CS 744 - Management Issues in Software Engineering

Spring 2018

A Visual Tool for Monitoring Network Communication

1 Overview

This project aims at developing a visual tool to monitor network communication. Credit card processing has been chosen as the application domain and hence communications based on credit card transactions will be modeled in this project.

A credit card company such as VISA or MasterCard issues thousands of credit cards to its customers. The company generally has a processing center where it keeps the database of all credit cards issued and their current details (balance, card limit, customers personal details and so on). Customers may use these cards anywhere in the world. For simplicity of this project, the card usage is restricted to a closed region. Figure 1 shows a simplified architecture of the network intended to be implemented.

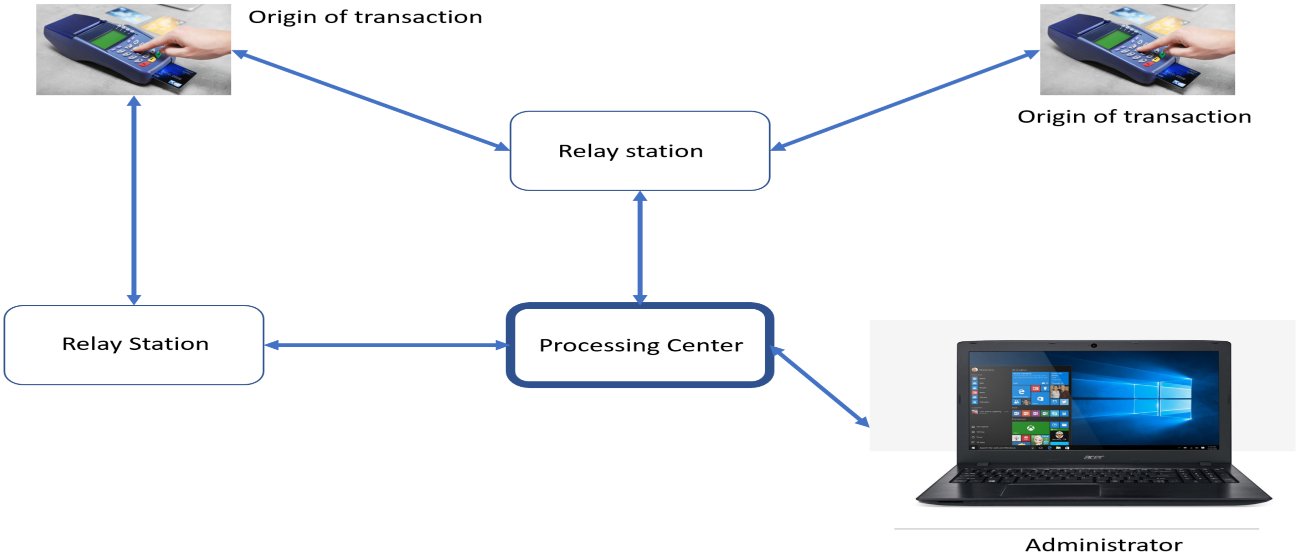

Figure 1: Architecture of the credit card network

When a customer uses the credit card in a location[1], the transaction details (card details, location IP address, date and time) are sent to one of the relay stations to which the store is directly attached. This relay station then sends (actually 'relays', meaning it does not process/alter the information) the information to the processing center. At the processing center, the transaction details are processed and a response is sent back to the store indicating the approval or denial of the transaction. Notice that there could be several relay stations between the store and the processing center. A relay station actually may receive several thousands of these transactions, and so it may queue the transactions first before forwarding to the processing center (or to the next relay station). Further, a relay station needs to find the shortest path to the processing center when it relays the transaction. This path may vary sometimes because another relay station on the shortest path may be inactive/down. Figure 2 shows an areal view of the entire network. In this figure, the rounded rectangle represents the processing center, rectangles indicate relay stations and circles indicate stores.

[1] A store, restaurant or business - shown as"origin of transaction" in Figure 1. For convenience, we use the term 'store' hereafter to represent the origin of transaction.

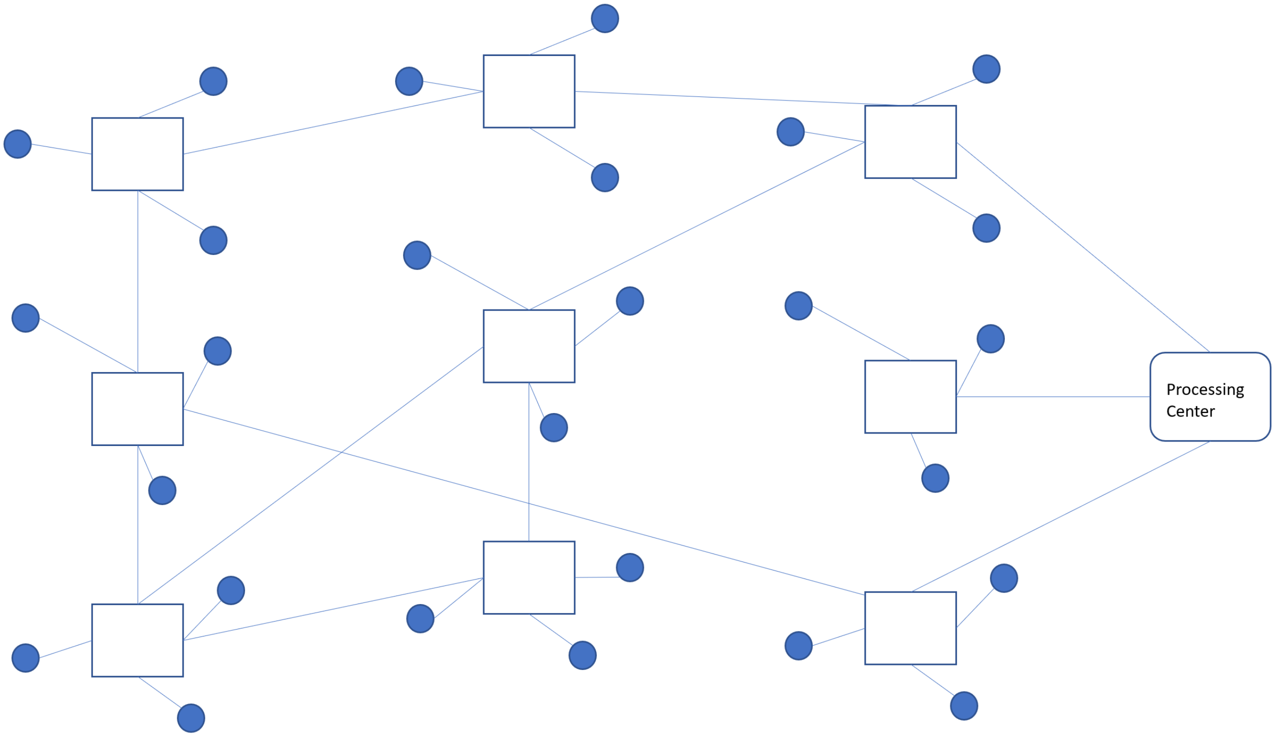

Figure 2: An areal view of the credit card network

2 Implementation of the network

The rest of the document describes the details of the credit card network implementation and communication protocol to be used.

2.1 Processing Center

The processing center hosts the database of all credit cards and also supports administrative tasks. Some of the administrative tasks such as approving/denying a credit card transaction will be done automatically by the software while other tasks such as adding/deleting/modifying a credit card's information must be done manually by an administrator.

2.1.1 Database

The database must store all credit cards and transactions. Each credit card must include the following details:

- A unique 16-digits credit card number. This number must not start with a zero or nine. Those two digits are reserved for certain applications. The project team is strongly encouraged to research on credit card numbers; some of these details are given at `http://www.cardrates.com/advice/7-things-you-didnt-know-about-credit-card-numbers/`
- Customer's first name and last name. The length of each name must be between 2 to 15, inclusive.
- Expiration date, in the format MM/YYYY.
- A 3-digits security code. The first digit must not be a zero.

A separate account must be maintained for each credit card. The details for this account include the following:

- Credit card information as defined above.
- Account holder's first name and last name. The length of each name must be between 2 to 15, inclusive. The account holder's name may be different from the name on the card; the latter shows who is using the card whereas the account holder is responsible for paying the bills.
- Address of the account holder. Use US address format (building number, street name, apartment number (optional), city, state and ZIP code). An example would be "187 North Main Street Apt.4, La Crosse, WI 54601."
- Phone number of the account holder. Use the format (XXX) XXX-XXXX. Example: (608) 785-8012.
- Maximum/Limit of spending allowed (should be greater than zero and less than or equal to $25,000.)
- Current balance.
- A list of transactions on this account. Each transaction must include the following details:
  - A unique transaction ID (a 10-digits number). The first digit must not be zero.
  - Date and time of transaction that represents the instant of origination of the transaction. The date and time (treated as one entity) can be stored internally using the ISO-8601 standard (see the web site `https://en.wikipedia.org/wiki/ISO_8601/`). When it is displayed on the GUI, it should be displayed in the format MMM DD, YYYY HH:MM. HH:MM uses 24-hour clock. An example: Feb 23, 2018 14:32.
  - Amount.
  - Credit (increase the balance on the card) or debit (decrease the balance on the card).
  - Name of the merchant, if credit. For debit, it can be the name of the merchant if the customer is returning something or 'SELF' if the customer is paying the bill. See the section "Store" below for details on the format for merchant's name.
  - Status - APPROVED or DENIED and its date and time. This 'date and time' represents the instant of the response created for the transaction. The date and time should be internally stored in ISO-8601 standard but should be displayed in the MMM DD, YYYY HH:MM format.

2.1.2 Administrator Tasks

The administrative tasks can be performed by one or more administrators in the system.

Login

An administrator account must have a unique username (must be at least six characters long) and a password (must be at least six characters long). The project team is encouraged to use its own additional restrictions on usernames and passwords to make the system more secure and realistic. To log in, the user must provide the username and password and must also answer to a security question. There should be three security questions associated with each account and one of them will be randomly displayed at the time of login. These questions will be selected by the user and the corresponding answers are also stored at the time when the account is created. If the user does not answer to the first security question, the second and third questions should be displayed in random order. If the user does not answer to any of the three security questions correctly, the user's account should be blocked. The decision on how to release the account is left to the project team to decide.

For this project, there is no need to include any functionality to add, delete or modify users' (administrators') accounts. These tasks can be done outside the scope of this project.

Visualize a credit card transaction

An administrator must be able to view the processing of a credit card transaction. This includes the display of the entire network. The display must show where a transaction originates, how it is passing through the relay stations before reaching the processing center and how a response from the processing center reaches back to the store. For the purposes of this project, the administrator must manually originate a transaction from one of the stores (the blue circles in Figure 2). This will be a two-step process - create a transaction, and encrypt it before it is sent. An administrator should be able to manually create a transaction and should also be able to encrypt it. The project is therefore using this visual tool to test the network, and does not represent the actual network itself.

When a transaction (or its response) is in flow, the corresponding nodes in the path of the transaction must be highlighted showing an animation of the communication. The project team should choose a reasonable time delay (a few seconds should work) between the nodes in the graph so that the animation can be viewed/demonstrated. The administrator should be able to stop (and restart) the flow at any station in between the store and the processing center to observe the details of communication. For example, the credit card information should be encrypted during the flow. So, an administrator may want to ensure that it is properly encrypted by stopping the flow at any instant and observing the details. Notice that the administrator should be able to do the same thing for a response sent from the processing center to the store.

For encryption and decryption of transactions, the project team has the freedom to choose any method. The team members are strongly encouraged to do some research work on information security and find a suitable method.

Activate or Inactivate a relay station

An administrator should be able to make a relay station ACTIVE or INACTIVE. See the next section for more details on the status of a relay station.

Activate or Inactivate a connection

An administrator should be able to make connection between any two nodes (store, relay station, processing center) ACTIVE or INACTIVE. For example, a connection between a store and a relay station can be made inactive. If the store is connected to another relay station, the store can still send transactions (and receive responses) through the other active connections.

Observe a relay station

An administrator should be able to observe the internal details of a relay station. If selected, this operation should display the information such as ID of the station, its current status (ACTIVE or INACTIVE), and the queue of transactions at that time.

2.1.3 Processing of a Transaction

The processing of a transaction is a two-step process: decrypt the transaction and create a response for the transaction. A transaction will be approved only if it meets all the conditions stated below:

- The credit card details match with one of the entries in the database.
- The card has not expired.
- The account has enough balance for the amount mentioned in the transaction.
- The merchant's name and ID match with one of the entries in the database[2].

[2] Every store has a merchant name and a unique ID (see the section "Store" for details). The database must have a list of all store names and their IDs. Only transactions originating from these stores are considered to be valid.

The processing center will send a response to the store after the transaction is processed. The response will include the transaction ID, its status (approved or denied), the date and time at which the response is created and the store ID. A response should be encrypted before it is sent.

2.2 Relay Station

A relay station has a unique ID. In the real world, this is the IP address of the computing resource at that station. To make it more realistic, we will also represent the ID number of a relay station in the IP address format: XXX.XXX.XXX.XXX. However, to make it simple, let us have the first three parts of the IP address as 192.168.0 so that every node ID in the network (including relay stations, stores and processing center) will have the same prefix except for the last three digits. These three digits must be in the range of one to 254. The numbers zero and 255 are generally used for some reserved operations.

A relay station will receive transactions from the stores attached to it or from other relay stations connected to it. It will also receive responses from the processing center or from other relay stations. Since the purpose of a relay station is simply to forward/relay, it does not distinguish between a transaction (the one going from a store to the processing center) and a response (the one going from the processing center to the store). The relay station queues all incoming transactions and responses, and dispatches them one at a time. For simplicity, we will use First-In-First-Out queue. Except for the ID of the store and that of the processing center, the rest of the transaction/response details must be encrypted. Therefore, a relay station will only see the source and destination IDs of the transaction/response. The job of a relay station is to find the shortest path between itself and the destination, and forward the transaction/response to the first node in the shortest path. The project team has the freedom of choosing any shortest path algorithm for this purpose. Notice that the network is an undirected but weighted graph (see Figure 2).

There are two status associated with a relay station - ACTIVE and INACTIVE. A relay station will be functional when it is ACTIVE, and is non-functional (meaning no transaction/response will pass through the relay station) when it becomes INACTIVE. An administrator should be able to make a relay station ACTIVE or INACTIVE manually.

2.3 Store

A store has a merchant's name which will be used in every transaction originating from that store. Merchant's name is a string with the length in the range of 2 to 30 characters. A store also has a unique ID which is represented in the same format of an IP address. A store can originate a transaction and receive a response back from the processing center. It is a synchronous operation meaning that every transaction will be complete only if it receives a response from the processing center. A store must be connected to at least one relay station; it may be connected to more than one relay station. No store is directly connected to the processing center.

3 Deployment

The project must be implemented as a web application. Each team will be given a separate server. The team is required to launch the product from the web server assigned to that team.

3.1 Configuration

The network must consist of exactly one processing center, at least 10 relay stations and at least 30 stores. The project team has the option of choosing the connections between the stores and the relay stations and those between relay stations. All relay stations will be ACTIVE initially. The processing center and all stores will be ACTIVE all the time; they will never become INACTIVE. Initially, all connections are also ACTIVE.

For the final project demo, the instructor will give the configuration of the network. Therefore, it is strongly recommended NOT to have a fixed network configuration. Instead, the team should be able to redefine the network and its connections at any time.