QUEUING NETWORKS TO EVALUATE WEB APPLICATIONS: SIMULATION

LEONARDO PASINI AND MORENA BARBONI

Department of Computer Science, University of Camerino
Via Madonna delle Carceri, 62032 Camerino, Italy

(received: 25 October 2018; revised: 3 December 2018;
accepted: 21 December 2018; published online: 9 January 2019)

Abstract: This work fits into the context of our studies on traffic simulation for computer and telecommunication networks [1–5]. The technique that we use is based on complex queuing network models that allow us to simulate the functioning of network devices, and the traffic flows between them. In a previous work [2] we defined a library of objects for the modeling of the computer network. This library also models the network traffic induced by the execution of web applications run in a distributed fashion on various network devices. In this work we describe the changes applied to the service procedures of some library objects, and we develop a specific study for the evaluation of the execution times of three types of applications:

1. Navigation
2. Download Manager
3. Mobile Agents

Keywords: traffic simulators, queuing networks, web application, distributed simulation

DOI: https://doi.org/10.17466/tq2019/23.3/d

1. Introduction

The aim of this work is to evaluate the performance of web applications in a computer network, both in the presence and in the absence of a single sign-on system. Every simulation program defined during the development of this study has been built with `QNAP2` [3], a description and analysis tool for complex queuing networks.

In order to assess the processing times of the tree types of the web application considered in this study, we built two simulators for each architectural paradigm. Therefore, we will describe and analyze in details a total of six simulation cases: Navigation, Download Manager and Mobile Agents, both in the standard and in SSO network configurations.

The main difference introduced by the activation of a single sign-on system lies in the management of the authentication procedures for the client. An

authentication system allows each server to identify the user through his or her credentials. These credentials are verified by performing a database query. If the user is authorized, that is, if he or she has the permission to access the requested data and services, then he or she can continue without being blocked. The authentication procedure for accessing network services can be diversified according to the level of secrecy of the information established by the administrator. For instance, the user can be identified by scanning his or her retina, his or her smart card, or more traditionally by entering a username and a password. The authentication and authorization phases allow filtering users who try to access specific resources. However, they also make operations more burdensome for the system, since every time when a user needs to access a resource, he or she has to go through the entire authentication process.

An identification system called **Single Sign-on** has been devised to overcome this problem. This system reduces the authentication operations to a single step, so that a user can access the network in a very short time. This means that any client within the system will have to be authenticated only once during a session when accessing a server-based application. This initial step authenticates the end user for all the servers to which he or she has been given the rights by the administrator, eliminating any future login procedures. Therefore, a system based on Single Sign-On simplifies the access operations for each web application.

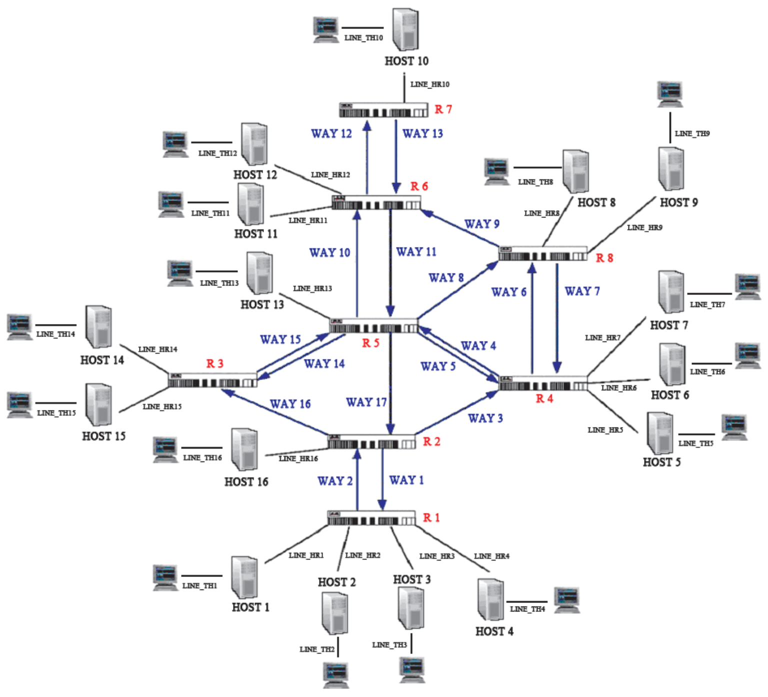


Figure 1. Computer network

We will show how the introduction of a Single Sign-On system reduces execution times by running a simulation for each web application case: Navigation, Download Manager, ad Mobile Agents. Moreover, we will show the improvements introduced by the implementation of a routing system based on the definition of static routes. Figure 1 shows the structure of the reference network, which consists of different types of objects. Since the structure of these objects has already been described in a previous work [2], we are mainly going to focus on the changes applied to the routers and to the communication channels.

Section 2 outlines the difference between the two types of traffic generated by the hosts, and how this traffic impacts the execution times of web applications. Sections 3 and 4 show how network components have been modified in order to better observe the effects of congestion on routers. Section 5 outlines the structure of the three different types of communication channels present in the system, with particular attention paid to the delays introduced by the transmission of data packets. Sections 6 and 7 define the implementation of the routing tables, and the criteria used for the definition of the best path selection strategy. In Section 8 we will describe the functioning of each web application, and the message flow between the devices involved in the simulation. The results of the simulations will be compared and discussed in more details in Section 9.

2. Network traffic

Each host object can generate two different types of traffic.

- **Web Application traffic**: this is the traffic generated by the terminals that take part in the simulation of the web application. The entity of this type of traffic depends on the simulation case considered. In fact, both the size and the quantity of packets vary significantly according to the type of application, and the presence of a Single Sign-On system. The characteristics and the exchange of web application packets are described inside the `server.app` file for each simulation case. If a Single Sign-On server is present, the SSO response packets can be found inside the `auth.app` file.
- **Standard traffic**: the standard traffic is generated by each terminal of the network, regardless of their involvement in the simulation of the web application. These packets have a fixed size, and their generation rate depends on the Tempo RQ global variable defined inside the `model.dat` configuration file. This value is then multiplied by a 100 factor inside the Terminal.User station defined inside the `Object.qnp` file. The Object.qnp file contains the QNAP2 [3] code for the definition of the library objects described in a previous work [2].

Figure 2 shows the structure of the Terminal.User queue. This queue receives packets from the overlying terminal, and sends them to subsequent queues in order to simulate the TCP protocol.

The standard traffic is in no way related to the web application traffic, but it is necessary to make the simulation as realistic as possible. For this reason, the generation rate of the standard traffic has been set to 1 packet per

```
/STATION/
NAME = *TERMINAL.USER;
TYPE = SOURCE;
SERVICE = BEGIN
IF (ACTIVES) THEN BEGIN
  EXP(TEMPO*100);
  RQ := NEW(REQUEST);
  RQ.ORIG:=IDT;
  RQ.SIZE:= REQLENGHT;
  DEST := RINT(1,NHTEST);
  WHILE (TERM#(DEST).ACTIVES=FALSE) DO
    DEST := RINT(1,NHTEST);
    RQ.DESTI := DEST;
    LUNGH := RQ.SIZE;
    TRANSIT(RQ,CPU,EMISSION);
    END
ELSE
  CST(T_MAX);
END;
TRANSIT = OUT;
```

Figure 2. Definition of the User queue within the Terminal object

second regardless of the simulation case considered. This allows us to analyze the execution of the web applications and the behavior of the network devices in a more realistic scenario. The congestion levels inside the network increase due to the higher traffic generation rate. The web application packets take longer to reach their destination, and in some cases the acknowledgement cannot be returned before the timeout expiration. As a consequence, the hosts will try to re-transmit the packets, which further contributes to the system congestion. For this reason, the introduction of an optimal routing system is necessary to handle the execution of applications in high traffic conditions.

3. Error Probability

As has been mentioned in Section 2, increasing the standard traffic generation rate results in the impossibility of terminating web applications. The higher number of data packets congests the system to such an extent as to cause significant delays in the delivery of web application packets. As a consequence, the sender receives the acknowledgements after the timeout expiration. Since the delay is interpreted as a loss, the host re-transmits the packet contributing to the system congestion. Moreover, increased traffic results in a higher number of packets that may be lost. The function that determines whether a packet will be lost or not is defined inside the routers. This function allows us to simulate possible transmission errors that prevent packets of data from reaching their destination. In particular the probability of a packet being lost is applied inside the Cpu queue.

As we can see from Figure 3, each packet is initially processed according to a service time relative to the packet type, which can be standard traffic, acknowledgement, or web application traffic. After the initial processing, the probability of error for the data packet is calculated by means of the Boolean

```
/STATION/
NAME = *ROUTER.CPU;
SCHED = FIFO,PRIOR;
SERVICE = BEGIN
IF (TYP = "info") THEN
  EXP((HOSTS#(SOUR_ID).LI * 8)/RATE)
ELSE IF (TYP = "app" ) AND (CUSTOMER IS WEBAPP) THEN
  EXP(((CUSTOMER::WEBAPP).N * 8)/RATE)
ELSE EXP((HOSTS#(SOUR_ID).LA * 8)/RATE);
IF DRAW(P_ERR) THEN TRANSIT(OUT) ELSE
IF (HOSTS#(DEST_ID).ID_R = ID) THEN
IF (CCLASS=WEBAP) THEN TRANSIT(S(DEST_ID))
ELSE TRANSIT(S(DEST_ID),RECEPT);
END;
TRANSIT=INSTR(1 STEP 1 UNTIL N),PROB,OUT;
```

Figure 3. Definition of the Cpu queue within the Router object

function DRAW (P_ERR). This function is applied to every packet, regardless of their type. The value of the error probability is specified inside the `model.dat` file.

```
& ROUTERS
&   ID   Interface          Rate   P_Err
     1           1         100.0   0.000;
     2           3         123.0   0.000;
     3           1         111.0   0.000;
     4           2         122.0   0.000;
     5           5         100.0   0.000;
     6           2         123.0   0.000;
     7           1         111.0   0.000;
     8           2          50.0   0.000;
```

Figure 4. Router data definition inside the Model.dat file

Since we have increased the standard traffic generation rate, each router's Cpu will process a higher amount of data. The total number of packets lost during the simulation will therefore be greater. For this reason, the error probability has been set to 0, regardless of the simulation case considered. This will prevent packets from being lost, eliminating the need for re-transmissions inside the network. However, the expiration of the sender's timeout can still occur due to delays in the delivery of the packets. In the next section we are going to discuss this problem in more detail.

4. Timeout

Even though the re-transmissions of packets have been considerably reduced, they are still present. Any acknowledgement that fails to reach its destination before the timeout expiration will cause the sender to re-transmit the packet. In fact, it is not possible for the host to distinguish between a delay caused by excessive congestion, and the loss of a packet. However, since packet loss can no longer occur, any packet that is experiencing delay will eventually reach its destination.

We can therefore remove the timeout to prevent the hosts from re-transmitting packets that would cause an increase in congestion levels. In our model the timeout value is specified by the *Wtout* parameter. To prevent the timeout from expiring, it is necessary to modify all the occurrences of Wtout inside the queues that handle the web application packets.

```
N:=GET(FA,INTEGER);
IF (N=0) THEN PRINT ("ERROR: Data is not consistent");
BCPU:=GET(FA,INTEGER);
BD:=GET(FA,INTEGER);
WTOUT:= 1E+07;
NewLn(FA);
OP:=1;
TRANSIT (TERM#(EXIT).SCH);
```

Figure 5. Defintion of the Wtout parameter inside the Program queue

Figure 5 shows a fragment of the code for the definition of the Program queue inside the Software object. In this case the Wtout parameter has been set to 10^7, so that the timeout will never expire before the end of the simulation.

5. Communication Channels

This section aims to show the structure of the communication channels within the network, and the definition of the transmission delays applied to the data packets. Our model defines three channel types: *Terminal-Host lines*, *Host-Router lines*, and *Ways*. Each channel type connects different network devices, and is characterized by a specific transmission rate and direction of communication.

5.1. Terminal-Host Line

The TH line represents the full-duplex communication channel between each host and its terminal. This object consists of two queues:

- **QH:** forwards data from the terminal to the host;
- **HQ:** forwards data from the host to the terminal;

Figure 7 shows the structure of the QH and HQ queues. Each queue defines a service time to simulate the amount of time required to push all the data packets over the link. This transmission delay depends on the value TH, and it is applied both to web application and standard traffic packets.

5.2. Host-Router Line

This object represents the communication link between a host and a router; the connection is half-duplex, as it only transmits the outgoing packets from the host to the router. The incoming packets are sent by the router CPU directly into the host's Access_R queue.

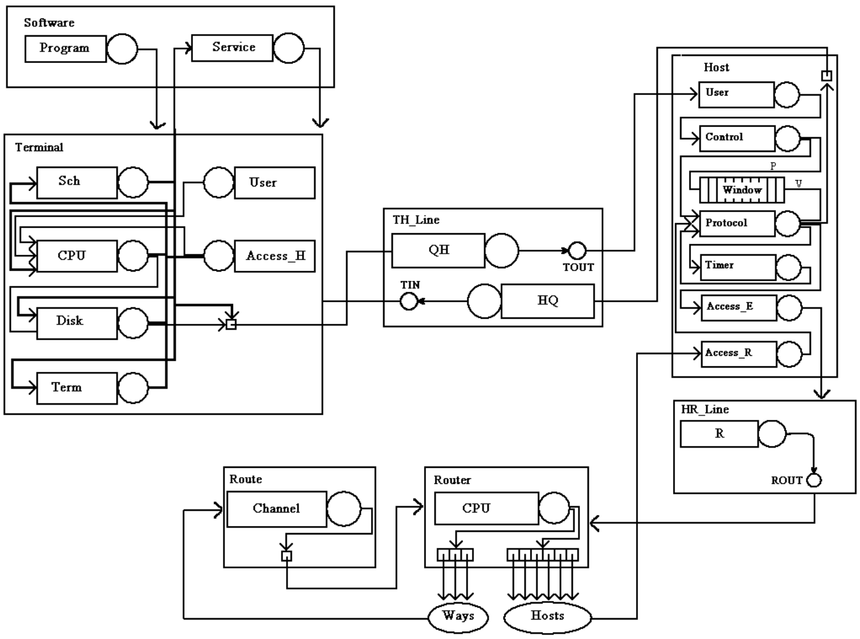


Figure 6. Connections between network components

```
/STATION/
NAME = *LINE_TH.QH;
SERVICE = CST(TH);
TRANSIT = TOUT;

/STATION/
NAME = *LINE_TH.HQ;
SERVICE = CST(TH);
TRANSIT = TIN;
```

Figure 7. TH Line queues

```
/STATION/
NAME = *LINE_HR.R;
TYPE = INFINITE;
SERVICE = CST(T);
TRANSIT = ROUT;
```

Figure 8. HR Line Station

The HR line object consists of a single queue R, which simulates the transmission delay for the data packets, as well as a pointer to the destination router.

5.3. Way

The way object represents the communication link between two routers; it is a half-duplex connection, as it is capable of sending information in one direction only.

```
OBJECT WAY(ID,TEM);
QUEUE CHANNEL;
REF QUEUE NEXT_R;
REAL TEM;
INTEGER ID;
END;
```

Figure 9. Way Object

The structure of the way object consists of a single queue that simulates the transmission delay, and a pointer to the CPU of the destination router. The ID constant uniquely identifies each way within the network.

```
/STATION/
NAME=*WAY.CHANNEL;
TYPE = SERVER, INFINITE;

SERVICE= BEGIN
IF (TYP = "info") THEN
EXP((HOSTS#(SOUR_ID).LI * 8)/200.)
ELSE EXP((HOSTS#(SOUR_ID).LA * 8)/200.);
END;

SERVICE(WEBAP)= BEGIN
IF (TYP = "app" )
THEN EXP(((CUSTOMER::WEBAPP).N *8)/200.)
ELSE EXP((HOSTS#(SOUR_ID).LA * 8)/200.);
END;

TRANSIT=NEXT_R
```

Figure 10. Definition code for the Channel station inside the Way object

The Way.Channel station (Figure 10) calculates the transmission delay for each packet through its service time procedure. However, this station features two different procedures that no longer depend on the constant service time value. This means that data packets belonging to different classes will not need the same amount of time to reach their destination. This makes the simulation more realistic and coherent, since the transmission delay will vary according to the size of each packet.

It is important to note that, in order to observe how the traffic congestion affects the routers, each channel has been defined as an **infinite servants station**. A queue represents a waiting line for a service; whenever customers (or data packets) reach the station, they queue to get the service from a set of resources or servants. After the execution of the service the packets leave the system, or are rerouted towards the next station. The specific service can be provided by a single servant, or by multiple servants in parallel.

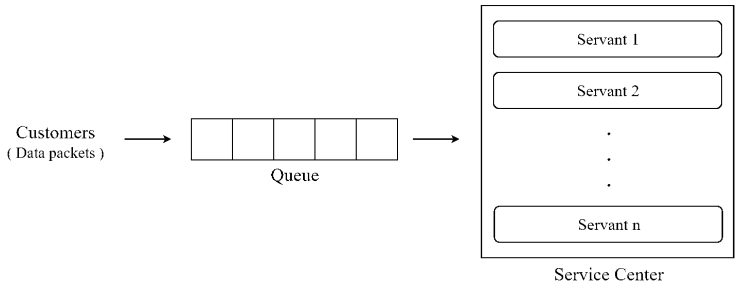


Figure 11. Model of a multiple servant station

In the case of a single servant station, as soon as a customer arrives at the service center it has to wait for the servant to become available. In fact, the servant can only provide a service to an incoming customer after completing a previous request. If the channels are defined as single servant stations, the ways become bottlenecks for the entire network.

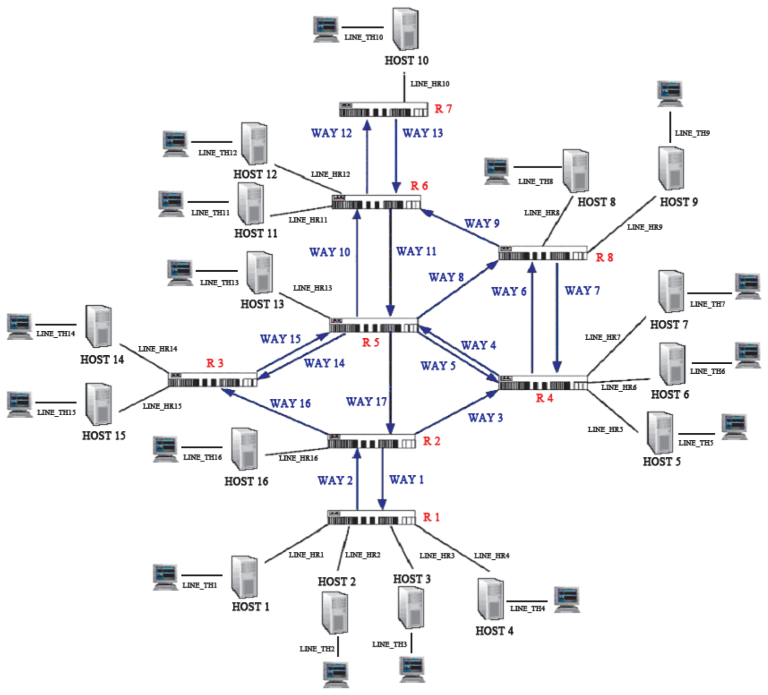
An infinite servant station, or a delay center, has the same structure of a multiple servant station (Figure 11). However, the former consists of an infinite number of resources. If an infinite servant station is implemented, every time a customer arrives at the station it will always find an available servant. As a result, the packets will not experience any delay caused by the presence of the waiting time.

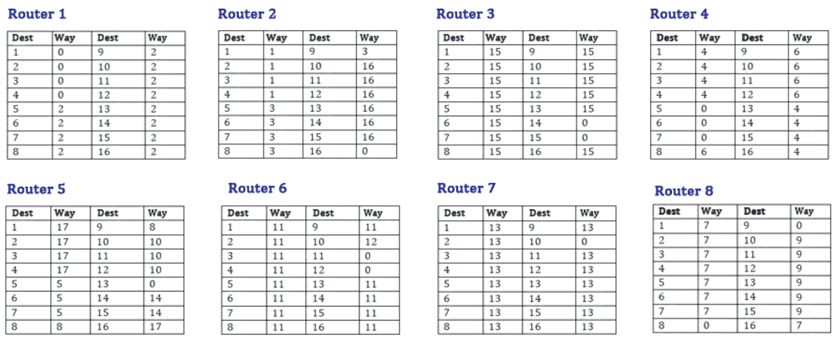
The introduction of delay centers eliminates congestion in the correspondence of the ways because packets are immediately served and accumulate inside the channel queue no longer. Since the ways are able to serve a higher number of customers, the effect of congestion in the correspondence of routers becomes much more evident. This allows us to identify the most critical points of the network, that is, the routers that show the highest levels of congestion. This information is essential to define an efficient routing algorithm that takes into account the traffic load balancing.

6. Routing

This section aims to show how the introduction of routing tables can significantly improve the simulation time of applications.

A **routing table** (Figure 13) is a data structure stored inside a router that defines the path to a specific network destination.





Router 1

Dest	Way	Dest	Way
1	0	9	2
2	0	10	2
3	0	11	2
4	0	12	2
5	2	13	2
6	2	14	2
7	2	15	2
8	2	16	2

Router 2

Dest	Way	Dest	Way
1	1	9	3
2	1	10	16
3	1	11	16
4	1	12	16
5	3	13	16
6	3	14	16
7	3	15	16
8	3	16	0

Router 3

Dest	Way	Dest	Way
1	15	9	15
2	15	10	15
3	15	11	15
4	15	12	15
5	15	13	15
6	15	14	0
7	15	15	0
8	15	16	15

Router 4

Dest	Way	Dest	Way
1	4	9	6
2	4	10	6
3	4	11	6
4	4	12	6
5	0	13	4
6	0	14	4
7	0	15	4
8	6	16	4

Router 5

Dest	Way	Dest	Way
1	17	9	8
2	17	10	10
3	17	11	10
4	17	12	10
5	5	13	0
6	5	14	14
7	5	15	14
8	8	16	17

Router 6

Dest	Way	Dest	Way
1	11	9	11
2	11	10	12
3	11	11	0
4	11	12	0
5	11	13	11
6	11	14	11
7	11	15	11
8	11	16	11

Router 7

Dest	Way	Dest	Way
1	13	9	13
2	13	10	0
3	13	11	13
4	13	12	13
5	13	13	13
6	13	14	13
7	13	15	13
8	13	16	13

Router 8

Dest	Way	Dest	Way
1	7	9	0
2	7	10	9
3	7	11	9
4	7	12	9
5	7	13	9
6	7	14	9
7	7	15	9
8	0	16	7

Figure 12. System network with routing tables

Each router has the task of forwarding data packets between hosts. When a packet reaches a router, its address information is used to search for the output interface inside the routing table. Thus, we can see a routing table as a set of rules, where each rule determines the next hop for the packet according to its destination.

Destination	Next_Hop
Host_1	Int_2
Host_2	Int_1
Host_3	Int_3

Figure 13. Routing table

Since our network topology is fixed, there is no need to define discovery procedures to update the content of the tables. The definition of static entries will be sufficient to guarantee a consistent routing between hosts. Figure 12 shows the implementation of the routing tables for each component of the network. The introduction of a static routing algorithm brings the following benefits:

- Each packet is always sent along the shortest path towards its destination host. This means that any unnecessary delay will be avoided.
- Since each packet hops from a device to another according to a deterministic algorithm, routers will not experience congestion caused by routing loops. Each packet will never perform more hops than necessary, and will never be processed multiple times by the same router.

The following subsections will show how the routing system has been implemented. In particular, it is necessary to apply changes to the following files: `Model.dat`, `Simulatore.qnp` and `Objects.qnp`.

6.1. Objects.qnp

The `Objects.qnp` file defines the model's object library. The file contains the definition of all the objects and procedures of the system. Figure 14 shows how the router object has been defined.

```
OBJECT ROUTER(ID,N);
INTEGER N;
INTEGER ID;
QUEUE CPU;
REF QUEUE S(NH);
INTEGER INSTR(16);
INTEGER PKTDEST(16);
REAL RATE;
REAL P_ERR;
END;
```

Figure 14. Router object

Each router object contains a queue that models the CPU, and two arrays of integers, Instr and PktDest, which represent the routing tables.

- **PktDest:** contains all the possible destinations of a packet;
- **Instr:** each element of the array contains the channel towards which a packet must be forwarded.

Since there is a total of 16 hosts in the system, both arrays have been declared with a length of 16. The actual routing algorithm has been defined inside the Cpu station of the Router object.

The Router.Cpu station handles incoming packets with a FIFO scheduling algorithm; the packets are queued and processed according to their order of arrival. However, all the packets belonging to the web application class have been assigned a higher priority with respect to the standard traffic. This is achieved through the PRIOR(WEBAP) function. The definition of a scheduling based on priority will

```
/STATION/
 NAME = *ROUTER.CPU;
 SCHED = FIFO,PRIOR;
 PRIOR(WEBAP)=2;
 SERVICE = BEGIN

    IF (HOSTS#(DEST_ID).ID_R = ID) THEN
        IF (CCLASS=WEBAP) THEN
           TRANSIT(S(DEST_ID))
        ELSE
           TRANSIT(S(DEST_ID),RECEPT);

    FOR J:=1 STEP 1 UNTIL 16 DO BEGIN
     IF (DEST_ID = PKTDEST(J)) THEN
      IF (INSTR(J)<>0) THEN
       TRANSIT(WAY#(INSTR(J)).CHANNEL);
    END;
 END;
```

Figure 15. Definition code for the Cpu station inside the Router object

ensure that web application packets will be served before any other packet type, optimizing the execution time of the simulation.

The router applies the routing algorithm to every packet, regardless of their class. At first the Cpu station handles the routing of the packets that have already reached the destination sub-net. These packets are directly forwarded towards the final host. If a packet is addressed to a host belonging to a different sub-net, the routing table is analyzed to determine its next hop. As we have noted earlier, the routing table is stored inside the router by means of the two arrays INSTR and PKTDEST. The algorithm goes through the table in the following way: If the destination of the current packet equals the host Id contained in the ith position of the PKTDEST array, then the packet is forwarded towards the channel queue of the way defined inside the ith position of the INSTR array.

6.2. Simulatore.qnp

The `Simulatore.qnp` file builds the system simulator by means of a `BuildMod` procedure that generates the model data from the `model.dat` file. The BuildMod procedure also contains the code that reads and initializes all the routing parameters.

```
FOR I:=1 STEP 1 UNTIL N_Router DO BEGIN
  idr:=GET(INTEGER);
  WITH ROUTER#(idr) DO BEGIN
      FOR J:=1 STEP 1 UNTIL 16 DO BEGIN
      INSTR(J):=GET(INTEGER);
      PKTDEST(J):=GET(INTEGER);
      END;
  END;
 END;
```

Figure 16. Initialization of router parameters inside the BuildMod procedure

Figure 16 shows how the BuildMod procedure initializes each router with its routing table. In particular, the routing parameters are stored inside the Instr and PktDest array that we already introduced in the previous subsection.

6.3. Model.dat

The `model.dat` file defines all the components of the model. It contains all the global variables needed to build the structure of the network, such as the number of hosts and the connections between them. This is where the routing tables have been declared too.

```
& ROUTING TABLES
  &ID    Way Dest  Way Dest  Way Dest  Way Dest  Way Dest  Way Dest  Way Dest  Way Dest
  1       0   1     0   2     0   3     0   4     2   5     2   6     2   7     2   8
          2   9     2  10     2  11     2  12     2  13     2  14     2  15     2  16;
  2       1   1     1   2     1   3     1   4     3   5     3   6     3   7     3   8
          3   9    16  10    16  11    16  12    16  13    16  14    16  15     0  16;
  3      15   1    15   2    15   3    15   4    15   5    15   6    15   7    15   8
         15   9    15  10    15  11    15  12    15  13     0  14     0  15    15  16;
  4       4   1     4   2     4   3     4   4     0   5     0   6     0   7     6   8
          6   9     6  10     6  11     6  12     4  13     4  14     4  15     4  16;
  5      17   1    17   2    17   3    17   4     5   5     5   6     5   7     8   8
          8   9    10  10    10  11    10  12     0  13    14  14    14  15    17  16;
  6      11   1    11   2    11   3    11   4    11   5    11   6    11   7    11   8
         11   9    12  10     0  11     0  12    11  13    11  14    11  15    11  16;
  7      13   1    13   2    13   3    13   4    13   5    13   6    13   7    13   8
         13   9     0  10    13  11    13  12    13  13    13  14    13  15    13  16;
  8       7   1     7   2     7   3     7   4     7   5     7   6     7   7     0   8
          0   9     9  10     9  11     9  12     9  13     9  14     9  15     7  16;
```

Figure 17. Definition of routing tables inside the Model.dat file

Figure 17 shows the structure of the routing tables. Each entry contains all the information needed to forward a packet towards its final destination. The routing parameters are:

- **ID:** Identifies the router;
- **Dest:** Represents the final destination of the packet. Each destination corresponds to the ID of a single host within the network. Based on this information the router will determine the exit interface for the packet, which points to a unique communication channel.
- **Way:** Represents the way through which the packet will be transmitted. Every way is connected to a destination router that represents the next hop for the packet. If the way parameter is equal to 0, it means that the current router is located in the same sub-net of the destination host. In this case the packet will be directly forwarded to the final host.

In order to guarantee the implementation of an efficient routing algorithm, it is necessary to identify the best path in the function of each router and destination. This aspect will be discussed in details in Section 7.

7. Routing paths

Each router object contains a different set of rules in the form of routing tables. However, all the tables have been built according to the same criteria: A packet must always follow the shortest path to reach its destination. That is, the one that includes the smallest number of hops to the target sub-net. If all

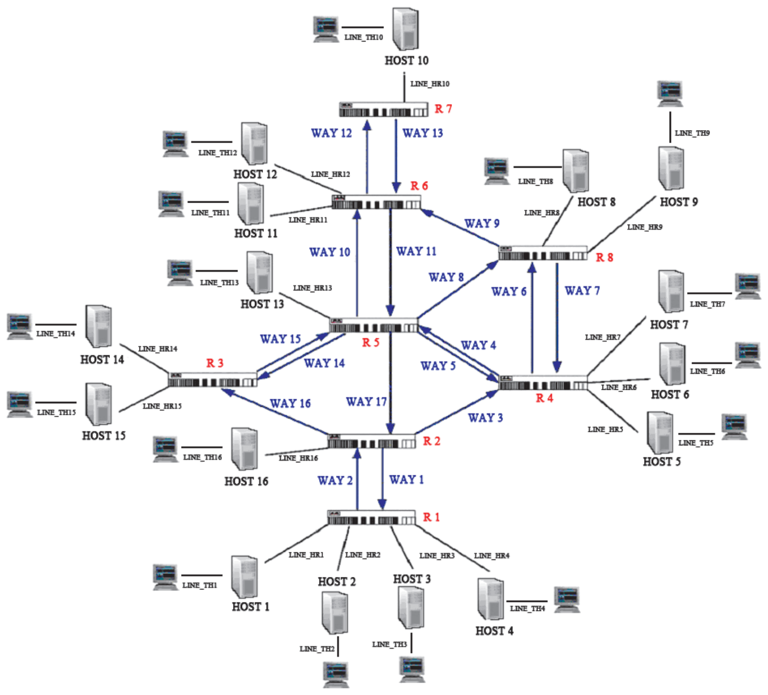


Figure 18. Reference network

the paths have equal length, then the levels of congestion of the routers along the path are considered in order to make the decision.

According to the results of the simulations, Routers 3, 5, 6 and 8 are those that process the highest number of web application packets. This is because they are directly connected to the hosts involved in the execution of web application requests and replies. Let us then discuss how the rules have been defined in the presence of equal length alternatives:

- **Router 8:** This router can forward all the packets intended for Hosts 1 to 4 (and 13 to 16), to Channel 7 or 9. Both choices will require the same amount of hops for the packets to reach their destination. In this situation it was decided to select the following routes: each packet addressed to Hosts 1 to 4, and to Host 16, is forwarded to Channel 7. On the other hand, traffic addressed to Hosts 13 to 15 is sent over Channel 9. This is the most efficient way of routing the packets, because Channel 7 is already congested by the web application packets exchanged between Client 8, and Servers 11 and 12. By separating the two web application flows we can distribute the traffic in a more uniform way, reducing the congestion levels on the router.
- **Router 4:** This router presents a similar situation because the packets addressed to Hosts 10 to 12 can be sent indifferently over Channel 4 or 6. In this case picking Channel 6 is the best choice. In fact, Channel 4 leads to Router 5 which is the central point of the network and therefore one of the most congested.

- **Router 2:** In this case the packets addressed to Host 10–13 can be either sent over Channel 3, or over Channel 16. According to our simulation results, Router 3 presents lower congestion levels than Router 4. Therefore, we select Channel 16 to achieve load balancing.

8. Simulation cases

This paragraph will focus on the analysis of individual simulation cases, and on the benefits introduced by the implementation of the new routing system. We must define the roles performed by each host during the simulation to describe the flow of messages between the network devices. Therefore, we establish that:

- Host 8 will act as client with Role 1.
- Host 11 will act as the Single Sign-On server with Role 3;
- All the other roles will be performed by various servers, which will reside on Hosts 12, 14 and 15.

Having made these assumptions, we can now begin to define the case studies. For each case we will describe the general behavior of the web application, the exchange of packets between network components, and finally we will show an extract of the relative simulation trace.

8.1. Navigation

The standard navigation case simulates a user's browsing activity in the absence of a single sign-on system.

The communication between the hosts takes place as follows: the client sends an initial request to Server 12. This request, which is 5KB in size, contains all the information needed to authenticate the user. Once Server 12 has received this message, it can send a cookie back to the client. This cookie is 35KB in size as it contains all the profiling information of the user. At this point the client can send a new request to the server. However, this second request also contains the identifier of the cookie for 10 KB in total size. The server can now respond to the initial client's message by returning the requested web page. In our example the requested page is 12KB in size. After receiving the server's response, we suppose that the client tries to access another web page hosted on Server 13. However, since no Single Sign-On system has been implemented, all the authentication steps must be repeated each time that the client requests a page from a new server. After the second authentication phase the client obtains the 16KB page from Server 13, and the web application terminates.

Figure 20 shows the definition of each web application packet transmitted during the navigation case. The `Src` and `Dest` columns contain the source and destination host of each packet. The remaining columns define different types of time delays applied during the transmission and reception phases. These consist of the CPU, disk and terminal elaboration times, both for the client (A) and server (B) side. The -1 value defined inside the Acpu column represents a special terminator that allows us to end the simulation. The execution of this application

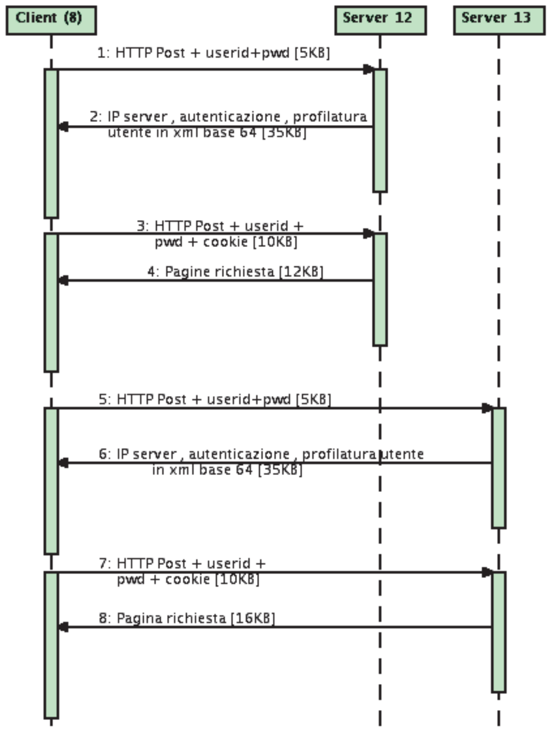

Client (8)
Server 12
Server 13
1: HTTP Post + userid+pwd [5KB]
2: IP server , autenticazione , profilatura utente in xml base 64 [35KB]
3: HTTP Post + userid + pwd + cookie [10KB]
4: Pagine richiesta [12KB]
5: HTTP Post + userid+pwd [5KB]
6: IP server , autenticazione , profilatura utente in xml base 64 [35KB]
7: HTTP Post + userid + pwd + cookie [10KB]
8: Pagina richiesta [16KB]

Figure 19. Sequence Diagram: Navigation

```
& Src  Dest    Acpu  Adisk   Aterm   Net     Bcpu  Bdisk
    1     2     100   100     500    5000    1500  1500;
    2     1     100   50      200    35000   1000  500;
    1     2     100   100     300    10000   400   100;
    2     1     100   100     300    12000   400   150;
    1     3     100   100     500    5000    1500  1500;
    3     1     100   50      200    35000   1000  500;
    1     3     100   100     300    10000   400   100;
    3     1     100   100     300    16000   400   150;
    0     0      -1;                && Terminated
```

Figure 20. Navigation – server.app file

takes approximately 44 seconds. For the sake of simplicity, we only report the final steps of the simulation trace.

Figure 21 shows the transmission of the last packet which contains the 16KB page requested by Client 8. This packet has been emitted by Server 13 at 34 seconds from the beginning of the simulation. Router 5 then proceeds to forward the data towards Way 8, which is directly connected to Router 8. Since the latter is part of the destination sub-net, the packet is finally delivered to Host 8.

```
*********************************
        PACKET SENT:
*********************************
- DIMENSION:     16000
- TYPE: risweb
- SOURID:        13
- SOURNP:         3
- DESTID:         8
- DESTNP:         1
- SEQNUM:         4
- WTOUT:  10000000
- TIME:   0.3490E+05
*********************************
------------ ROUTING ------------
--->  ROUTER: 5     --->  WAY:    8
-----------------------------------
SOUR:   13              DEST:    8
DIM:     16000
TIME:   0.3682E+05
-----------------------------------

********  PACKET RECEIVED  ********
DIM.RECEIVED:    16000
- SOUR:        13 - DEST:        8
TIME:   0.4252E+05
************************************

10 0    0  -1; && Terminated
APPLICATION TERMINATED AT:  0.4417E+05
```

Figure 21. Navigation simulation trace

8.2. Single sign-on navigation

The implementation of a single sign-on system implies the definition of a different network topology. In fact, we need to introduce an additional server that will provide access to all the system's resources once the user is authenticated.

As we have previously noted, the role of the SSO server will be performed by Host 11.

The communication takes place in the following way: the client sends the initial authentication request to Server 12. Server 12 forwards the request to the SSO server, which handles the authentication procedure and returns the 35KB cookie to the client. At this point the client is able to obtain any web page by adding the cookie identifier to its request, for a total size of 1KB. The authentication provided by the SSO server will be valid for every other server within the network. Therefore, there is no need to repeat all the authentication steps each time the client wants to access a page hosted by a different server. The benefits of introducing SSO servers are evident: Both the number of packet exchanges, and the size of the requests have been significantly reduced.

Figure 23 shows the definition of the web application packets exchanged during the simulation. It is important to note that, in order to compare the two

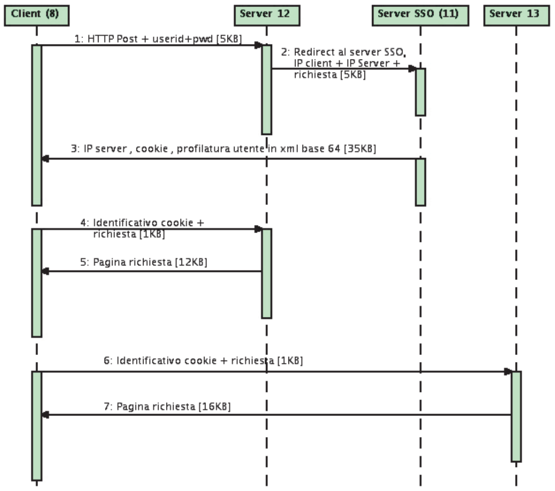

Client (8)
Server 12
Server SSO (11)
Server 13
1: HTTP Post + userid+pwd [5KB]
2: Redirect al server SSO, IP client + IP Server + richiesta [5KB]
3: IP server , cookie , profilatura utente in xml base 64 [35KB]
4: Identificativo cookie + richiesta [1KB]
5: Pagina richiesta [12KB]
6: Identificativo cookie + richiesta [1KB]
7: Pagina richiesta [16KB]

Figure 22. Sequence Diagram: SSO Navigation

```
& Src  Dest     Acpu  Adisk  Aterm  Net    Bcpu   Bdisk
    1     2     100   100    500    5000   1500   1500;
    2     3     100   50     200    5000   1000   500;
    1     2     100   100    300    1000   400    100;
    2     1     100   100    300    12000  400    150;
    1     4     100   100    500    1000   1500   1500;
    4     1     100   50     200    16000  1000   500;
    0     0     -1;              && Terminated

& SSO Server
& Src  Dest     Acpu  Adisk    Aterm  Net      Bcpu    Bdisk
   3    1       100   50     200    35000  2500   2500;
   0    0       -1; && Terminated
```

Figure 23. SSO Navigation – server.app and auth.app files

simulation cases, both the size and the number of web page requests have been defined to reflect the message flow of the standard navigation case.

The simulation of the application (Figure 24) now terminates at about 30 seconds. The routing of the last request does not change, the packet follows the same path of the previous simulation. However, the 16KB page requested by Client 8 has been emitted 10 seconds earlier than the standard navigation case. As expected, the reduction in the number of requests improves the simulation time. Although the benefits are already visible for two client requests, it is clear that the difference between the two navigation cases would be even more evident for a higher number of client-server interactions.

8.3. Download Manager

Let us now examine the structure of the download manager application. In this case, the client wants to download a file that resides on different file servers. Each server stores a part of the file, and the client can only use the resource once

```
*********************************
        PACKET SENT:
*********************************
- DIMENSION:     16000
- TYPE: risweb
- SOURID:        13
- SOURNP:         4
- DESTID:         8
- DESTNP:         1
- SEQNUM:         7
- WTOUT:  10000000
- TIME:   0.2439E+05
*********************************
------------- ROUTING -------------
--->  ROUTER: 5     --->  WAY:    8
SOUR:        13      DEST:       8
DIM:     16000
TIME:   0.2474E+05
-----------------------------------
********  PACKET RECEIVED  ********
DIM.RECEIVED:    16000
- SOUR:        13 - DEST:        8
TIME:   0.3064E+05
************************************

9   0     0     -1; && Terminated
APPLICATION TERMINATED AT:   0.3076E+05
```

Figure 24. SSO Navigation simulation trace

it has obtained all the parts from different servers. The role of the file servers will be performed by Hosts 12, 13, 14 and 15, while the client will reside on Host 8 as usual.

The application starts with the client sending a request to Server 12. Once again, this request contains all the information needed for the authentication process. After authenticating the client, Server 12 sends the authentication information to all the other servers involved in the file request. It is important to note that these messages are asynchronous, as Server 12 does not wait for any type of response from the other servers. Once the authentication process is over, each file server sends its portion of the requested file back to the client. In our example, the file is composed by equal size parts of 720KB each.

We can see from the application file (Figure 26) that the type of data processed during this simulation is very different from the navigation case. The files downloaded by the client have a much larger size, and the user has to be authenticated on four different servers.

The last steps of the simulation trace (Figure 27) show that Client 8 receives in succession files from Servers 13, 14, 15 and 12. The file hosted on Server 12 is the last transmitted file. In fact, Server 12 must forward the authentication information to all the other file servers before sending its own file to the client. The simulation of the download manager application ends at 667 seconds. The increase in the simulation time is caused by the download of the 720KB file parts, which

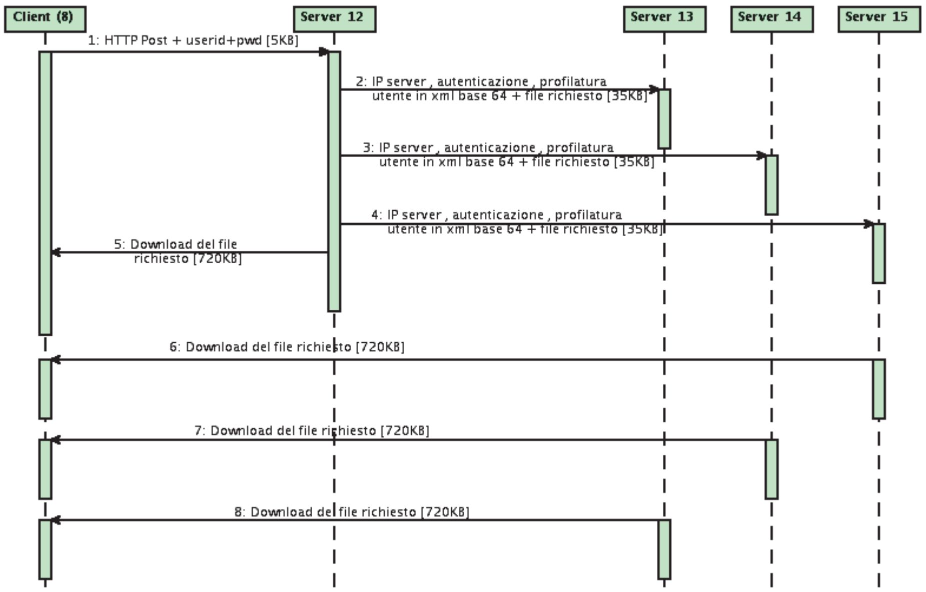

Client (8)
Server 12
Server 13
Server 14
Server 15
1: HTTP Post + userid+pwd [5KB]
2: IP server , autenticazione , profilatura utente in xml base 64 + file richiesto [35KB]
3: IP server , autenticazione , profilatura utente in xml base 64 + file richiesto [35KB]
4: IP server , autenticazione , profilatura utente in xml base 64 + file richiesto [35KB]
5: Download del file richiesto [720KB]
6: Download del file richiesto [720KB]
7: Download del file richiesto [720KB]
8: Download del file richiesto [720KB]

Figure 25. Sequence Diagram: Download Manager

```
& Src Dest Acpu  Adisk  Aterm  Net     Bcpu   Bdisk
  1    2    100   100    500    5000    1500   1500;
  2    3    1500  1000   50     35000   1000   1000;
  2    4    1500  1000   50     35000   1000   1000;
  2    5    1500  1000   50     35000   1000   1000;
  2    1    100   100    100    720000  100    100;
  3    1    100   100    100    720000  100    100;
  4    1    100   100    100    720000  100    100;
  5    1    100   100    100    720000  100    100;
  0    0    -1; && Terminated
```

Figure 26. Download Manager – server.app file

take longer to be processed and transmitted by the network devices. However, the large size of these files is not the only factor that negatively impacts the network performance. In fact, the transmission of the authentication data for each file server contributes to the extension of the simulation time. In the next subsection we are going to show how the introduction of a Single Sign-On system can partially solve this problem.

8.4. Single sign-on Download Manager

We need to introduce a Single Sign-On server in our network topology to implement a Single Sign-On system. As usual, the SSO server will reside on Host 11. The operations performed by the download manager application in the presence of an SSO system are similar to those described for the navigation case.

Figure 28 shows the interaction between the hosts involved in the simulation of the application. The client requests a specific resource from Server 12. The client

```
******** PACKET RECEIVED ***********
DIM.RECEIVED:   720000
- SOUR:         13 - DEST:         8
TIME:   0.2177E+06
************************************
******** PACKET RECEIVED ***********
DIM.RECEIVED:   720000
- SOUR:         14 - DEST:         8
TIME:   0.2282E+06
************************************
******** PACKET RECEIVED ***********
DIM.RECEIVED:   720000
- SOUR:         15 - DEST:         8
TIME:   0.2297E+06
************************************
------------- ROUTING -------------
--->  ROUTER: 5     --->  WAY:    8
SOUR:         12       DEST:       8
DIM:     720000
TIME:   0.3444E+06
-----------------------------------
******** PACKET RECEIVED ***********
DIM.RECEIVED:   720000
- SOUR:         12 - DEST:         8
TIME:   0.6676E+06
************************************

11 -1; && Terminated
APPLICATION TERMINATED AT:  0.6678E+06
```

Figure 27. Download Manager simulation trace

needs to be authenticated to access this resource. The client's request is then redirected to the SSO server which, after the authentication operations, returns the 35KB cookie. The client can now use the cookie ID to repeat its request to Server 12, for a total size of 1KB. Server 12 forwards this packet to all the servers that store parts of the requested file. Each one of these servers recognizes the client as authenticated, and transmits its part of the file back to the client.

From the application file (Figure 29) it is evident that the introduction of the SSO system entails a variation in terms of the quantity and size of packets. In fact, Server 12 sends much smaller packets to the other file hosting servers in comparison to the previous case. This is due to the fact that the client has already been authenticated by the SSO server, and it is therefore sufficient to include the cookie ID inside the client's request.

Figure 30 shows the final steps of the simulation trace. We can observe how the large size of the files still slows down the execution of the application. However, we also need to consider the fact that the SSO message flow contains 3 more interactions with respect to the standard version. In fact, Server 12 redirects the initial request to the SSO server, which sends the cookie back to the client. The client must then repeat its request to Server 12. Despite a higher number of messages exchanged, the introduction of the SSO system still manages to

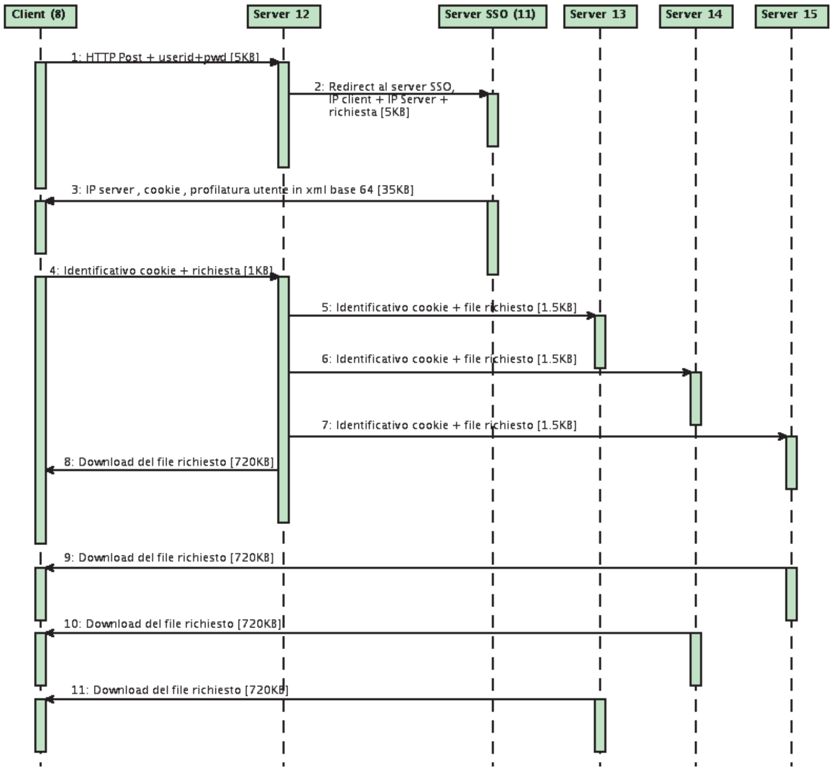

Client (8)
Server 12
Server SSO (11)
Server 13
Server 14
Server 15
1: HTTP Post + userid+pwd [5KB]
2: Redirect al server SSO, IP client + IP Server + richiesta [5KB]
3: IP server , cookie , profilatura utente in xml base 64 [35KB]
4: Identificativo cookie + richiesta [1KB]
5: Identificativo cookie + file richiesto [1.5KB]
6: Identificativo cookie + file richiesto [1.5KB]
7: Identificativo cookie + file richiesto [1.5KB]
8: Download del file richiesto [720KB]
9: Download del file richiesto [720KB]
10: Download del file richiesto [720KB]
11: Download del file richiesto [720KB]

Figure 28. Sequence Diagram: SSO Download Manager

```
& Src Dest  Acpu  Adisk  Aterm  Net     Bcpu  Bdisk
  1     2     100   100    500    5000    1500  1500;
  2     3     1500  1000   50     5000    1000  1000;
  1   2     100   100    500    1000    1500  1500;
  2     4     1500  1000   50     1500    1000  1000;
  2     5     1500  1000   50     1500    1000  1000;
  2     6     1500  1000   50     1500    1000  1000;
  2     1     100   100    100    720000  100   100;
  4     1     100   100    100    720000  100   100;
  5     1     100   100    100    720000  100   100;
  6     1     100   100    100    720000  100   100;
  0     0     -1; && Terminated

& SSO Server
& Src Dest  Acpu  Adisk  Aterm  Net       Bcpu     Bdisk
  3   1     100   50     200    35000   2500  2500;
  0   0     -1; && Terminated
```

Figure 29. SSO Download Manager - server.app and auth.app files

```
******** PACKET RECEIVED ***********
DIM.RECEIVED:   720000
- SOUR:        13 - DEST:         8
TIME:   0.1553E+06
***********************************
------------- ROUTING -------------
--->  ROUTER: 5     --->  WAY:    8
SOUR:        14       DEST:       8
DIM:     720000
TIME:   0.1953E+06
----------------------------------
------------- ROUTING -------------
--->  ROUTER: 5     --->  WAY:    8
SOUR:        12       DEST:       8
DIM:    720000
TIME:   0.2008E+06
----------------------------------
******** PACKET RECEIVED ***********
DIM.RECEIVED:   720000
- SOUR:        15 - DEST:         8
TIME:   0.5009E+06
***********************************

******** PACKET RECEIVED ***********
DIM.RECEIVED:   720000
- SOUR:        14 - DEST:         8
TIME:   0.5266E+06
***********************************
******** PACKET RECEIVED ***********
DIM.RECEIVED:   720000
- SOUR:        12 - DEST:         8
TIME:   0.6226E+06
***********************************

13 -1; && Terminated
APPLICATION TERMINATED AT:  0.6227E+06
```

Figure 30. SSO Download Manager simulation trace

reduce the simulation time by approximately 45 seconds, because it makes the authentication operations less burdensome for the network.

8.5. Mobile Agents

We can now move on to the definition of the last simulation case. In a mobile agent system we define the agent as the program running on behalf of the user. The fundamental characteristic of this type of architecture is that the agent is completely moved from one server to another. Since this distributed application is based on a strong mobility paradigm, the data migration involves both the program and the memory state. Therefore, this simulation will be characterized by very large data packets.

Figure 31 shows the message flow between the hosts. Client 8 starts the process, which is then moved from one server to another. Each server contributes to the execution of the process and sends the results of the computation back to

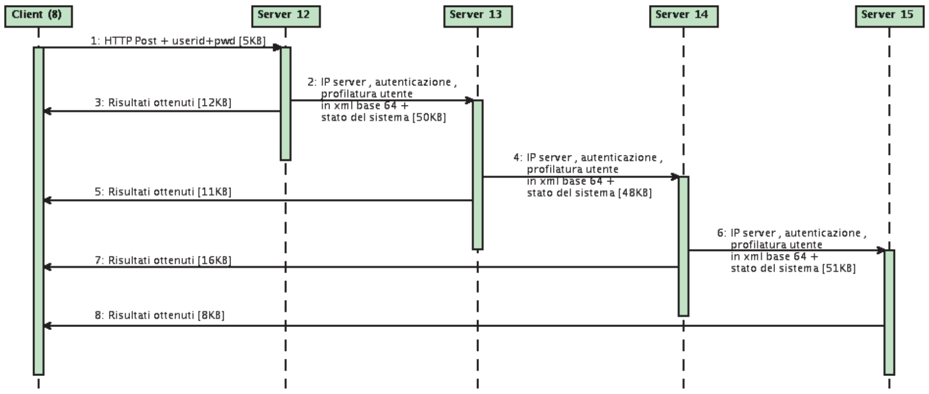

Client (8)
Server 12
Server 13
Server 14
Server 15
1: HTTP Post + userid+pwd [5KB]
2: IP server , autenticazione , profilatura utente in xml base 64 + stato del sistema [50KB]
3: Risultati ottenuti [12KB]
4: IP server , autenticazione , profilatura utente in xml base 64 + stato del sistema [48KB]
5: Risultati ottenuti [11KB]
6: IP server , autenticazione , profilatura utente in xml base 64 + stato del sistema [51KB]
7: Risultati ottenuti [16KB]
8: Risultati ottenuti [8KB]

Figure 31. Sequence Diagram: Mobile Agents

the client. The client needs to be authenticated in order to perform server requests. Therefore, when a server moves the process to another server, the packet will also contain the client's authentication data. As a result the destination server will be able to authenticate the client and send back the result of the computation.

```
& Src  Dest   Cpu   Adisk  Aterm  Net    Bcpu  Bdisk
  1    2      100   100    500    5000   3500  3500;
  2    3      100   100    500    50000  3500  3500;
  2    1      1500  1000   50     12000  1000  1000;
  3    4      100   100    500    48000  3500  3500;
  3    1      1500  1000   50     11000  1000  1000;
  4    5      100   100    500    51000  3500  3500;
  4    1      1500  1000   50     16000  1000  1000;
  5    1      1500  1000   50     8000   1000  1000;
  0    0    -1; && Terminated
```

Figure 32. Mobile Agents – server.app file

The first web application packet (Figure 32) represents the client's request, through which the process is activated. After that, each server proceeds to migrate the process to the next server, and to send back a response containing the computed data. The packets through which the migration is performed are very large, because they contain both the data calculated so far and the client authentication information (35KB).

The simulation trace (Figure 34) shows the transmission and the routing of the last two data packets, which contain the results computed by Servers 14 and 15. We can also see how Server 15 can only send its response to the client after receiving the process data. The optimal routing of the packets allows the application to end in 38 seconds without any major delays.

8.6. Single sign-on Mobile Agents

We end our simulation analysis by examining what happens to the mobile agent application in the presence of a Single Sign-On system. In this case the

```
***********************************
        PACKET SENT:
***********************************
- DIMENSION:     16000
- TYPE: risweb
- SOURID:        14
- SOURNP:         3
- DESTID:         8
- DESTNP:         1
- SEQNUM:         4
- WTOUT:  10000000
- TIME:   0.2102E+05
***********************************
******** PACKET RECEIVED **********
DIM.RECEIVED:   51000
- SOUR:        14 - DEST:        15
TIME:   0.2754E+05
***********************************
------------- ROUTING -------------
--->  ROUTER: 3     --->  WAY:  15
SOUR:       14       DEST:       8
DIM:    16000
TIME:   0.3148E+05
-----------------------------------
------------- ROUTING -------------
--->  ROUTER: 5     --->  WAY:   8
SOUR:       14       DEST:       8
DIM:    16000
TIME:   0.3220E+05
-----------------------------------
```

Figure 33. Mobile Agents simulation trace: intermediate phase

client authenticates itself before joining the network. Therefore, the process will be executed only after the initial authentication, through which Client 8 receives a ticket from the SSO server.

Figure 35 shows the message flow of mobile agents. The client sends an initial request to the SSO server, which contains its authentication data. The SSO server responds with the 35KB cookie, granting the client permission to interact with Server 12. The introduction of this initial authentication process has a positive effect on the application. In fact, the size of the process migration packets has been significantly reduced because it is no longer necessary to transmit the entire profiling of the user to each server. Including the cookie Id together with the process data is sufficient to identify the client whenever a new server is contacted.

As we can see from Figure 36, the structure of the application is similar to the standard version. However, the client only needs to send a 1KB packet to start the process on Server 12. The `auth.app` file contains the SSO server response, which consists of the 35KB cookie necessary to speed up the authentication procedures.

Figure 37 shows the transmission of the last two data packets sent by Servers 14 and 15. The application now ends at 35 seconds from the beginning of the

```
******** PACKET RECEIVED *********
DIM.RECEIVED:   16000
- SOUR:        14 - DEST:        8
TIME:   0.3375E+05
**********************************

**********************************
        PACKET SENT:
**********************************
- DIMENSION:      8000
- TYPE: risweb
- SOURID:        15
- SOURNP:         4
- DESTID:         8
- DESTNP:         1
- SEQNUM:         5
- WTOUT:  10000000
- TIME:   0.3422E+05
**********************************
------------- ROUTING -------------
--->  ROUTER: 3     --->  WAY:  15
SOUR:       15       DEST:       8
DIM:    8000
TIME:   0.3434E+05
-----------------------------------
------------- ROUTING -------------
--->  ROUTER: 5     --->  WAY:   8
SOUR:       15       DEST:       8
DIM:    8000
TIME:   0.3618E+05
-----------------------------------
******** PACKET RECEIVED *********
DIM.RECEIVED:   8000
- SOUR:        15 - DEST:        8
TIME:   0.3780E+05
**********************************

11  -1; && Terminated
APPLICATION TERMINATED AT: 0.3791E+05
```

Figure 34. Mobile Agents simulation trace: final phase

simulation. The introduction of the SSO is beneficial as it reduces the simulation time; however, the difference with the standard case is not as evident as in the download manager or navigation applications. In the next section we are going to compare and analyze the results of each simulation case in more details.

9. Simulation Results

In this section we will analyze the network congestion levels generated by the execution of the three different web application types. We will also show a final comparison of the results, both in the presence and absence of a Single Sign-On system.

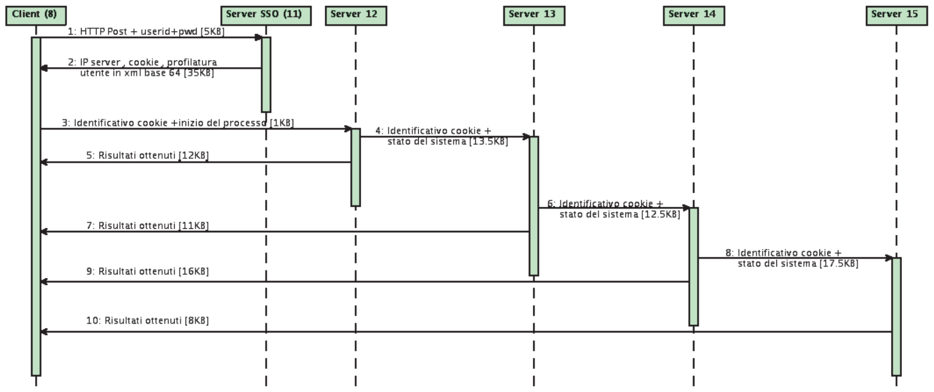

Client (8)
Server SSO (11)
Server 12
Server 13
Server 14
Server 15
1: HTTP Post + userid+pwd [5KB]
2: IP server , cookie , profilatura utente in xml base 64 [35KB]
3: Identificativo cookie +inizio del processo [1KB]
4: Identificativo cookie + stato del sistema [13.5KB]
5: Risultati ottenuti [12KB]
6: Identificativo cookie + stato del sistema [12.5KB]
7: Risultati ottenuti [11KB]
8: Identificativo cookie + stato del sistema [17.5KB]
9: Risultati ottenuti [16KB]
10: Risultati ottenuti [8KB]

Figure 35. Sequence Diagram: SSO Mobile Agents

```
& Src Dest   Acpu  Adisk Aterm  Net   Bcpu  Bdisk
    1    3   100   100   500    5000  3500  3500;
    1    2   100   100   500    1000  3500  3500;
    2    4   100   100   500    13500 3500  3500;
    2    1   1500  1000  50     12000 1000  1000;
    4    5   100   100   500    12500 3500  3500;
    4    1   1500  1000  50     11000 1000  1000;
    5    6   100   100   500    17500 3500  3500;
    5    1   1500  1000  50     16000 1000  1000;
    6    1   1500  1000  50     8000  1000  1000;
    0    0   -1; && Terminated

& SSO Server
& Src  Dest  Acpu  Adisk Aterm  Net   Bcpu  Bdisk
    3    1   100   50    200    35000 2500  2500;
    0    0     -1; && Terminated
```

Figure 36. SSO Mobile Agents – server.app and auth.app files

9.1. Congestion Levels

In the following pages we will show tables containing the congestion values observed during different simulations. For the sake of simplicity, we only consider the routers and the ways involved in the transmission of the web application traffic. Each entry of the table identifies a specific network device within the network, as well as the relative service stations. For each device, the following values were reported:

- `SERVICE:` average queue service time T_{serv} ;
- `BUSY PCT:` queue congestion level;
- `CUST NB:` average number of customers in the queue;
- `RESPONSE:` average queue response time T_{resp}. The response time can be calculated as $T_{resp} = T_{wait} + T_{serv}$, where T_{wait} is the average queue waiting time;
- `SERV NB:` the total number of customers served by the station.

```
*********************************
        PACKET SENT:
*********************************
- DIMENSION:      8000
- TYPE: risweb
- SOURID:        15
- SOURNP:         5
- DESTID:         8
- DESTNP:         1
- SEQNUM:         7
- WTOUT:  10000000
- TIME:   0.2437E+05
*********************************
------------- ROUTING -------------
--->  ROUTER: 3     --->  WAY:  15
SOUR:       14       DEST:       8
DIM:    16000
TIME:   0.3117E+05
-----------------------------------
------------- ROUTING -------------
--->  ROUTER: 3     --->  WAY:  15
SOUR:       15       DEST:       8
DIM:    8000
TIME:   0.3168E+05
-----------------------------------
------------- ROUTING -------------
--->  ROUTER: 5     --->  WAY:   8
SOUR:       14       DEST:       8
DIM:    16000
TIME:   0.3186E+05
-----------------------------------
------------- ROUTING -------------
--->  ROUTER: 5     --->  WAY:   8
SOUR:       15       DEST:       8
DIM:    8000
TIME:   0.3306E+05
-----------------------------------
******** PACKET RECEIVED ***********
DIM.RECEIVED:   16000
- SOUR:        14 - DEST:         8
TIME:   0.3354E+05
************************************
******** PACKET RECEIVED ***********
DIM.RECEIVED:   8000
- SOUR:        15 - DEST:         8
TIME:   0.3499E+05
************************************

12 -1; && Terminated
APPLICATION TERMINATED  0.3557E+05
```

Figure 37. SSO Mobile Agents simulation trace

Moreover, the results were grouped according to the specific traffic class:

- `EMISSION`: standard traffic class packets emitted by the station;
- `RICHIN`: standard traffic class packets received by the station;
- `WEBAP`: web application class packets transmitted by the station.

```
*******************************************************************
*   NAME   *  SERVICE * BUSY PCT *  CUST NB * RESPONSE *  SERV NB *
*******************************************************************
*          *          *          *          *          *          *
*ROUTER 5. *          *          *          *          *          *
*CPU     . * 2.726    *0.9751E-01*0.4859    * 13.59    *    21458*
*(EMISSION)* 1.595    *0.4294E-01*0.3701    * 13.75    *    16150*
*(RICHIN  )* 4.441    *0.3920E-01*0.1004    * 11.37    *     5296*
*(WEBAP   )* 768.2    *0.1536E-01*0.1538E-01* 769.0    *       12*
*          *          *          *          *          *          *
*ROUTER 6. *          *          *          *          *          *
*CPU     . * 2.305    *0.5300E-01*0.1169    * 5.086    *    13796*
*(EMISSION)* 1.287    *0.2222E-01*0.7842E-01* 4.543    *    10358*
*(RICHIN  )* 3.698    *0.2112E-01*0.2885E-01* 5.053    *     3426*
*(WEBAP   )* 483.2    *0.9664E-02*0.9664E-02* 483.2    *       12*
*          *          *          *          *          *          *
*ROUTER 8. *          *          *          *          *          *
*CPU     . * 6.137    *0.1064    *0.7506    * 43.30    *    10400*
*(EMISSION)* 3.121    *0.4070E-01*0.5951    * 45.63    *     7824*
*(RICHIN  )* 9.199    *0.3925E-01*0.1290    * 30.23    *     2560*
*(WEBAP   )* 991.1    *0.2643E-01*0.2656E-01* 996.1    *       16*
*          *          *          *          *          *          *
*WAY    8. *          *          *          *          *          *
*CHANNEL  . * 3.179    *0.0000E+00*0.1001E-01* 3.179    *     1890*
*(EMISSION)*0.8205    *0.0000E+00*0.1979E-02*0.8205    *     1447*
*(RICHIN  )* 2.236    *0.0000E+00*0.1621E-02* 2.236    *      435*
*(WEBAP   )* 481.1    *0.0000E+00*0.6415E-02* 481.1    *        8*
*          *          *          *          *          *          *
*WAY    9. *          *          *          *          *          *
*CHANNEL  . * 1.659    *0.0000E+00*0.9030E-02* 1.659    *     3265*
*(EMISSION)*0.7908    *0.0000E+00*0.3200E-02*0.7908    *     2428*
*(RICHIN  )* 2.270    *0.0000E+00*0.3136E-02* 2.270    *      829*
*(WEBAP   )* 202.0    *0.0000E+00*0.2693E-02* 202.0    *        8*
*          *          *          *          *          *          *
*WAY   11. *          *          *          *          *          *
*CHANNEL  . * 1.273    *0.0000E+00*0.1445E-01* 1.273    *     6810*
*(EMISSION)*0.7851    *0.0000E+00*0.6688E-02*0.7851    *     5111*
*(RICHIN  )* 2.217    *0.0000E+00*0.6248E-02* 2.217    *     1691*
*(WEBAP   )* 113.8    *0.0000E+00*0.1517E-02* 113.8    *        8*
*          *          *          *          *          *          *
*******************************************************************
```

Figure 38. Navigation simulation results

We can see from the simulation results that despite the elevated number of standard traffic packets flowing into the system, the congestion levels remain within acceptable limits. Even though the number of web application packets transmitted by the network devices might seem low, this is due to the fact that data packets can never be lost and re-transmitted by the hosts.

In general, the highest congestion levels can be observed in the standard download manager application. The Cpu station of Router 5 presents 16.81% of congestion, while in Router 8 this value goes up to 26.40%. In the SSO version, these two values decrease to 14.61% and 24.67%, respectively. These values cannot be considered elevated, and they will not be a source of major delays during the execution of the application. However, the fact that Router 5 presents above

```
*******************************************************************
*   NAME   *  SERVICE * BUSY PCT *  CUST NB * RESPONSE *  SERV NB *
*******************************************************************
*          *          *          *          *          *          *
*ROUTER 4. *          *          *          *          *          *
*CPU     . * 1.881    *0.5354E-01*0.5988E-01* 2.104    *     17081*
*(EMISSION)* 1.290    *0.2759E-01*0.3239E-01* 1.514    *     12833*
*(RICHIN  )* 3.665    *0.2594E-01*0.2747E-01* 3.882    *      4246*
*(WEBAP   )* 3.184    *0.1061E-04*0.2030E-04* 6.090    *         2*
*          *          *          *          *          *          *
*ROUTER 5. *          *          *          *          *          *
*CPU     . * 2.492    *0.9099E-01*0.2888    * 7.909    *     21911*
*(EMISSION)* 1.628    *0.4468E-01*0.2185    * 7.958    *     16471*
*(RICHIN  )* 4.453    *0.4032E-01*0.6435E-01* 7.108    *      5432*
*(WEBAP   )* 449.5    *0.5993E-02*0.5996E-02* 449.7    *         8*
*          *          *          *          *          *          *
*ROUTER 6. *          *          *          *          *          *
*CPU     . * 2.470    *0.5470E-01*0.3389    * 15.30    *     13290*
*(EMISSION)* 1.277    *0.2124E-01*0.3017    * 18.14    *      9979*
*(RICHIN  )* 3.420    *0.1882E-01*0.2259E-01* 4.107    *      3301*
*(WEBAP   )* 878.7    *0.1465E-01*0.1466E-01* 879.8    *        10*
*          *          *          *          *          *          *
*ROUTER 8. *          *          *          *          *          *
*CPU     . * 5.531    *0.9589E-01*0.4021    * 23.20    *     10402*
*(EMISSION)* 3.058    *0.3984E-01*0.3259    * 25.02    *      7818*
*(RICHIN  )* 8.936    *0.3831E-01*0.5842E-01* 13.63    *      2572*
*(WEBAP   )* 887.4    *0.1775E-01*0.1778E-01* 888.9    *        12*
*          *          *          *          *          *          *
*WAY    4. *          *          *          *          *          *
*CHANNEL . * 1.178    *0.0000E+00*0.1181E-01* 1.178    *      6013*
*(EMISSION)*0.8070    *0.0000E+00*0.6090E-02*0.8070    *      4528*
*(RICHIN  )* 2.275    *0.0000E+00*0.5622E-02* 2.275    *      1483*
*(WEBAP   )* 28.55    *0.0000E+00*0.9518E-04* 28.55    *         2*
*          *          *          *          *          *          *
*WAY    7. *          *          *          *          *          *
*CHANNEL . * 1.117    *0.0000E+00*0.6170E-02* 1.117    *      3315*
*(EMISSION)*0.7567    *0.0000E+00*0.3119E-02*0.7567    *      2473*
*(RICHIN  )* 2.165    *0.0000E+00*0.3031E-02* 2.165    *       840*
*(WEBAP   )* 5.895    *0.0000E+00*0.1965E-04* 5.895    *         2*
*          *          *          *          *          *          *
*WAY    8. *          *          *          *          *          *
*CHANNEL . * 2.044    *0.0000E+00*0.6193E-02* 2.044    *      1818*
*(EMISSION)*0.7637    *0.0000E+00*0.1731E-02*0.7637    *      1360*
*(RICHIN  )* 2.193    *0.0000E+00*0.1652E-02* 2.193    *       452*
*(WEBAP   )* 280.9    *0.0000E+00*0.2809E-02* 280.9    *         6*
*          *          *          *          *          *          *
*WAY    9. *          *          *          *          *          *
*CHANNEL . * 1.215    *0.0000E+00*0.4622E-02* 1.215    *      2283*
*(EMISSION)*0.8418    *0.0000E+00*0.2417E-02*0.8418    *      1723*
*(RICHIN  )* 2.182    *0.0000E+00*0.2022E-02* 2.182    *       556*
*(WEBAP   )* 27.49    *0.0000E+00*0.1833E-03* 27.49    *         4*
*          *          *          *          *          *          *
*WAY   11. *          *          *          *          *          *
*CHANNEL . * 1.361    *0.0000E+00*0.1372E-01* 1.361    *      6050*
*(EMISSION)*0.8080    *0.0000E+00*0.6091E-02*0.8080    *      4523*
*(RICHIN  )* 2.263    *0.0000E+00*0.5743E-02* 2.263    *      1523*
*(WEBAP   )* 283.5    *0.0000E+00*0.1890E-02* 283.5    *         4*
*          *          *          *          *          *          *
```

Figure 39. SSO Navigation simulation results

```
*******************************************************************
*   NAME   *  SERVICE * BUSY PCT *  CUST NB * RESPONSE *  SERV NB *
*******************************************************************
*          *          *          *          *          *          *
*ROUTER 3. *          *          *          *          *          *
*CPU     . * 4.303    *0.7510E-01* 13.16    * 753.9    *    52359*
*(EMISSION)* 1.435    *0.1919E-01* 12.39    * 926.0    *    40126*
*(RICHIN  )* 3.969    *0.1618E-01*0.7308    * 179.3    *    12225*
*(WEBAP   )*0.1490E+05*0.3973E-01*0.4123E-01*0.1546E+05*        8*
*          *          *          *          *          *          *
*ROUTER 5. *          *          *          *          *          *
*CPU     . * 4.634    *0.1681    * 163.8    * 4517.    *   108791*
*(EMISSION)* 1.643    *0.4587E-01* 158.1    * 5661.    *    83768*
*(RICHIN  )* 4.411    *0.3677E-01* 5.523    * 662.5    *    25009*
*(WEBAP   )*0.1830E+05*0.8542E-01*0.2137    *0.4578E+05*       14*
*          *          *          *          *          *          *
*ROUTER 6. *          *          *          *          *          *
*CPU     . * 4.507    *0.1160    * 47.25    * 1836.    *    77198*
*(EMISSION)* 1.330    *0.2654E-01* 46.35    * 2322.    *    59874*
*(RICHIN  )* 3.569    *0.2059E-01*0.6490    * 112.5    *    17311*
*(WEBAP   )*0.1589E+05*0.6885E-01*0.2511    *0.5794E+05*       13*
*          *          *          *          *          *          *
*ROUTER 8. *          *          *          *          *          *
*CPU     . * 11.89    *0.2640    * 488.4    *0.2200E+05*    66607*
*(EMISSION)* 3.230    *0.5569E-01* 428.6    *0.2486E+05*    51721*
*(RICHIN  )* 8.893    *0.4410E-01* 59.61    *0.1202E+05*    14876*
*(WEBAP   )*0.4927E+05*0.1642    *0.1895    *0.5685E+05*       10*
*          *          *          *          *          *          *
*WAY    8. *          *          *          *          *          *
*CHANNEL . * 8.068    *0.0000E+00*0.2734E-01* 8.068    *    10166*
*(EMISSION)*0.8001    *0.0000E+00*0.2094E-02*0.8001    *     7853*
*(RICHIN  )* 2.226    *0.0000E+00*0.1712E-02* 2.226    *     2308*
*(WEBAP   )*0.1412E+05*0.0000E+00*0.2353E-01*0.1412E+05*        5*
*          *          *          *          *          *          *
*WAY    9. *          *          *          *          *          *
*CHANNEL . * 1.100    *0.0000E+00*0.9718E-02* 1.100    *    26513*
*(EMISSION)*0.7829    *0.0000E+00*0.5397E-02*0.7829    *    20683*
*(RICHIN  )* 2.207    *0.0000E+00*0.4285E-02* 2.207    *     5825*
*(WEBAP   )* 21.74    *0.0000E+00*0.3623E-04* 21.74    *        5*
*          *          *          *          *          *          *
*WAY   10. *          *          *          *          *          *
*CHANNEL . * 1.124    *0.0000E+00*0.4245E-02* 1.124    *    11329*
*(EMISSION)*0.7836    *0.0000E+00*0.2264E-02*0.7836    *     8669*
*(RICHIN  )* 2.237    *0.0000E+00*0.1981E-02* 2.237    *     2657*
*(WEBAP   )*0.4330E-01*0.0000E+00*0.4330E-07*0.4330E-01*        3*
*          *          *          *          *          *          *
*WAY   11. *          *          *          *          *          *
*CHANNEL . * 1.346    *0.0000E+00*0.1698E-01* 1.346    *    37842*
*(EMISSION)*0.8570    *0.0000E+00*0.8385E-02*0.8570    *    29353*
*(RICHIN  )* 2.219    *0.0000E+00*0.6273E-02* 2.219    *     8481*
*(WEBAP   )* 869.2    *0.0000E+00*0.2318E-02* 869.2    *        8*
*          *          *          *          *          *          *
*WAY   14. *          *          *          *          *          *
*CHANNEL . * 1.316    *0.0000E+00*0.6392E-02* 1.316    *    14571*
*(EMISSION)*0.8218    *0.0000E+00*0.3069E-02*0.8218    *    11203*
*(RICHIN  )* 2.212    *0.0000E+00*0.2480E-02* 2.212    *     3364*
*(WEBAP   )* 632.1    *0.0000E+00*0.8429E-03* 632.1    *        4*
*          *          *          *          *          *          *
*WAY   15. *          *          *          *          *          *
*CHANNEL . * 3.099    *0.0000E+00*0.2749E-01* 3.099    *    26609*
*(EMISSION)*0.8113    *0.0000E+00*0.5504E-02*0.8113    *    20352*
*(RICHIN  )* 2.165    *0.0000E+00*0.4514E-02* 2.165    *     6253*
*(WEBAP   )*0.1311E+05*0.0000E+00*0.1747E-01*0.1311E+05*        4*
*          *          *          *          *          *          *
```

Figure 40. Download Manager simulation results

```
*******************************************************************
*   NAME   *  SERVICE * BUSY PCT *  CUST NB * RESPONSE *  SERV NB *
*******************************************************************
*          *          *          *          *          *          *
*ROUTER 3. *          *          *          *          *          *
*CPU     . * 2.705    *0.4599E-01* 1.066    * 62.69    *     51012*
*(EMISSION)* 1.445    *0.1880E-01* 1.026    * 78.88    *     39028*
*(RICHIN  )* 3.972    *0.1586E-01*0.1735E-01* 4.347    *     11976*
*(WEBAP   )* 4250.    *0.1133E-01*0.2254E-01* 8451.    *         8*
*          *          *          *          *          *          *
*ROUTER 5. *          *          *          *          *          *
*CPU     . * 4.122    *0.1461    * 131.6    * 3713.    *    106296*
*(EMISSION)* 1.641    *0.4472E-01* 125.5    * 4604.    *     81761*
*(RICHIN  )* 4.402    *0.3598E-01* 5.893    * 720.9    *     24520*
*(WEBAP   )*0.1307E+05*0.6536E-01*0.1831    *0.3662E+05*        15*
*          *          *          *          *          *          *
*ROUTER 6. *          *          *          *          *          *
*CPU     . * 2.891    *0.7152E-01* 7.536    * 304.6    *     74228*
*(EMISSION)* 1.308    *0.2493E-01* 7.435    * 390.1    *     57175*
*(RICHIN  )* 3.601    *0.2045E-01*0.4921E-01* 8.666    *     17036*
*(WEBAP   )* 4614.    *0.2615E-01*0.5194E-01* 9166.    *        17*
*          *          *          *          *          *          *
*ROUTER 8. *          *          *          *          *          *
*CPU     . * 11.40    *0.2467    * 621.3    *0.2871E+05*     64929*
*(EMISSION)* 3.212    *0.5373E-01* 550.8    *0.3293E+05*     50183*
*(RICHIN  )* 8.838    *0.4341E-01* 70.10    *0.1427E+05*     14734*
*(WEBAP   )*0.3739E+05*0.1496    *0.3947    *0.9866E+05*        12*
*          *          *          *          *          *          *
*WAY    8. *          *          *          *          *          *
*CHANNEL . * 25.18    *0.0000E+00*0.8127E-01* 25.18    *      9684*
*(EMISSION)*0.8506    *0.0000E+00*0.2116E-02*0.8506    *      7464*
*(RICHIN  )* 2.174    *0.0000E+00*0.1605E-02* 2.174    *      2214*
*(WEBAP   )*0.3877E+05*0.0000E+00*0.7754E-01*0.3877E+05*         6*
*          *          *          *          *          *          *
*WAY    9. *          *          *          *          *          *
*CHANNEL . * 1.097    *0.0000E+00*0.9159E-02* 1.097    *     25049*
*(EMISSION)*0.7570    *0.0000E+00*0.4853E-02*0.7570    *     19231*
*(RICHIN  )* 2.198    *0.0000E+00*0.4258E-02* 2.198    *      5812*
*(WEBAP   )* 24.41    *0.0000E+00*0.4882E-04* 24.41    *         6*
*          *          *          *          *          *          *
*WAY   10. *          *          *          *          *          *
*CHANNEL . * 1.101    *0.0000E+00*0.4074E-02* 1.101    *     11104*
*(EMISSION)*0.7597    *0.0000E+00*0.2158E-02*0.7597    *      8522*
*(RICHIN  )* 2.229    *0.0000E+00*0.1916E-02* 2.229    *      2579*
*(WEBAP   )*0.3699E-01*0.0000E+00*0.3699E-07*0.3699E-01*         3*
*          *          *          *          *          *          *
*WAY   11. *          *          *          *          *          *
*CHANNEL . * 1.621    *0.0000E+00*0.1954E-01* 1.621    *     36154*
*(EMISSION)*0.8126    *0.0000E+00*0.7542E-02*0.8126    *     27841*
*(RICHIN  )* 2.225    *0.0000E+00*0.6158E-02* 2.225    *      8304*
*(WEBAP   )* 1945.    *0.0000E+00*0.5836E-02* 1945.    *         9*
*          *          *          *          *          *          *
*WAY   14. *          *          *          *          *          *
*CHANNEL . * 1.125    *0.0000E+00*0.5360E-02* 1.125    *     14297*
*(EMISSION)*0.8025    *0.0000E+00*0.2927E-02*0.8025    *     10942*
*(RICHIN  )* 2.148    *0.0000E+00*0.2399E-02* 2.148    *      3351*
*(WEBAP   )* 25.56    *0.0000E+00*0.3408E-04* 25.56    *         4*
*          *          *          *          *          *          *
*WAY   15. *          *          *          *          *          *
*CHANNEL . * 4.594    *0.0000E+00*0.3987E-01* 4.594    *     26034*
*(EMISSION)*0.8096    *0.0000E+00*0.5397E-02*0.8096    *     20001*
*(RICHIN  )* 2.219    *0.0000E+00*0.4459E-02* 2.219    *      6029*
*(WEBAP   )*0.2251E+05*0.0000E+00*0.3001E-01*0.2251E+05*         4*
*          *          *          *          *          *          *
```

Figure 41. SSO Download Manager simulation results

```
*******************************************************************
*   NAME   *  SERVICE * BUSY PCT *  CUST NB * RESPONSE *  SERV NB *
*******************************************************************
*          *          *          *          *          *          *
*ROUTER 3. *          *          *          *          *          *
*CPU     . * 2.383    *0.3903E-01*0.3357    * 20.49    *     49149*
*(EMISSION)* 1.396    *0.1717E-01*0.2774    * 22.57    *     36877*
*(RICHIN  )* 3.950    *0.1615E-01*0.5041E-01* 12.33    *     12264*
*(WEBAP   )* 2145.    *0.5721E-02*0.7887E-02* 2958.    *         8*
*          *          *          *          *          *          *
*ROUTER 5. *          *          *          *          *          *
*CPU     . * 2.311    *0.7700E-01*0.1478    * 4.436    *     99970*
*(EMISSION)* 1.551    *0.3882E-01*0.9719E-01* 3.884    *     75074*
*(RICHIN  )* 4.378    *0.3631E-01*0.4851E-01* 5.849    *     24884*
*(WEBAP   )* 465.9    *0.1864E-02*0.2114E-02* 528.4    *        12*
*          *          *          *          *          *          *
*ROUTER 6. *          *          *          *          *          *
*CPU     . * 1.868    *0.4289E-01*0.5882E-01* 2.562    *     68874*
*(EMISSION)* 1.259    *0.2172E-01*0.3277E-01* 1.900    *     51733*
*(RICHIN  )* 3.535    *0.2019E-01*0.2454E-01* 4.297    *     17132*
*(WEBAP   )* 328.6    *0.9857E-03*0.1507E-02* 502.5    *         9*
*          *          *          *          *          *          *
*ROUTER 8. *          *          *          *          *          *
*CPU     . * 4.651    *0.9204E-01*0.2335    * 11.80    *     59362*
*(EMISSION)* 3.042    *0.4516E-01*0.1570    * 10.57    *     44538*
*(RICHIN  )* 8.763    *0.4327E-01*0.7295E-01* 14.77    *     14814*
*(WEBAP   )* 1083.    *0.3609E-02*0.3629E-02* 1089.    *        10*
*          *          *          *          *          *          *
*WAY    8. *          *          *          *          *          *
*CHANNEL . * 1.188    *0.0000E+00*0.3595E-02* 1.188    *      9077*
*(EMISSION)*0.7515    *0.0000E+00*0.1694E-02*0.7515    *      6764*
*(RICHIN  )* 2.193    *0.0000E+00*0.1687E-02* 2.193    *      2308*
*(WEBAP   )* 127.7    *0.0000E+00*0.2128E-03* 127.7    *         5*
*          *          *          *          *          *          *
*WAY    9. *          *          *          *          *          *
*CHANNEL . * 1.133    *0.0000E+00*0.8728E-02* 1.133    *     23107*
*(EMISSION)*0.7812    *0.0000E+00*0.4526E-02*0.7812    *     17382*
*(RICHIN  )* 2.185    *0.0000E+00*0.4165E-02* 2.185    *      5720*
*(WEBAP   )* 21.77    *0.0000E+00*0.3629E-04* 21.77    *         5*
*          *          *          *          *          *          *
*WAY   10. *          *          *          *          *          *
*CHANNEL . * 1.131    *0.0000E+00*0.4015E-02* 1.131    *     10655*
*(EMISSION)*0.7693    *0.0000E+00*0.2040E-02*0.7693    *      7957*
*(RICHIN  )* 2.197    *0.0000E+00*0.1975E-02* 2.197    *      2697*
*(WEBAP   )*0.4769E-01*0.0000E+00*0.1590E-07*0.4769E-01*         1*
*          *          *          *          *          *          *
*WAY   11. *          *          *          *          *          *
*CHANNEL . * 1.240    *0.0000E+00*0.1395E-01* 1.240    *     33760*
*(EMISSION)*0.7812    *0.0000E+00*0.6618E-02*0.7812    *     25413*
*(RICHIN  )* 2.216    *0.0000E+00*0.6162E-02* 2.216    *      8341*
*(WEBAP   )* 585.0    *0.0000E+00*0.1170E-02* 585.0    *         6*
*          *          *          *          *          *          *
*WAY   14. *          *          *          *          *          *
*CHANNEL . * 1.227    *0.0000E+00*0.5585E-02* 1.227    *     13654*
*(EMISSION)*0.7851    *0.0000E+00*0.2705E-02*0.7851    *     10338*
*(RICHIN  )* 2.149    *0.0000E+00*0.2373E-02* 2.149    *      3313*
*(WEBAP   )* 506.7    *0.0000E+00*0.5067E-03* 506.7    *         3*
*          *          *          *          *          *          *
*WAY   15. *          *          *          *          *          *
*CHANNEL . * 1.164    *0.0000E+00*0.9680E-02* 1.164    *     24938*
*(EMISSION)*0.7698    *0.0000E+00*0.4781E-02*0.7698    *     18633*
*(RICHIN  )* 2.213    *0.0000E+00*0.4650E-02* 2.213    *      6302*
*(WEBAP   )* 248.7    *0.0000E+00*0.2487E-03* 248.7    *         3*
*          *          *          *          *          *          *
```

Figure 42. Mobile Agent simulation results

```
*******************************************************************
*   NAME   *  SERVICE * BUSY PCT *  CUST NB * RESPONSE *  SERV NB *
*******************************************************************
*          *          *          *          *          *          *
*ROUTER 3. *          *          *          *          *          *
*CPU     . * 2.238    *0.3638E-01*0.1918    * 11.80    *     48757*
*(EMISSION)* 1.385    *0.1691E-01*0.1609    * 13.19    *     36610*
*(RICHIN  )* 3.965    *0.1605E-01*0.2203E-01* 5.443    *     12139*
*(WEBAP   )* 1285.    *0.3428E-02*0.8824E-02* 3309.    *         8*
*          *          *          *          *          *          *
*ROUTER 5. *          *          *          *          *          *
*CPU     . * 2.284    *0.7595E-01*0.1535    * 4.617    *     99753*
*(EMISSION)* 1.528    *0.3811E-01*0.1039    * 4.165    *     74813*
*(RICHIN  )* 4.382    *0.3641E-01*0.4796E-01* 5.772    *     24926*
*(WEBAP   )* 305.8    *0.1427E-02*0.1685E-02* 361.1    *        14*
*          *          *          *          *          *          *
*ROUTER 6. *          *          *          *          *          *
*CPU     . * 1.866    *0.4274E-01*0.5067E-01* 2.212    *     68720*
*(EMISSION)* 1.257    *0.2162E-01*0.2805E-01* 1.632    *     51573*
*(RICHIN  )* 3.535    *0.2019E-01*0.2141E-01* 3.748    *     17134*
*(WEBAP   )* 216.1    *0.9365E-03*0.1214E-02* 280.1    *        13*
*          *          *          *          *          *          *
*ROUTER 8. *          *          *          *          *          *
*CPU     . * 4.739    *0.9356E-01*0.2235    * 11.32    *     59228*
*(EMISSION)* 3.102    *0.4605E-01*0.1579    * 10.64    *     44532*
*(RICHIN  )* 8.970    *0.4390E-01*0.6033E-01* 12.33    *     14682*
*(WEBAP   )* 774.8    *0.3616E-02*0.5228E-02* 1120.    *        14*
*          *          *          *          *          *          *
*WAY    8. *          *          *          *          *          *
*CHANNEL  . * 1.811    *0.0000E+00*0.5330E-02* 1.811    *      8829*
*(EMISSION)*0.7694    *0.0000E+00*0.1698E-02*0.7694    *      6622*
*(RICHIN  )* 2.255    *0.0000E+00*0.1654E-02* 2.255    *      2200*
*(WEBAP   )* 847.6    *0.0000E+00*0.1978E-02* 847.6    *         7*
*          *          *          *          *          *          *
*WAY    9. *          *          *          *          *          *
*CHANNEL  . * 1.128    *0.0000E+00*0.8632E-02* 1.128    *     22962*
*(EMISSION)*0.7729    *0.0000E+00*0.4462E-02*0.7729    *     17318*
*(RICHIN  )* 2.195    *0.0000E+00*0.4124E-02* 2.195    *      5637*
*(WEBAP   )* 19.82    *0.0000E+00*0.4624E-04* 19.82    *         7*
*          *          *          *          *          *          *
*WAY   10. *          *          *          *          *          *
*CHANNEL  . * 1.165    *0.0000E+00*0.4076E-02* 1.165    *     10500*
*(EMISSION)*0.8031    *0.0000E+00*0.2126E-02*0.8031    *      7943*
*(RICHIN  )* 2.289    *0.0000E+00*0.1950E-02* 2.289    *      2556*
*(WEBAP   )*0.4469E-01*0.0000E+00*0.1490E-07*0.4469E-01*         1*
*          *          *          *          *          *          *
*WAY   11. *          *          *          *          *          *
*CHANNEL  . * 1.211    *0.0000E+00*0.1351E-01* 1.211    *     33462*
*(EMISSION)*0.7777    *0.0000E+00*0.6474E-02*0.7777    *     24975*
*(RICHIN  )* 2.156    *0.0000E+00*0.6093E-02* 2.156    *      8479*
*(WEBAP   )* 353.1    *0.0000E+00*0.9417E-03* 353.1    *         8*
*          *          *          *          *          *          *
*WAY   14. *          *          *          *          *          *
*CHANNEL  . * 1.158    *0.0000E+00*0.5231E-02* 1.158    *     13553*
*(EMISSION)*0.7786    *0.0000E+00*0.2622E-02*0.7786    *     10101*
*(RICHIN  )* 2.200    *0.0000E+00*0.2529E-02* 2.200    *      3449*
*(WEBAP   )* 79.59    *0.0000E+00*0.7959E-04* 79.59    *         3*
*          *          *          *          *          *          *
*WAY   15. *          *          *          *          *          *
*CHANNEL  . * 1.144    *0.0000E+00*0.9404E-02* 1.144    *     24655*
*(EMISSION)*0.7823    *0.0000E+00*0.4841E-02*0.7823    *     18564*
*(RICHIN  )* 2.175    *0.0000E+00*0.4413E-02* 2.175    *      6088*
*(WEBAP   )* 149.9    *0.0000E+00*0.1499E-03* 149.9    *         3*
*          *          *          *          *          *          *
```

Figure 43. SSO Mobile Agent simulation results

average congestion levels is explained by the network structure itself. Being located in a central position, this router has the greatest number of connections in the network. Thus, it has to serve a higher number of packets compared to the other routers. On average, this number accounts for 108 791 both standard and web application data packets.

Router 8 presents higher congestion levels than Router 5, even if the latter serves a higher number of packets. The reason behind this is that Client 8 resides in the same sub-net defined by Router 8. During the final part of the application, this client must receive the four 720KB files from different servers. The task of processing and routing these large files falls on Router 8, which will inevitably experience a rise in the congestion levels.

The other simulation cases do not present any noteworthy results. All the applications experience normal congestion levels, and are able to terminate in a reasonable time without experiencing any major delays. This is imputable both to the introduction of a static routing system, and the elimination of re-transmissions to compensate the packet loss.

9.2. Comparison of results

We conclude our discussion by comparing the results obtained from the simulation of the web applications defined in Section 8. The introduction of the new routing system, as well as the changes described in the previous paragraphs, make it possible to build a system that represents an optimal environment for the execution of web applications. In fact, each simulation case terminates within a reasonable time despite the presence of an elevated number of standard traffic packets.

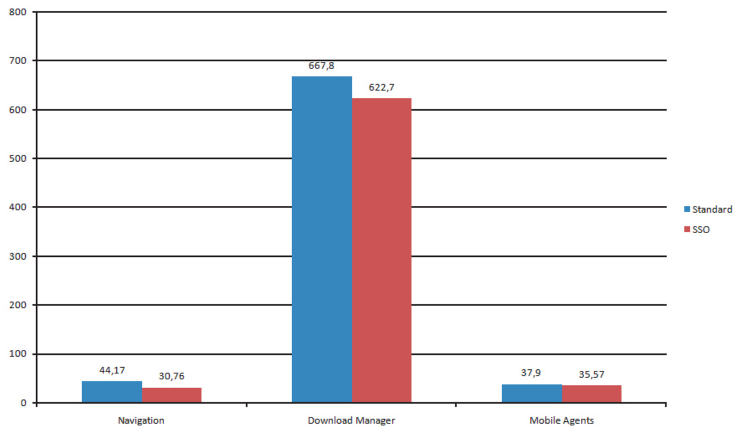


Figure 44. Simulation Results

Figure 44 shows a comparison of the execution times expressed in seconds of the three simulation cases.

We can observe from the navigation case results a difference of about 14 seconds between the standard version and the SSO version. This means that the activation of an SSO server entails a reduction of the simulation time by about one third. Both applications involve two client requests for web pages of equal size that are hosted by the same servers. However, since the SSO system speeds up the authentication phase, the latter terminates much earlier. We need to keep in mind that the navigation case deals with smaller packets and with fewer hosts when compared to the other simulation cases. It is clear that the improvements derived from the activation of the SSO server would have be more evident if the client had to perform a higher number of requests. In fact, for each new request performed by the user, the standard version experiments a delay caused by the repetition of the authentication process.

In the case of the Download Manager the activation of the SSO system allows reducing the duration of the simulation by about 45 seconds. However, the large size of the files downloaded by the client, and the elevated number of hosts involved in the simulation still cause very long execution times. The presence of a static routing system that optimizes the transmission of such large files is essential in this scenario. In fact, a probabilistic routing would cause higher congestion levels in correspondence of channels and routers. In such high congestion conditions, the application might fail to terminate within the maximum simulation time.

The Mobile Agents case shows different behavior, as the execution times of the standard simulator and the SSO simulator are very similar. This is explained by the fact that packet re-transmissions can no longer occur, leading to very similar execution times.

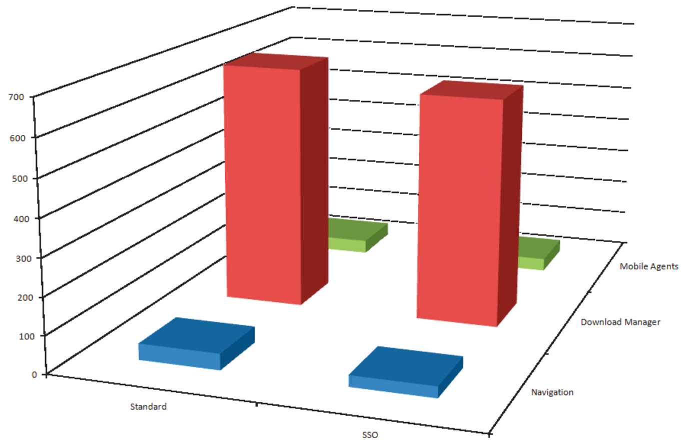


Figure 45. Simulation Results

Figure 45 highlights more clearly the difference between the standard and SSO simulation cases. In light of these results, we can affirm that the activation of a Single Sign-On system is very useful at the application level, since it significantly reduces the number and size of the packets needed to authenticate the client. However, it does not introduce any benefit at the network (IP) level. It is important to note that the underlying networking protocol does not allow the re-transmission of packets. This translates into a reduced number of web application packets flowing in the system. For this reason, the difference between the two simulation cases becomes more subtle.

10. Conclusions

In this work, we applied computer network simulation techniques based on the use of queuing network models to study the behavior of web applications run in a distributed way on a computer network. We considered three types of applications executed according to three different operating paradigms: Navigation, Download Manager and Mobile Agents. For each one of these applications we built two simulators that differed from each other with regard to the client authentication system. The first simulator executed applications by performing authentication procedures locally on each server. On the other hand, the second simulator activated a Single Sign-On server which allowed executing authentication procedures only once for each session. The simulation results show that introducing a Single Sign-On server brings significant benefits to the network, since all the web applications terminate much earlier. In particular, the navigation case is the one that benefits the most from the introduction of the SSO server, as the execution time is reduced by one third.

References

[1] D'Ambrogio A, Iazeolla G and Pasini L 2007 *Simulation Modeling Practice and Theory* **15** 605

[2] Pasini L 2017 *TASK Quart.* **21** (3) 197

[3] Simulog qnap2 reference manual version 9.2.

[4] Pasini L and Feliziani S 2007 *TASK Quart.* **11** (3) 203

[5] Pasini L and Feliziani S 2009 *TASK Quart.* **13** (4) 315