MSc Electronic and Computer Engineering



Computer and Comunication Networks

2019-4-25

University of birmingham

Kai Chen (1948361)

Contents

[1.Section A 2](#_Toc7021488)

[1.1 Abstract 2](#_Toc7021489)

[1.2 Part 1: Describe three congestion control approaches 2](#_Toc7021490)

[1.2.1 Policing 2](#_Toc7021491)

[1.2.2 Traffic shaping 2](#_Toc7021492)

[1.2.3 Real-time Transport Protocol (RTP) 3](#_Toc7021493)

[1.2.4 Reason of choosing these three approaches 3](#_Toc7021494)

[1.3 Part 2: Explain behaviour 4](#_Toc7021495)

[1.3.1 Constant, high-volume traffic conditions 4](#_Toc7021496)

[1.3.2 Overall low-volume traffic with short bursts of high-volume activity 4](#_Toc7021497)

[1.4 Part 3: Approach to management popular video conferencing network 5](#_Toc7021498)

[2. Section B 6](#_Toc7021499)

[2.1 Description of the flooding algorithm implemented, and network topology used 6](#_Toc7021500)

[2.2 Graphical presentation of the results for each node 7](#_Toc7021501)

[2.3 Analysis and discussion of the results 8](#_Toc7021502)

[2.4 Proposed solution for reducing the problem of duplicate packets 9](#_Toc7021503)

[2.5 Conclusion 9](#_Toc7021504)

[2.6 Code 9](#_Toc7021505)

[References 11](#_Toc7021506)

# 1.Section A

## 1.1 Abstract

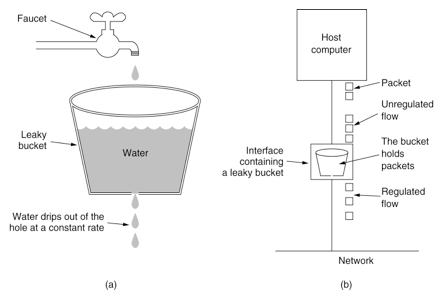
Congestion occurs when the demand for resources exceeds the resources available, the purpose of congestion control is to prevent excessive data from being injected into the network, so that routers or links in the network are not overloaded.

## 1.2 Part 1: Describe three congestion control approaches

Then, I will introduce the following three congestion control approaches which have been introduced in lecture 2, they are policing, traffic shaping and Real-time Transport Protocol (RTP).

### 1.2.1 Policing

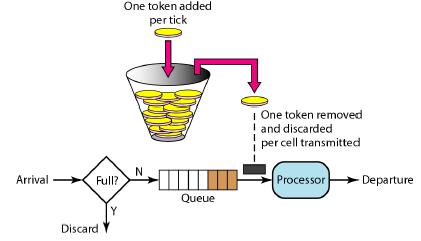
Each API interface has an access limit. When the access frequency or concurrency exceeds its tolerance range, we must consider the current limit to ensure the availability of the interface or degrade the availability. That is, the interface also needs to be installed with a fuse to prevent system embarrassment caused by unexpected requests. The idea of Leaky Bucket algorithm is very simple. The water (request) firstly enters the leaky bucket, and water leak from leaky bucket at a certain speed (interface has a response rate). When the water inflow speed exceeds the limited value (the access frequency exceeds the interface response rate), then reject the request, so we can know that leaky bucket algorithm can force to limit the data transfer rate. The main purpose is to control the rate at which data is injected into the network, smoothing burst traffic on the network. The leaky bucket algorithm provides a mechanism by which burst traffic can be shaped to provide a stable flow to the network. The leaky buckets algorithm can be showed by following picture:

**

*Figure 1 Leaky bucket algorithm*

### 1.2.2 Traffic shaping

Traffic shaping is used to limit the traffic and bursts of a connection that flows out of a network, so that such packets are sent out at a relatively uniform rate. The traffic shaping is usually performed by using a buffer and a token bucket. When the packet is sent too fast, the buffer is first buffered, and the buffered packets are evenly sent under the control of the token bucket. A typical application is to control the output of local traffic based on the TP metrics of downstream network nodes. The main difference between traffic shaping and traffic policing is that traffic shaping caches packets that need to be discarded in traffic policing—usually putting them into a buffer or queue, also known as Traffic Shaping (TS). When the token bucket has enough tokens, the cached packets are sent out evenly. Another difference between traffic shaping and traffic policing is that shaping can increase latency, while regulation introduces little additional latency. The token bucket algorithm can be showed by the following picture:



*Figure 2 Token bucket algorithm*

### 1.2.3 Real-time Transport Protocol (RTP)

Real-Time Transport Protocol (RTP) provides end-to-end delivery services with real-time features such as interactive video audio or simulate data under multicast or unicast network services. Applications typically run RTP over UDP to use their multi-node and checkout services; both offer the functionality of a transport layer protocol. However, RTP can be used with other suitable underlying networks or transport protocols. If the underlying network provides multicast mode, RTP can use the multicast table to transfer data to multiple destinations.

RTP itself does not provide on-time delivery mechanisms or other quality of service (QoS) guarantees, which rely on the underlying services to implement this process. RTP does not guarantee delivery or prevent out-of-order delivery, nor is it reliable for the underlying network. The RTP implements an orderly transmission. The sequence number in the RTP allows the receiver to reassemble the sender's packet sequence, and the sequence number can also be used to determine the appropriate packet location. For example, in video decoding, sequential decoding is not required.

### 1.2.4 Reason of choosing these three approaches

The leaky buckets algorithm belongs to policing: It can be used for fixed packet length protocols, such as ATM, can also be used for variable packet length protocols, such as IP. Besides, Optimizing the use of network resources and improving network utilization are also its advantage.

The token bucket algorithm belongs to traffic shaping: The token bucket algorithm allows idle hosts to accumulate transmission rights so that large data bursts can be sent later. The token bucket holds the token. When the bucket is full, it discards the token and the packet is not discarded.

Multicasting congestion control belongs to RTP: Multicast congestion control can solve the problem of rapid bandwidth consumption and network congestion.

## 1.3 Part 2: Explain behaviour

### 1.3.1 Constant, high-volume traffic conditions

Leaky bucket algorithm based on policing: Firstly, when using a policing-based leaky bucket algorithm to solve constant high-capacity conditions, we must get the speed of high-capacity traffic. If the speed of high-capacity traffic is lower than the constant speed at which the leaky bucket sends packets to the destination, then the leaky bucket algorithm can handle this situation; if the speed of the high-capacity traffic is greater than the constant speed at which the leaky bucket sends the packet to the destination, the data will quickly fill the leaky bucket and cause congestion. When the bucket is full, the subsequent packet will be thrown away. In this case, the leaky bucket algorithm cannot solve this problem.

Token bucket algorithm based on traffic shaping: If the average sending rate configured by the user is r, the token will be added to the bucket every 1 / r seconds. The bucket can hold the b token at most, if the token is full when the bucket is full. When you arrive, discard it. Each token in the token bucket represents one byte. In the case of constant, high-capacity traffic, if the traffic speed r is less than the token generation speed, the bucket deletes the same number of tokens as the traffic packet and transmits normally; if the traffic speed r is greater than the token generation speed, it will occur. Congestion phenomenon. Because every 1 byte of data transmitted will consume 1 token, but the speed r of the transmission is faster than the token generation. When the total amount of tokens b in the bucket is consumed, the packet will be stored in the buffer. Packets in the buffer exceed a certain level, causing network congestion. Therefore, the token bucket algorithm does not solve the constant, high-capacity traffic conditions well.

### 1.3.2 Overall low-volume traffic with short bursts of high-volume activity

Leaky bucket algorithm based on policing: When using the leaky bucket algorithm to resolve low-volume traffic with short high-capacity active bursts, packets are sent from the sender to the destination at a constant rate through the leaky bucket. Since the speed of low-volume traffic is less than the speed at which the leaking bucket sends packets to destination, the system is not congested. But when high-volume activities break out, the burst speed is much faster than the constant speed. As a result, sudden high-volume activities will quickly fill the bucket and cause network congestion. Therefore, the leaky bucket algorithm does not solve the overall low-volume traffic of a large number of activities in a short time.

Token bucket algorithm based on traffic shaping: Suppose the bucket has a capacity of b and generates a token every 1 / r second. Each token in the token bucket represents one byte. When low traffic is used, if the source wants to send n packets, the bucket must delete n tokens and then transmit the packets, in addition to which the system will still generate a new token to fill the token bucket. When the high-capacity activity breaks out, if the speed of the high-capacity activity is lower than the filling speed, the bucket must delete the same number of tokens and then propagate normally; if the size of the high-capacity activity is greater than the token filling speed, the system will delete the excess in the bucket. The number of tokens and transmits the same size as the current number of tokens, and then stores the remaining packets into the buffer. As long as the token exists in the token bucket, the data is allowed to be transmitted in bursts until the user configuration threshold is reached.

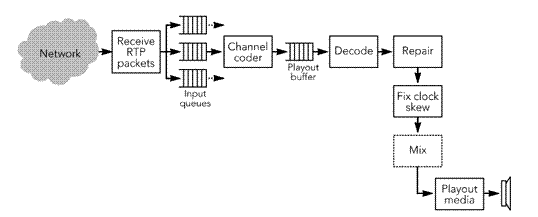
## 1.4 Part 3: Approach to management popular video conferencing network

When using policing to solve the popular video conferencing network, if short burst high-volume traffic occurs, all packets will be discarded, which will influence our project.

When using traffic shaping to solve video conferencing network, the token bucket algorithm and the leaky bucket algorithm are used together for ensuring that the data is transmitted at a constant rate in each node and that no congestion occurs when burst high-volume traffic happen. The video conferencing network needs the real-time transmission of packets to related node. When the speed of data transmitting is faster than the speed of token generation, the data will be stored in the packet buffer, then they cannot be transmitted to the destination in real time.

In summary, I suppose that multicasting congestion control based on RTP is most appropriate to manage video conferencing network. Besides, I will analyse how multicast congestion control to solve congestion problem from the video conferencing network by the following two aspects:

Real-Time Transport Protocol (RTP) provides end-to-end delivery services with real-time features such as interactive video audio or simulate data under multicast or unicast network services. RTP itself does not provide on-time delivery mechanisms or other quality of service (QoS) guarantees, which rely on the underlying services to implement this process. RTP does not guarantee delivery or prevent out-of-order delivery, nor is it reliable for the underlying network. The RTP implements an orderly transmission. The sequence number in the RTP allows the receiver to reassemble the sender's packet sequence, and the sequence number can also be used to determine the appropriate packet location. For example, in video decoding, sequential decoding is not required.



*Figure 3 Block diagram of the receiving process*

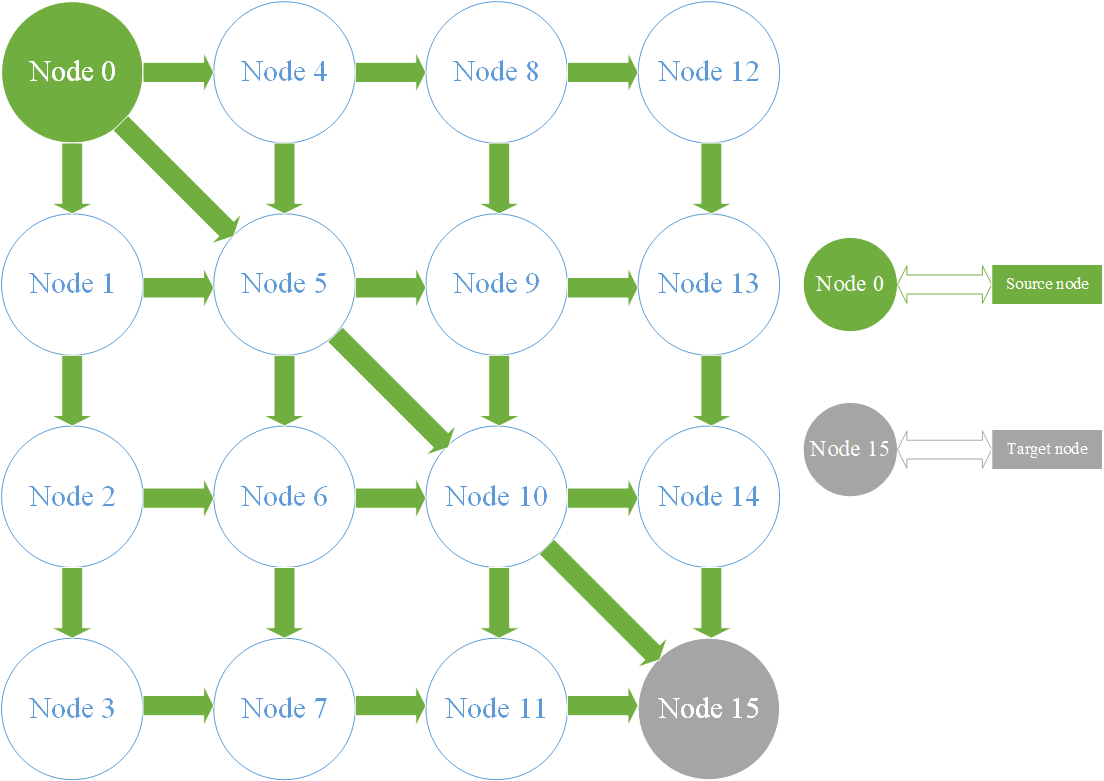
UDP+RTP transmission mode is RTP data encapsulation transmission through UDP packets. The simple UDP transmission mode cannot be directly applied to the development of video conferencing software. Although UDP can transmit data in real time, its UDP transmission is based on connectionless and unreliable. The transmission mode, the data could not be guaranteed to arrive in time and in order, so it is difficult to control the packet loss of data directly by using the UDP transmission method. Through the transmission mode of UDP+RTP, not only real-time transmission, but also data transmission control through RTCP. We can give an example. In the audio transmission of video conferencing software, we can transmit in the UDP+RTP mode. When there is a loss of data packets, this method can easily detect the lost data and thus process error masking for data restoration by encoding the audio.

# 2. Section B

## 2.1 Description of the flooding algorithm implemented, and network topology used

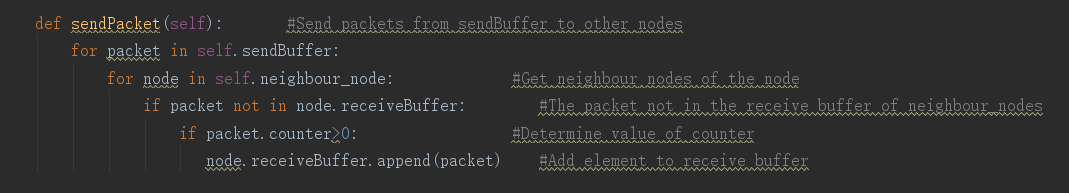
Flood routing algorithm is a simple and effective routing algorithm. The basic idea is that each node broadcasts the received data packet by other nodes. The flooding protocol will cause the data packets to be spread around the source node.

In this task, I designed the topology to be 4\*4 size, using python code to implement the flooding algorithm, which contains 16 nodes, node0 is the source node, and node15 is the target node. In the beginning, packet\_1 carries the message: "Practical python". It is stored in a packet in the send buffer of the source node. Besides, the counter is designed. When the packet passes through a node, the counter is decremented by 1. When the counter is reduced to zero, the packet will be discarded. The following figure show the basic information and structure of the network topology.



*Figure 4 Topology network structure*

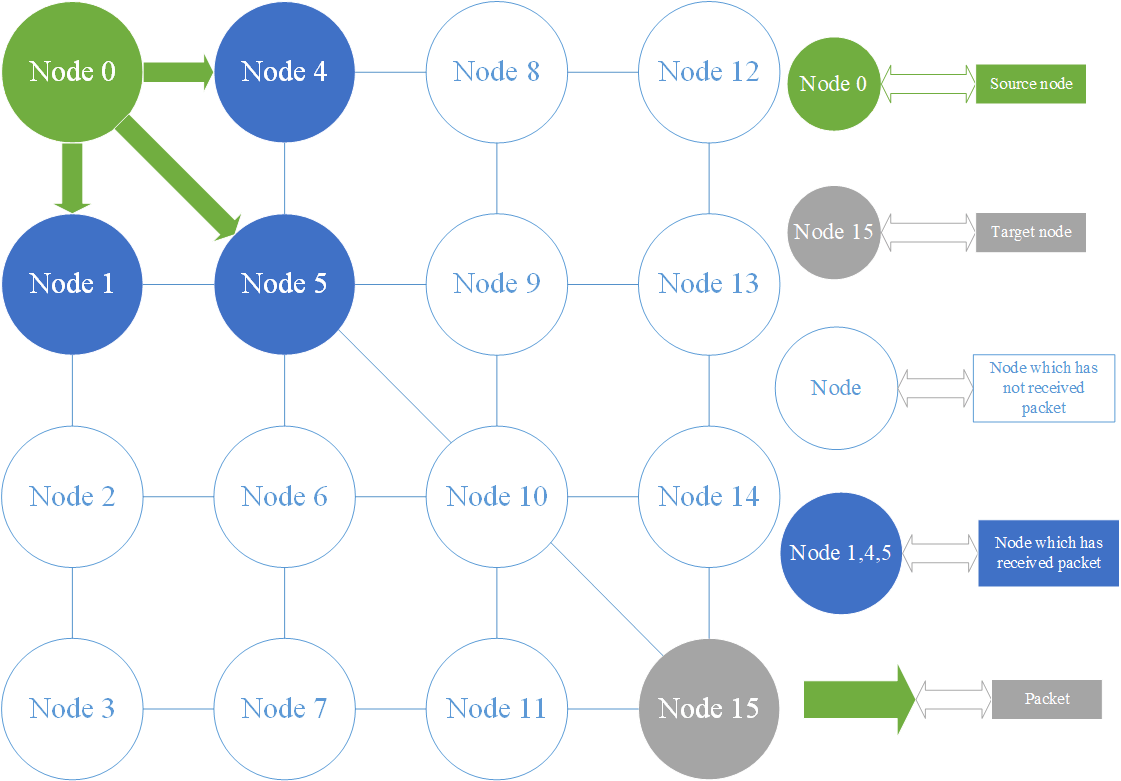
In the program of this job, all nodes except the source node and the target node have a receive buffer and a send buffer. The receive buffer is used to store packets sent by neighbouring nodes. The send buffer is used to send packets to the neighbouring buffer. When packet\_1 is sent for the first time, the source node's send buffer gets command to send packet to the neighbour node of the source node: node1, 4, 5. Therefore, the message "Practical python" is stored in the receive buffer of the three nodes. Besides, the counter has been reduced from 10 to 9. When packet\_1 is sent a second time, node1, 4, 5 will detect if the current node is the target node. If it is a normal node and the counter is greater than zero, the receive buffer sends the packet to the transmit buffer. After receiving the send command, the packet will be sent again to the receive buffer of the neighbour node of the above node. Therefore, the receive buffer of node2, 5, 6, 8, 9, 10 will store the packet. At the same time, the counter is decremented again. At this point, the packets from the source node to node 1 and node 5 are the same as the packets to node 4 and node 5, resulting in duplicate packets. In my code, a solution has been proposed:



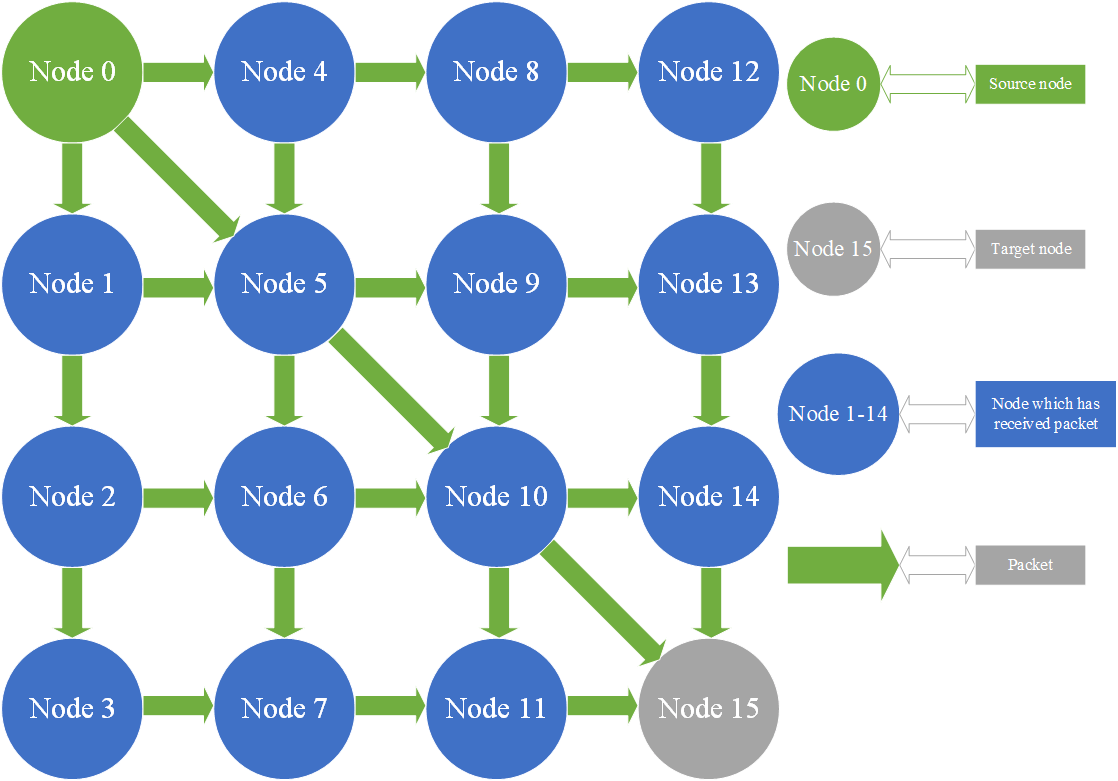
*Figure 5 Duplicated packet problem*

Even if the same packet has a different transmission path, the time to reach the destination node will be different. Therefore, even if the data packet from the source node to the node 1, 5 is the same as the data packet from the source node to the node 4, 5, the transmission path is different and the arrival time is also different. Before the arrival of the packet, it is determined whether the receiving buffer of the destination node has received the same packet, and then decides whether to send the packet to the receiving buffer of the destination node. If it does not exist, the packet will be stored in the receive buffer. Thus, only one packet is stored in the receive buffer of the target node. When packet\_1 is sent for the third time, node2,5,6,8,9,10 will detect if the current node is a destination node. If not, and the counter is greater than zero, the packet is sent to the send buffer. Once the send command is received, the packet is sent to the receive buffer of the neighbouring node. The receive buffer of node3,6,7,9,10,11,12,13,14,15 will store the packet. Besides, the counter is reduced to 7 again. Besides, when packet is moved at the fourth time and sixth time, duplicate packet will occur at node 10,15. We can use above method to solve it again.

## 2.2 Graphical presentation of the results for each node



*Figure 6 Presentation of node when first transmission*



*Figure 7 Presentation of node when third transmission*

## 2.3 Analysis and discussion of the results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Source Node | Next Node | Packet message | Target Node | Counter |
| 0 | 1 | Practical python | 15 | 9 |
| 0 | 4 | Practical python | 15 | 9 |
| 0 | 5 | Practical python | 15 | 9 |

*Table 1 The First Time Transmission*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| The Third Time Transmission | | | | |
| 2 | 3 | Practical python | 15 | 7 |
| 2 | 6 | Practical python | 15 | 7 |
| 5 | 6 | Practical python | 15 | 7 |
| 5 | 9 | Practical python | 15 | 7 |
| 5 | 10 | Practical python | 15 | 7 |
| 6 | 7 | Practical python | 15 | 7 |
| 6 | 10 | Practical python | 15 | 7 |
| 8 | 9 | Practical python | 15 | 7 |
| 8 | 12 | Practical python | 15 | 7 |
| 9 | 10 | Practical python | 15 | 7 |
| 9 | 13 | Practical python | 15 | 7 |
| 10 | 11 | Practical python | 15 | 7 |
| 10 | 14 | Practical python | 15 | 7 |
| 10 | 15 | Practical python | 15 | 7 |

*Table 2 The Third Time Transmission*

The last line of the table represents the shortest path to the destination node (0-5-10-15). After three transmissions, the message " Practical python " was sent to the receive buffer of Target node. Besides, as the number of transmissions increases, the number of repeated packets increases.

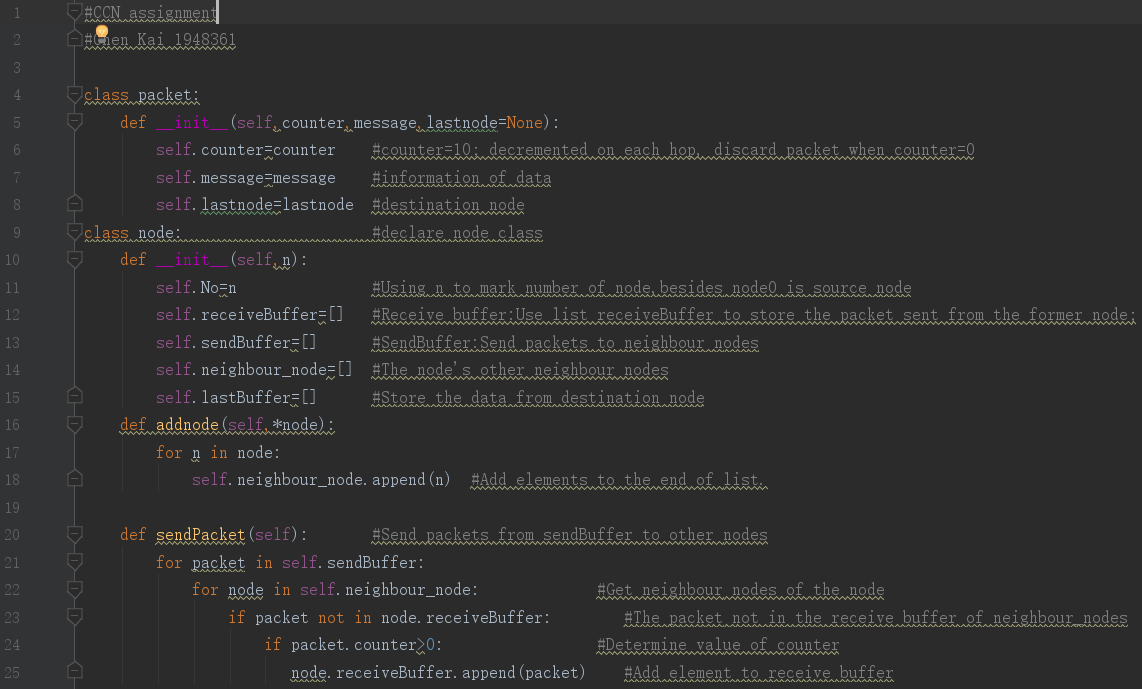
## 2.4 Proposed solution for reducing the problem of duplicate packets

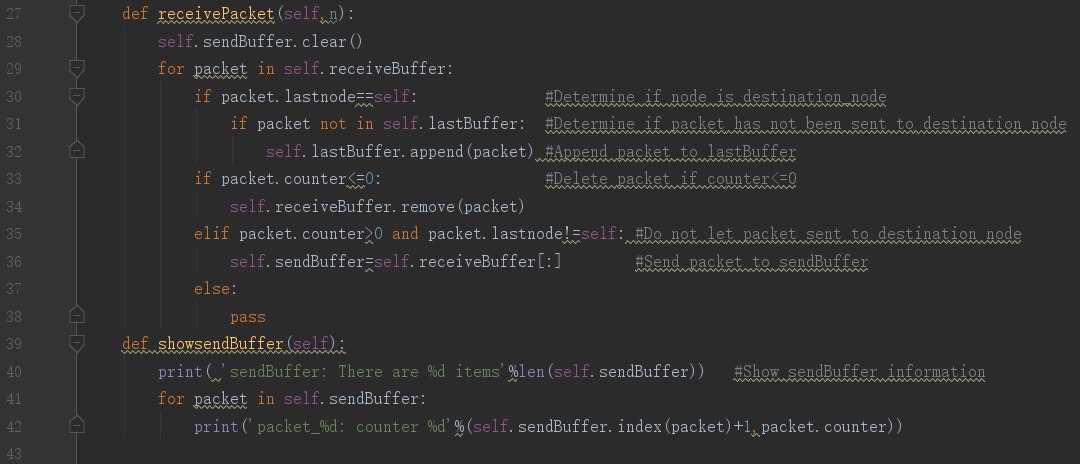
Often duplicate packets can cause network monitoring problems which are connected to ports. Duplicate packets will influence bandwidth, waste processing power, all of that will reduce effectiveness. Some solutions to duplicate packet problems, such as duplicate packet deletion on monitoring tools, are less than ideal. The network monitoring switch is the best solution to solve the problem of duplicate packets. In addition to addressing connectivity issues by allowing multiple types of monitoring tools to connect to TAP or SPAN port, advanced monitoring network monitoring switches can remove duplicate packets at high rates before problems occur. The most important feature of the deduplication feature in network monitoring switches is deduplication window’s size. This is time which a duplicate packet can be detected and deleted after seeing the first copy of the packet. The large deduplication window and the ability to decide size of window make deduplication very powerful.

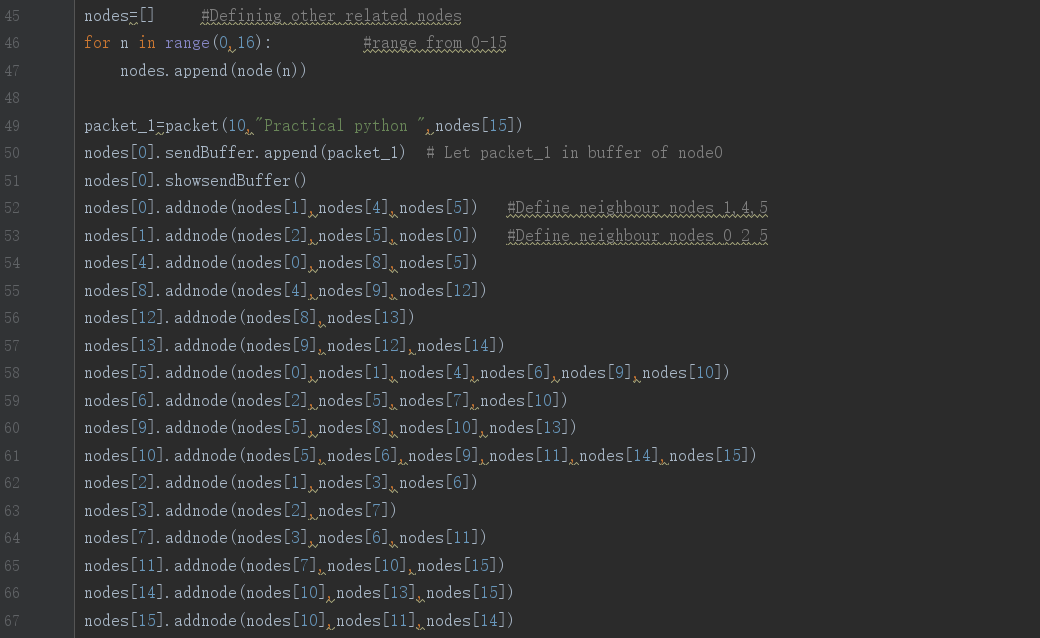
## 2.5 Conclusion

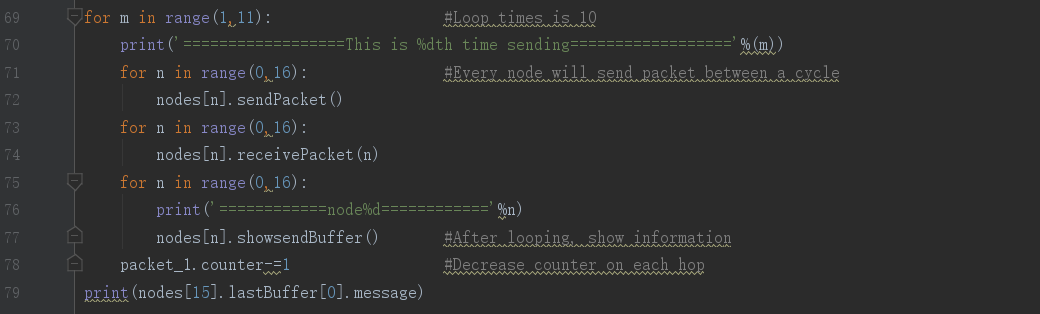
Solving duplicate packets with a network monitoring switch can reduce the monitoring tool processing burden. Performing deduplication with a network monitoring switch means that duplicate packets no longer reduce the bandwidth of monitoring tool. This idea also can eliminate data clutter.

## 2.6 Code









# References

[1]Subramanya, S.R., Mingsheng Peng, Sarangapani, J.. Rate-based end-to-end congestion control of multimedia traffic in packet switched networks[P]. Information Technology: Coding and Computing [Computers and Communications], 2003. Proceedings. ITCC 2003. International Conference on,2003.

[2]Subramanyan, S.R., Jagannathan, S., Mingsheng Peng. A scheme for fair, rate-based end-to-end congestion control of multimedia traffic in packet switched networks[P]. Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on,2003.

[3]Heungsoon Park, Hoseok Jang, Taewook Kwon. Popularity-based congestion control in named data networking[P]. Ubiquitous and Future Networks (ICUFN), 2014 Sixth International Conf on,2014.

[4]THIPKHAMPHANH PHETNOUVONG. Research on Greedy Routing Protocol for Inter-Vehicle Communication[D].湖南大学,2017.

[5]Syed Afsar Shah, Babar Nazir, Imran Ali Khan. Congestion control algorithms in wireless sensor networks: Trends and opportunities[J]. Journal of King Saud University - Computer and Information Sciences,2015.

[6]Aloizio P. Silva, Scott Burleigh, Celso M. Hirata, Katia Obraczka. A survey on congestion control for delay and disruption tolerant networks[J]. Ad Hoc Networks,2015,25.