETs

4.2-1

- Dynamic voltage scaling (DVS) $P \sim f \cdot V^2$
- $$E_{tx} = T_{start} \cdot P_{start} + \frac{n}{R \cdot R_{code}} \cdot \left(P_{txElec} + \alpha_{amp} + \beta_{amp} \cdot P_{tx}\right)$$

4.2-2

- Energy consumption for transmitting over distance d is proportional to $c \cdot d^{\alpha}$

Topology control:

Flat network

Hierarchical network: backbones(back 连在一起); clustering(clusterhead 是分开)

4.3-MAC

- Hidden nodes mean increased probability of collision at receiver end.

One solution to avoid this is to have the channel sensing range much greater than the receiving range. Another solution is to use the Multiple Access with Collision Avoidance (MACA).

- Multiple Access with Collision Avoidance (MACA) is a slotted media access control protocol used in wireless LAN data transmission to avoid collisions caused by the hidden station problem and to simplify exposed station problem.

The basic idea of MACA is a wireless network node makes an announcement before it sends the data frame to inform other nodes to keep silent. When a node wants to transmit, it sends a signal called Request-To-Send (RTS) with the length of the data frame to send. If the receiver allows the transmission, it replies the sender a signal called Clear-To-Send (CTS) with the length of the frame that is about to receive.
Meanwhile, a node that hears RTS should remain silent to avoid conflict with CTS; a node that hears CTS should keep silent until the data transmission is complete.
WLAN data transmission collisions may still occur

- Exposed node means denied channel access unnecessarily which ultimately results in under-utilization of bandwidth resources. It also results in wastage of time-resource.
- It allows several users to share the same frequency channel by dividing the signal into different time slots. The users transmit in rapid succession, one after the other, each using its own time slot.

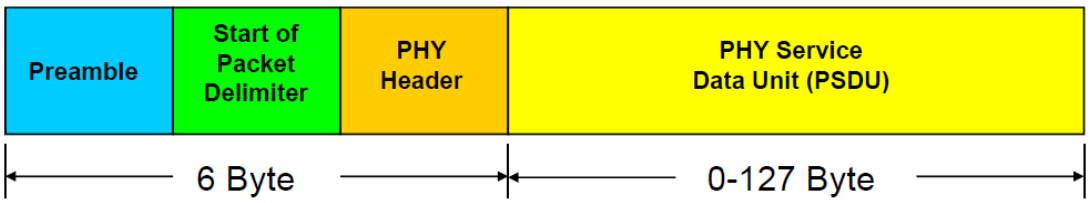
- CSMA/CA is in-band control frames（这是 ethernet，对 wsn 不太合适(不仅实现，电量，还有 hidden,exposed)）

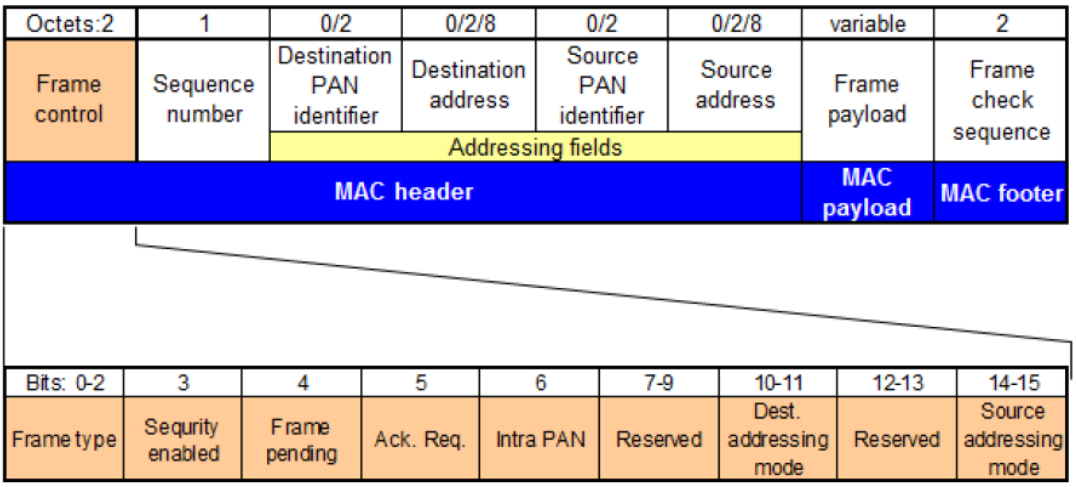
- Link layer: Medium Access Control (MAC)
  Network layer: Addressing and Routing

  感觉本章应该是一步一步递进的
  下面都是 contention-based
- MACA:RTS(request to send);CTS(clear to send)（问题是睡觉就听不到 rts，cts）
- Sensor-MAC:分 active，sleep。Active 又分成 SYNCH phase(令时间同步，share 自己啥时候睡) and data exchange phase（这个的问题就是同步，会形成 synchronized islands,耗电）
- Peamble Sampling(B-MAC):用长序言，data 放后面，大家随便醒
- WiseMAC: combine B-MAC and S-MAC;频繁交流的缩短 preamble

  下面是 schedule-based
  LEACH: setup phase(其实是 CSMA 竞争的), steady-state phase;

  之后介绍 802.15.4
- 802.15.4 is a technical standard which defines the operation of low-rate wireless personal area networks (LR-WPANs)（定义了 physical layer and MAC of LR-WPANs）
- Frame structure(前者在 layer1，后者 layer2):





MAC 分 4 类：beacon frame; command frame; data frame; acknowledge frame
Unslotted; slotted mode
(802.15.4 is contention-based, combined with reservation of slot)

- Frame structure: data link layer
  MAC frame format: physical layer
  https://en.wikipedia.org/wiki/Ethernet_frame
  and especially
  http://www.sharetechnote.com/html/WLAN_FrameStructure.html
- channel hopping: transmitting radio signals by rapidly switching a carrier among many frequency channels

4.4-addressing and routing

下面介绍的是 ad hoc（不是 wsn）

- （proactive）Optimized Link State Routing(OLSR)：broadcast from node X is only forwarded by its multipoint relays（two-hop neighborhood）
- （proactive）Destination Sequence Distance Vector (DSDV)：Add aging information helps to avoid routing loops（use bellman-ford）
- （reactive）Dynamic Source Routing (DSR): Store ID of each crossed router in a discovery packet; Fill in discovered route as path information into each data packet(就是最有名的 flood，最后原路返回)
- Ad-hoc On Demand Distance Vector (AODV)：similar to DSR, but maintain routing tables instead of using source routing

下面是 wsn

- Geometric Perimeter State Routing (GPSR):
  Use greedy, “most forward within range r” routing as long as possible
  If no progress possible: Switch to “face” routing

1. Face: largest possible region of the plane that is not cut by any edge of the graph; can be exterior or interior
2. Send packet around the face using right-hand rule
3. Use position where face was entered and destination position to determine when face can be left again, switch back to greedy routing

- Beacon Vector Routing (BVR):
    1. Greedy, geographic forwarding(只要哪一步没有进步就 fallback)
    2. Fallback Mode
    3. Scoped Flooding
- One option for expressing Aggregation Request is to Use database abstraction of WSN(SQL)
- Partial state records to represent intermediate results

- bellman-ford algorithm

```
function BellmanFord(list vertices, list edges, vertex source)
    ::distance[],predecessor[]

    // Step 1: initialize graph
    for each vertex v in vertices:
        distance[v] := inf                  // At the beginning , all vertices have a weight of infinity
        predecessor[v] := null              // And a null predecessor
    distance[source] := 0                   // Except for the Source, where the Weight is zero

    // Step 2: relax edges repeatedly
    for i from 1 to size(vertices)-1:
        for each edge (u, v) with weight w in edges:
            if distance[u] + w < distance[v]:
                distance[v] := distance[u] + w
                predecessor[v] := u

    // Step 3: check for negative-weight cycles
    for each edge (u, v) with weight w in edges:
        if distance[u] + w < distance[v]:
            error "Graph contains a negative-weight cycle"
    return distance[], predecessor[]
```

- Minimum battery cost routing: Path metric: Sum of reciprocal battery levels
- ETX of a route is sum. ETX of single is 1/(df*dr)
- Beacon vector routing: The key piece of intuition driving our design is that it is more important to move towards beacons than to move away from beacons

4.5-localization

- Received Signal Strength Indicator(RSSI)(按此式推出 d):

$$P_{\text{recv}} = c\frac{P_{\text{tx}}}{d^{\alpha}}$$

- Time Difference of Arrival (TDoA)：发射俩不同速度的，不过 expensive/energy-intensive hardware

- GPS 定位：Receiving time signals of ***four***

- 三边的公式可以推，多边的 minimize mean square error
- Range-free vs range-based
  a) Range-based
    i. Use exact measurements (point-to-point distance estimate (range) or angle estimates)
    ii. More expensive
    iii. Ranging: the process of estimating the distance between the pair of nodes
  b) Range-free
    i. Only need the existences of beacon signals
    ii. Cost-effective alternative to range-based solutions
- Path loss (or path attenuation) is the reduction in power density (attenuation) of an electromagnetic wave as it propagates through space

4.6-time sync

- $L_i(t) = \theta_i H_i(t) + \Phi_i$ （H is register value, L is software clock）
- Modify theta and fi, not H
- ppm (pulses per million): counts the additional pulses or lost pulses over the time of one million pulses at nominal rate
- LTS(Lightweight Time Synchronization)，
  Assume only phase shift, no drift
  delta=(Li(8)+Li(1)-Lj(5)-Lj(6))/2 (别看个数多，实际记 timestamp 里就这四个)

$$\Delta = \frac{L_i(t_8) - L_j(t_6)}{2} - \frac{L_j(t_5) - L_i(t_1)}{2}$$

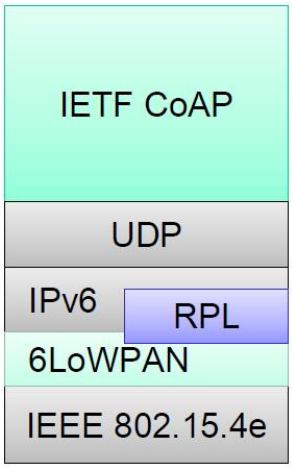
- 算上 transmitter and receiver: $4\sigma^2$

几 hop，就乘以几

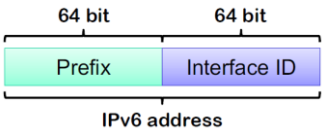
HRTS(Hierarchy Referencing Time Synchronization):

$$\begin{aligned} \Delta &= \\ L_j(t) - L_R(t) &= O - (L_i(t_2) - L_j(t_2')) \end{aligned}$$

5-IoT



- 6LoWPAN:



1. Addressing:
2. Header compression:
   Header 包括好几个地址，以及其他。
   - Omission of redundant header information
     - E.g., version, payload length
   - Compression of dynamic header content
     - E.g., next header, traffic class, flow label
   - IPv6 address compression
     - Prefix
       - Omit link-local prefix or compress prefix for well-known “contexts”
     - Interface ID
       - Omit for link-local communication (same as MAC address)
3. Packet fragmentation:
   1) First fragment needed for routing decision
   2) Packet reassembly at each hop to derive IP routing decision
4. Neighborhood discovery optimizations:
   Commissioning;
   Bootstrapping;
   Route initialization

- Ipv6 neighbor discovery:

1. Locate neighboring routers with Router Solicitation or wait for periodic Router Advertisement
   Advertisement contains prefix information for the advertising router
2. Generate address based on prefix
3. Perform Duplicate Address Detection via Neighbor Solicitation
4. Resolve MAC addresses for known IP addresses via Neighbor Solicitation

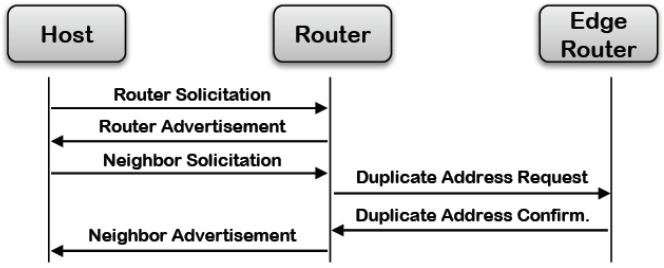
Why 6LoWPAN 不行：

1. Only 6LoWPAN edge router maintains prefix; Routers inside 6LoWPAN network share common prefix
2. Heavy use of multicast traffic
   a) Results in flooding on wireless medium
   b) Especially for messages destined to all-nodes or all-routers multicast groups
3. Periodic router advertisements
   a) Assumption that nodes are always on

6LoWPAN address configuration:

1. 6LoWPAN routers behave as hosts to upstream routers
   a) One-hop broadcast of router solicitation to find initial set of routers
   b) Configure global address based on prefix received in router advertisements
      i. Advertisements also contain 6LoWPAN header compression contexts
2. Additionally behave as routers to downstream hosts
   a) Multi-hop prefix and context dissemination
3. Hosts
   a) Address configuration based on prefix
   b) 6LoWPAN IP compression based on received context information

- Beaconless mode uses a pure CSMA channel access and operates quite like basic IEEE 802.11 without channel reservations. Beacon-enabled mode is more complex, with a superframe structure and the possibility to reserve time-slots for critical data
- 6LoWPAN node registration(避免 multicast)



- A RPL Instance is a set of one or more DODAGs that share a RPLInstanceID. At most, a RPL node can belong to one DODAG in a RPL Instance. Each RPL Instance operates independently of other RPL Instances
- Rank: path calculation according to objective metric
  1. Scalar that represents relative position within a DODAG
  2. Used to avoid and detect loops

3. Strictly increasing from the root

- RPL 3 ICMPv6 message:

DAG Information Object (DIO)

Discover a RPL Instance, learn its configuration and select DODAG parents

DAG Information Solicitation (DIS)

Solicit a DODAG Information Object from a RPL node

Destination Advertisement Object (DAO)

Propagate destination information upwards in the DODAG

- RPL: Optimized for many-to-one and one-to-many traffic patterns; Routing state is minimized

CoAP:

Restful; UDP;

CON, NON, ACK, RST

Observe:0

ACK block(nr=2,m=1,sz=128)…

Cache

Questions

1. ppt4.2, P19, yellow remark
2. ppt4.3, P17, 知道 rts and cts，但是啥时候恢复正常？P20 显示 receiver 是会有 ack 的
3. ppt4.4 P9, 算 transmission 期望是用 sum，不过后来来回算的是乘积的倒数？（之所以如此，感觉是因为 reverse delivery 并没有尽力，而是收到一个才应答一个。而多步，因为已经有应答作为保证，就可以减少尝试）
4. ppt4.4 P17, GPSR 的算法感觉没法保证找到路啊。看看复杂的例子
5. ppt4.5 P18, 确定 angle 的方法可以了解一下
6. ppt4.6 P6, 这 fi，h 啥的深入理解
7. ppt4.6 P19, 3packets 错了吧
8. ppt5, rpl, rank 的计算诡异
9. ppt1.1, gnutella0.6 也是 power law?
10. Ppt1.2-1,P13 为啥泊松，应该是 binomial。
11. Ppt1.2,P4,yellow remark

12. Ppt1.3-dht-1, P20, yellow remark, 大于号
13. Ppt1.3-dht-2,P2,难道 tie 时连两个？
14. CAN: Memory requirement: O(D) = O(1)为毛？
15. 1.4-i3-2,P15,yellow remark, 图没看懂
16. 2.2-2, P18, index file, data file, bloom filter 关系, Cassandra 论文
17. Exercise6 neighbor discovery 简直凶残
18. multi-hop routing protocol 中 loop 生成原因以及检测的方法