Tamanoir : a High Performance Active Network Framework

Jean-Patrick Gelas and Laurent Lefèvre

RESAM Laboratory - INRIA Action RESO

llefevre@resam.univ-lyon1.fr

http ://resam.univ-lyon1.fr/TAMANOIR

What do we expect from networks ?

Performance

- low latency, high bandwidth, low packet lost, fault tolerance...

Services

- QoS, multicast, DiffServ...

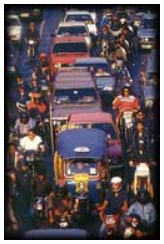
Elegance

- easy plug-and-play new services ;
- dynamic deployment of new protocols.

But reality is really different ! ! !

What do we obtain from networks ?

Performance



- traffic jam, reduced bandwidth

Services

- only few services available (best effort)...

Elegance

- static protocols...

An active approach ?

Plan

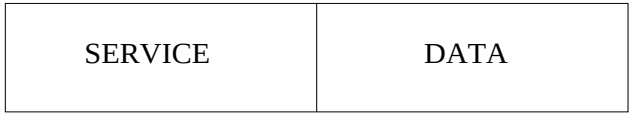
1. Introduction
   - Active networks : various approaches
   - Existing environments : ANTS and PAN
2. A new model for High Performance Active Networks
   - Packets, services, active nodes.
   - Implementation strategies (language, compiler)
   - Tools box : TAMT
3. Experiments :
   - latency, bandwidth...
   - qualitative aspects
4. Conclusion and future works

Active networks

- active research field (since 1995) ;
- Goal : Internet use in a more intelligent way
- Intelligent switches (nodes) run applications (services) to process data stream ;
- Dynamic services deployment.

Two approaches

Active packets :



- A service in each data packet

Active/programmable nodes :

- Services deployed concurrently to data one time
- Each packet contains a reference to a service present on the active node.

Now

Passive networks

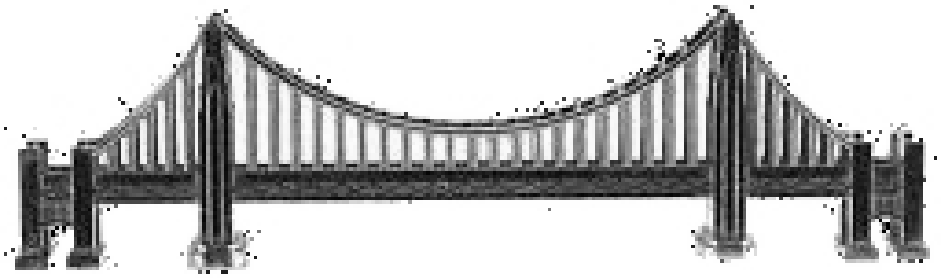
- High performance Core network
- IP
- Long distance

Active networks

- Services
- Pseudo dynamic deployment
- Optimized traffic

Our approach

Link the two worlds



Réseaux passifs

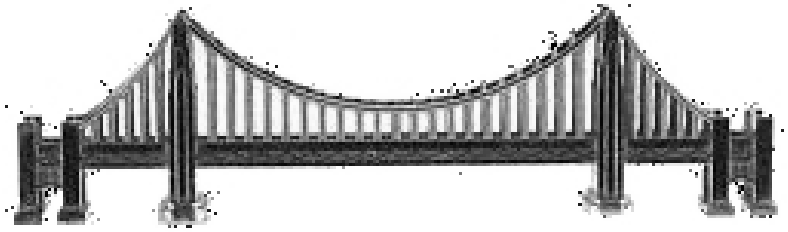
- Core Network Hautes performances
- IP
- Longue distance

Réseaux actifs

- Services
- Personnalisation
- Trafic optimisé

Our approach

Link the two worlds



→ High performances Active Networks

- ✓ High performance stream process
- ✓ Long distance
- ✓ Dynamic services

Existing Environment (I)

ANTS [David J.Wetherall, Fév. 1999]

- written in JAVA ;
- run in a JVM ;
  - security, code validation. . .
  - interpreted bytecode (portability)
- complete model ;
- limited implementation ;
  - no multi-application ;
  - no high performances.

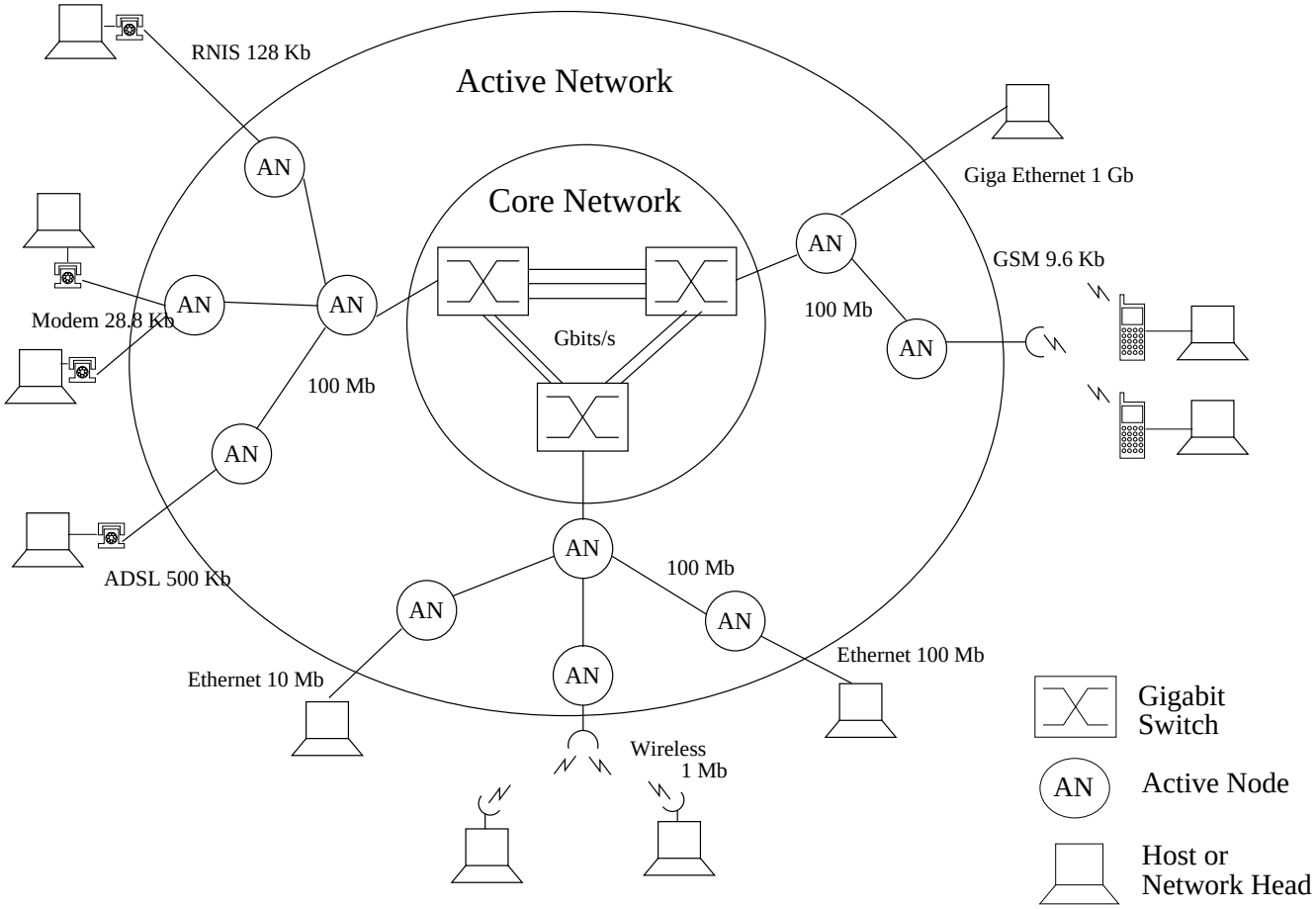
Existing Environment (II)

PAN [Erik L.Nygren, Fév. 1998]

- high performance ;
- written in C ;
- two implementations (kernel, user-space) ;
- good raw performances ;
- no security

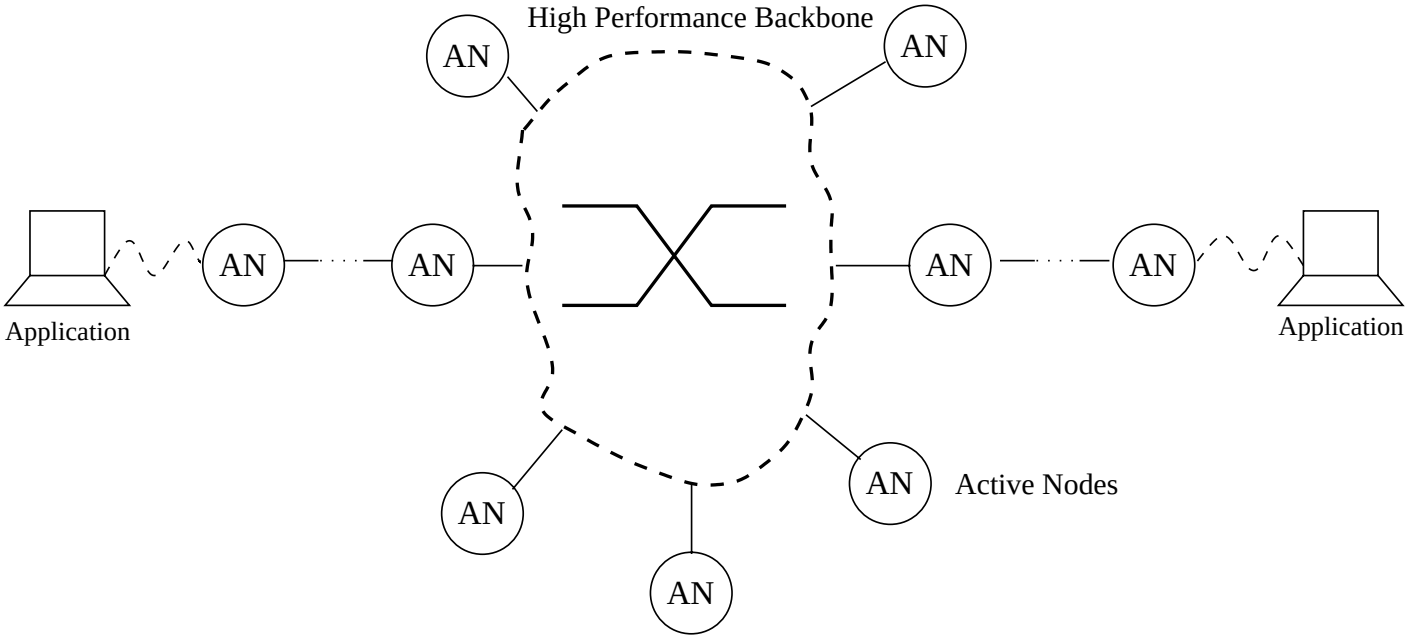
A new model of High performance Active Network

- peripheral model ;
- optimized active nodes ;
- standard ANEP packets ;
- multi-services ;
- dynamic deployment ;
- complete environment (tool box).



Active nodes on peripheral

High performance passive core network



Data streams

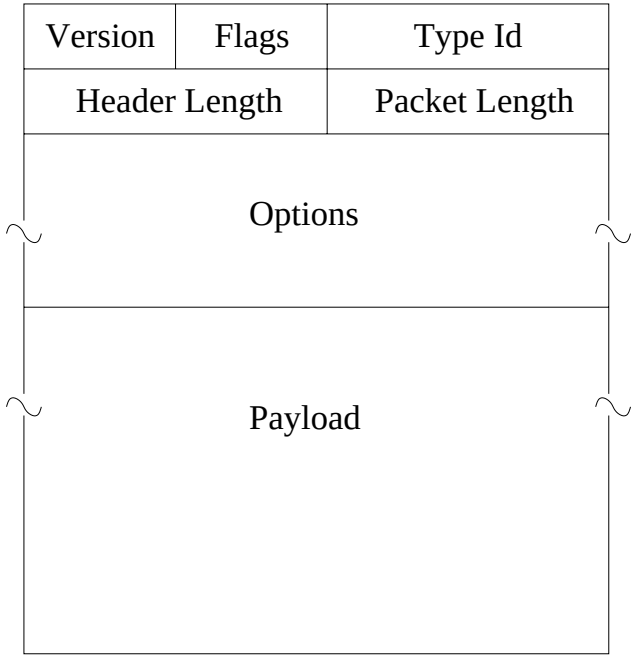
TAMANOIR Environment Implementation

- *JAVA* Language
  - ⊕ high level language, standard libraries ;
  - ⊖ interpretation too slow.
- Using *GCJ* : GNU Compiler for Java
  - portable and optimized JAVA compiler ;
  - native code execution.

Java (javac)	C (gcc)	Java (gcj)
2459	125	176 *(×13.9)*

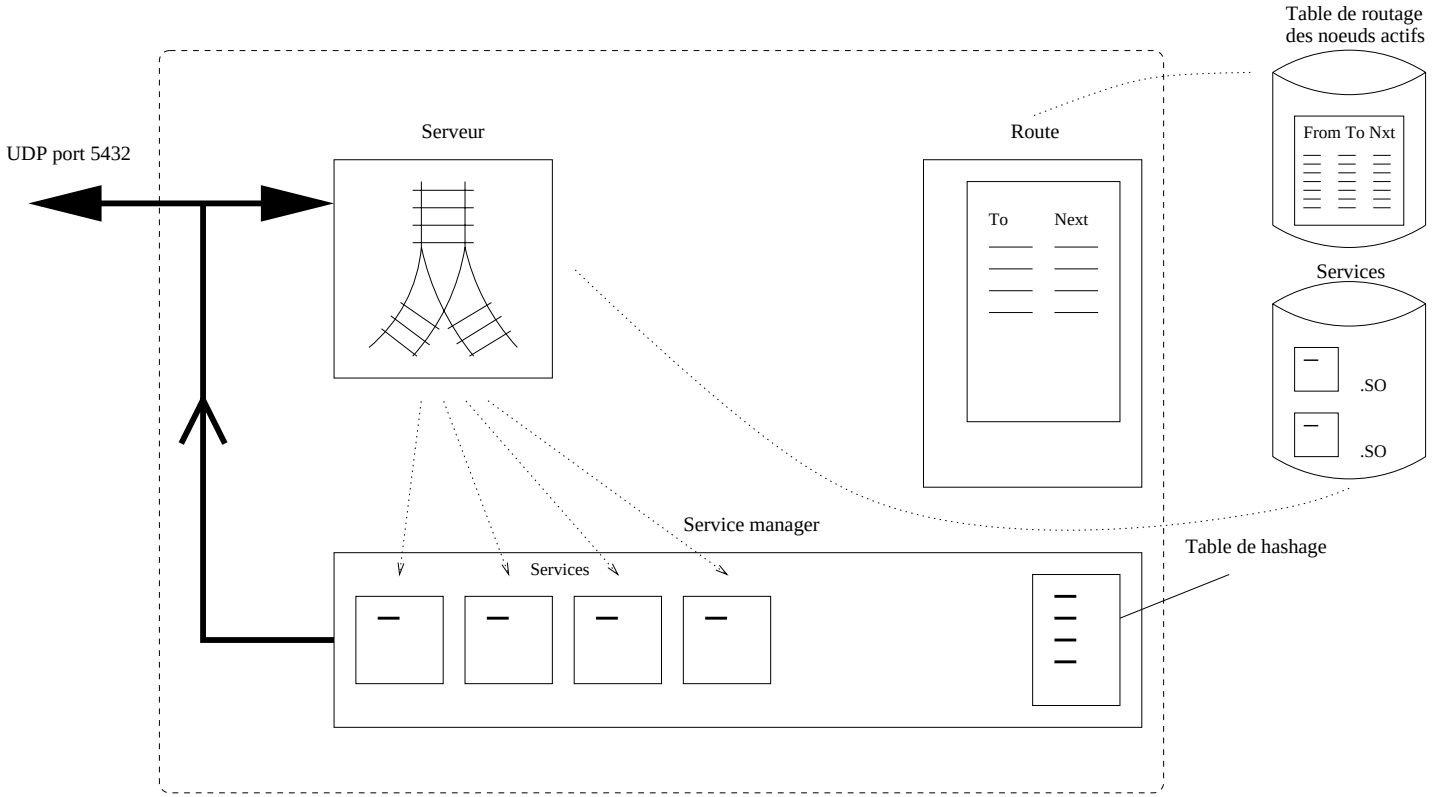
10.000.000 `for` loop (ms.)

Packets/capsules



ANEP packets *(Active Network Encapsulation Protocol)* defined by *Active Network Group*

A TAMANOIR Active Node



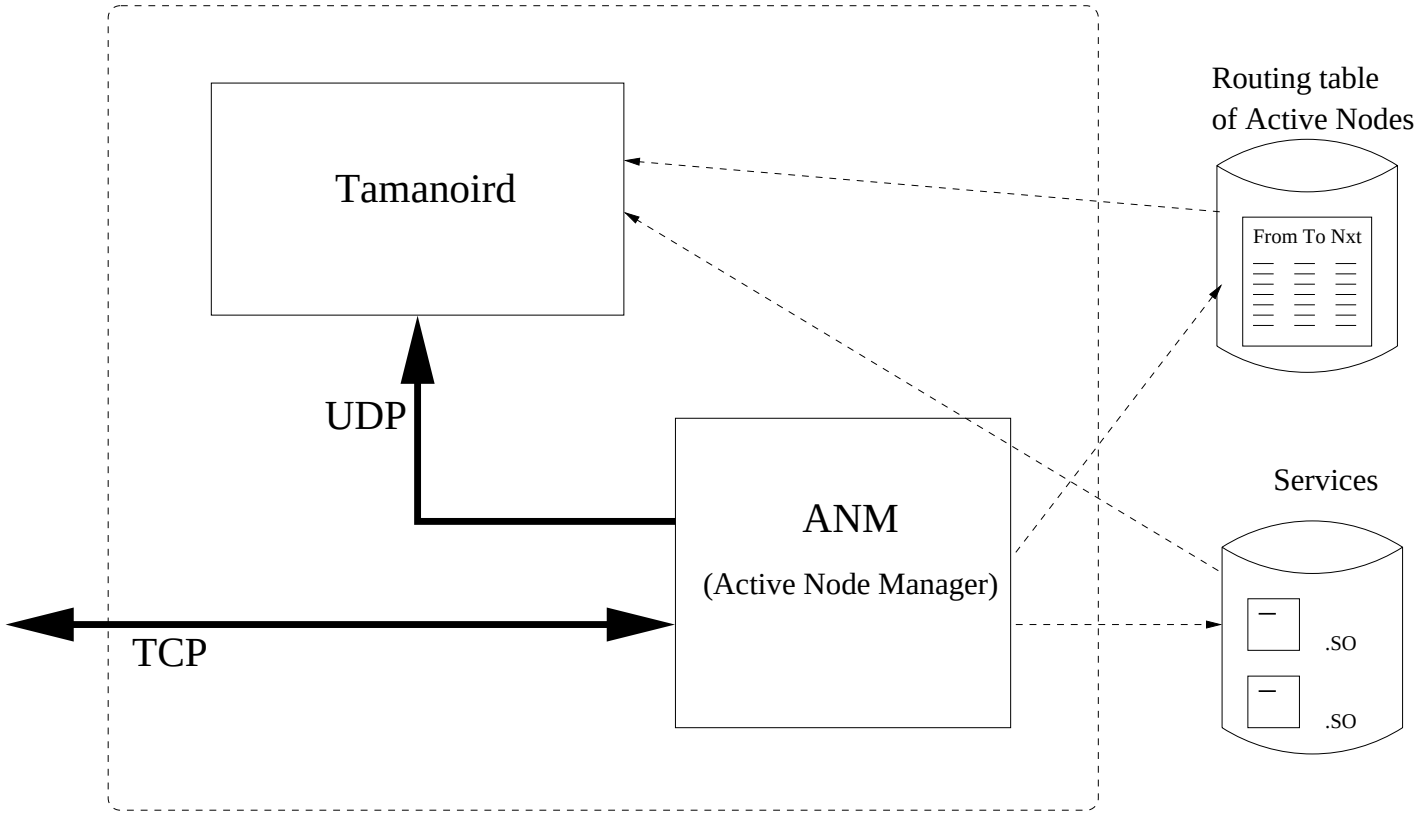
- Daemon waiting UDP packets
- Few-copies protocols
- Multi-threading

Services

- A service is a specific application with generic methods (`send()`, `recv()`)
- Each service is derived from a generic class.

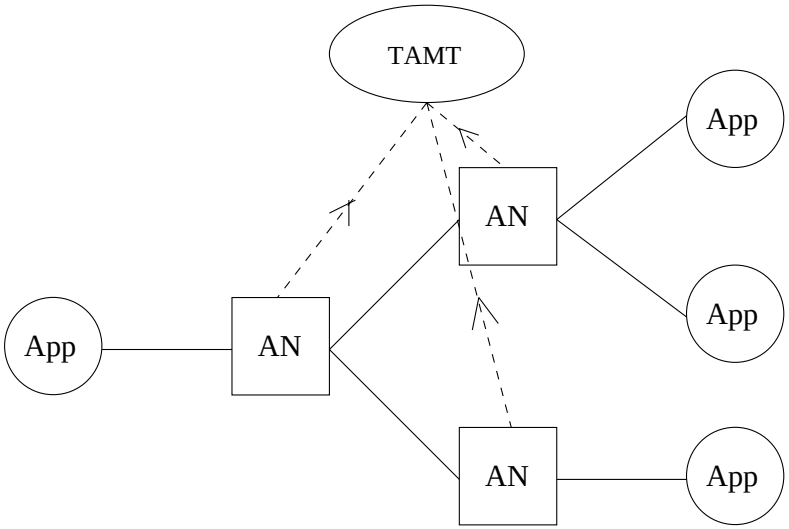
```
public class Service {
  (...)
  public void recv( String srcId, destId, lastId, byte [] payload ) {}
  public void send( String srcId, destId, lastId, byte [] payload ) {}
  (...)

public class PingService extends Service {
  (...)
  public void recv( String srcId, String destId, String lastId, byte [] payload ) {
    if( localId.equals(destId) ) {
          send( destId, srcId, lastId, payload ); // come back to the sender
    } else {
          send( srcId, destId, lastId, payload ); // forward
    }
    (...)
```

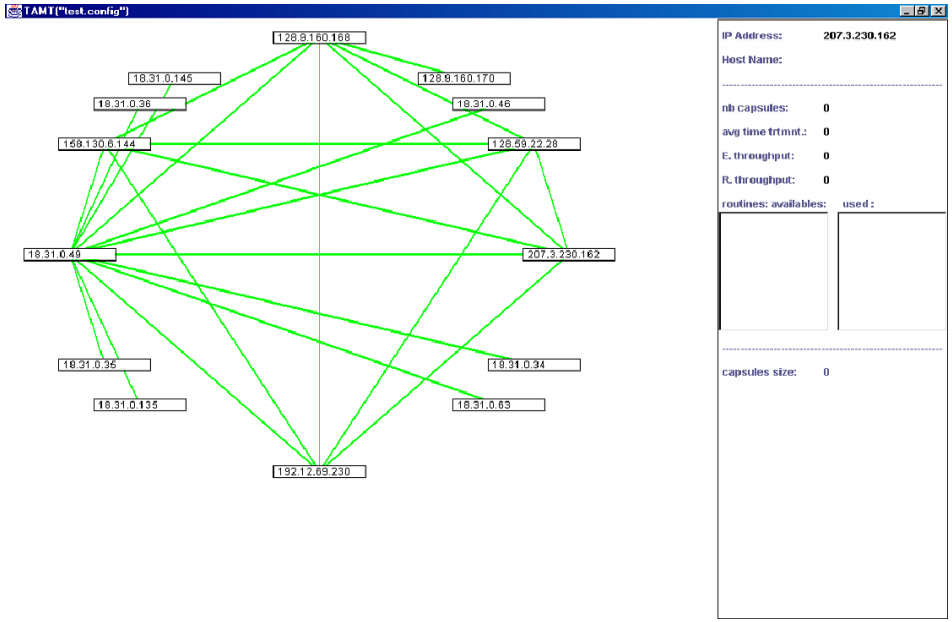
ANM
(Active Node Manager)



- dynamic service loading ;
- dynamic topology : local synchronizations of AN .

TAMT : Tamanoir-Ants Monitoring Tool



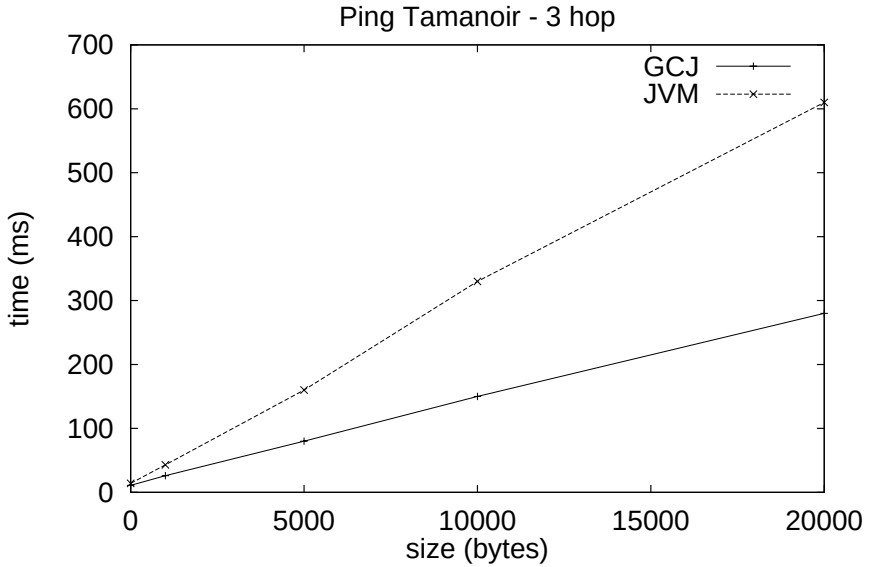


TAMT visualization :

- active architecture : routing topology ;
- active nodes information : address, name ;
- capsule data streams : size, throughput, number of capsules

Experiments

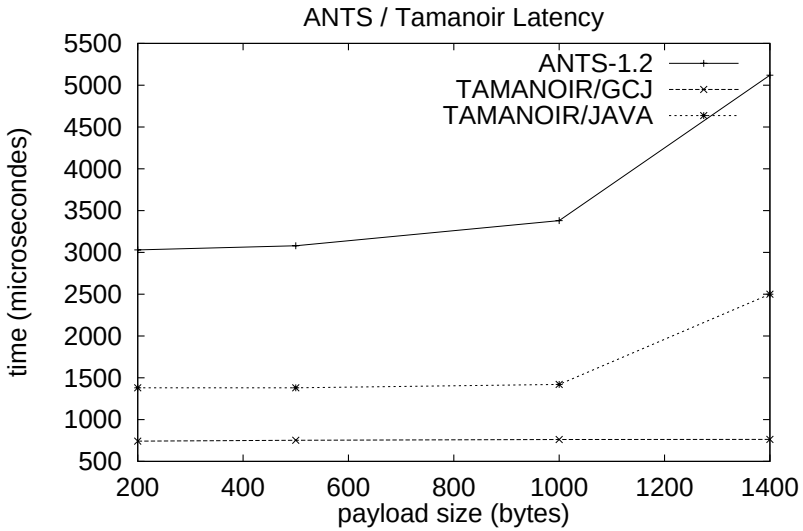
- Ping Application -



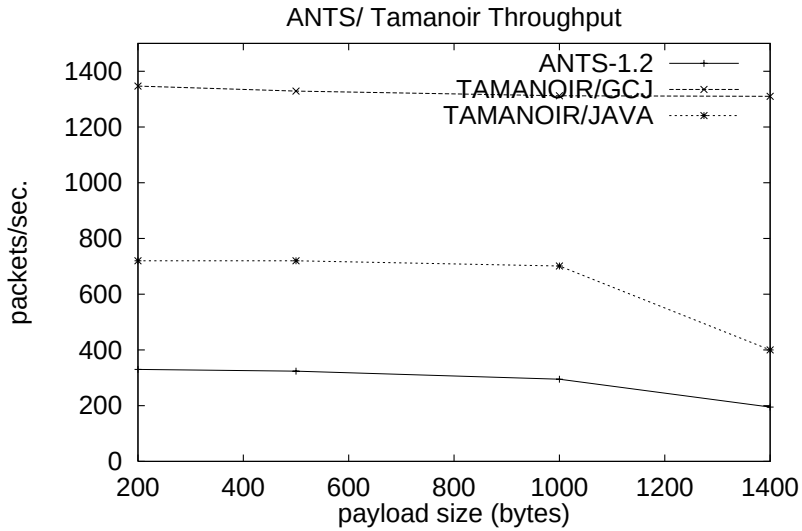
TAMANOIR	JVM	GCJ
latency	1400 μs	754 μs
bandwidth	720 p/s	1324 p/s

Experiments : video stream with ANTS and Tamanoir

Latency



Bandwidth



Qualitative aspects of projects implementations

	ANTS	PAN	TAMANOIR
open source	yes	no	yes
portability	yes (Java)	no (C)	yes (Java+GCJ)
performance aspects	no	yes	yes
security	yes	no	no
dynamic service load	yes	yes	yes
multi-services	no	yes	yes
dynamic topology	no	yes	yes
implementation	3 years	3 years	5 months

Conclusion

- Non all-active model : active nodes outside *core network* ;
- Tamanoir : a complete framework (>3200 JAVA lignes) ;
  - optimized implementation : reduced copies protocols, multithreading...
  - dynamic services deployment ;
  - multi-services and multi-applications support ;
  - dynamic topology.
- Experimental validation ;

Future works

- active node in kernel space ;
- new services :
  - distributed QoS service ;
  - multicast service.
- security ;
- active node on high performance cluster.