

Q & A Session for Coursework 4

GZ01/3035 Networked Systems

Georgios Nikolaidis

(some slides contributed by Lynne Salameh)

Department of Computer Science

University College London

Distance-Vector Routing

- One of the most important classes of routing protocols
- Simple and elegant
- Works well on small networks
- Interesting behaviour in dynamic conditions

Implementing Distance-Vector Routing

- Build a virtual router
- Most of the code already given
- Code in Java

Coursework Tasks

- 3 stages:
 - Baseline DV
 - Split Horizon with Poison Reverse
 - Expiration of table entries
- No need to implement triggered updates

Coursework Tasks

- 5 test configurations provided
- Tests are (.cfg) files
- Each stage has targeted set of tests

Baseline DV

- Implement “vanilla” DV routing in DV.java:
 - DV (interface RoutingAlgorithm)
 - DVRoutingTableEntry (interface RoutingTableEntry)
- 2 test cases: test1.cfg and test2.cfg

Split Horizon with Poison Reverse SH/PR

- Performance enhancement
- 2 test cases: test3.cfg and test4.cfg
- Step 1:
 - Run tests with SH/PR disabled.
 - What do you observe?
- Step 2:
 - Implement and enable SH/PR.
 - What do the 2 tests output now, and why?

Expire Routing Table Entries

- Stale links should not persist forever
- Enforce deadline for expiring entries
- Read RIP RFC2453
- Same timing constraints, as multiple of update interval
- Note: RFC deals with possibility of lost packets
- Test test5.cfg

So, how do I start?

- Use **lab machines** (remotely accessible)
 - ssh into one of the machines in coursework handout
 - Linux remote desktop:
<http://www.cs.ucl.ac.uk/csrw>
- *tar vzx* `~ucacbnk/gz01-2014/cw4.tar.gz`
- *make* and *make javadoc*

So, how do I start?

- All your code goes in DV.java
- Implement all methods that are empty
- 2 classes:
 - DV implements RoutingAlgorithm
 - DVRoutingTableEntry implements RoutingTableEntry

How do I test?

- Configuration files (.cfg)
- *java Simulator config.cfg*
- 5 test configurations provided
- The **machine code** of the solution also provided
- IMPORTANT: See coursework text about how to use solution!!!

Configuration Files

```
updateInt 10  
preverse off  
expire off
```

#router ID NumIfaces RoutingAlg

```
router 0 2 DVsolution  
router 1 2 DVsolution  
router 2 2 DVsolution
```

#links src.siface.weight dst.diface.weight

```
link 0.0.1 1.0.1  
link 1.1.1 2.0.1  
link 2.1.1 0.1.1
```

#send time src dst

```
send 10 0 1
```

```
.....
```

#link down: time router.iface router.iface

```
downlink 10 1.1 2.0
```

#link up: time router.iface router.iface

```
uplink 12 1.1 2.0
```

```
dumpPacketStats 14 all
```

```
dumpprt 14 all
```

```
stop 100
```

Flags

- *preverse* and *expire* in (.cfg) files
- Implement:
 - `setAllowPReverse(boolean flag)`
 - `setAllowExpire(boolean flag)`
- Use in code around enhancements

Simulated Events Order

- Simulator calls Router.go():
 - Process packets
 - Tidy table
 - Send routing message

Does it work?

- Yes, if it has the same behaviour as the solution
 - Same routing table contents
 - Same routing decisions

Does it work? (2)

- Once more: check in handouts how you run the solution!
- Output of dumpprt MUST be:
Router [n]
d [destid] i [intid] m [metric]
...
– And only the above!

Help!

- Read the lecture notes, textbook
- Read the code/documentation
- RIP RFC2453
- Piazza
- Office hours