The FireSim program allows groups of students and an instructor to practice configuring a simulated firewall in an interesting and interactive manner.

There are multiple HTML and PHP programs implementing FireSim. The student pages use the session variables groupID and player to keep track of the current player and their group.

# Database

config – Holds the firewall configurations of the players. Each line of a player's firewall configuration creates a row in the config table.

groupID varchar(16) group ID of the player

player varchar(16) player's name

seq int(11) sequence number to keep config lines in order.

ace varchar(256) flattened ACE object defining the configuration

primary key (groupID,player,seq)

games – Table of all the groupIDs (and therefore simulation games) in the system.

groupID varchar(16) group ID of this session

unique primary key (groupID),

message – Table of all messages. A message is for a specific player. Broadcasts create individual messages for each player. The message type gives the purpose of the message.

groupID varchar(16) group ID of the player to receive the message

player varchar(16) name of the player to receive the message

text varchar(256) text of the message

type int(11) message type with the values:

1 = message to be displayed on the player's screen

2 = player has been successfully attacked

3 = player has defended against an attack

4 = player's attack was successful

5 = player's attach was unsuccessful

6 = configuration was correctly updated

7 = configuration error

8 = new firewall requirement

9 = start of the simulation

10 = end of the simulation

players – Table of all players and their scores.

player varchar(16) name of the player

groupID varchar(16) group ID of the player

score int(11) current score of this player, initially 100

primary key (player,groupID)

# Login

playerLogin.html

Initial login page for a player. Invokes playerStart.php

playerStart.php

Invoked by playerLogin.html to initially register the player with the simulation. It checks for valid group and name. If all is well, it puts the player in the database players table and sends playerhome.html to the player. The session variables groupID and player are created here.

adminLogin.html

Initial login page for the administrator. Invokes adminStart.php

adminStart.php

Invoked by adminLogin.html to initially start a new simulation game. After checking for a valid group, it puts the groupID into the games database table. It sends adminhome.html to the administrator.

# Home pages

playerhome.html

The player's home page invoked by playerStart.php. A player stays on this page for the duration of the simulation.

adminhome.html

The administrator's home page invoked by adminStart.php. The administrator stays on this page for the duration of the simulation.

# Firewall configuration and attack

ACE.php

**A**ccess **C**ontrol **E**ntity holding the internal format for one line of a player's firewall configuration. The ACE objects are created by checkConfig.php which calls the ACE parse method. The parse method takes a line of firewall configuration text and converts it to the internal format. The tranCheck method checks if a simulated transmission would be impacted by this access control entity.

This object is stored in the config database table. It is flattened and reconstituted using the serialize and initialize methods of this file. This is problematic with binary values with a leading one bit. It might be better to store this thing in the database as a Blob or something else.

checkConfig.php

Invoked by playerhome.html to check a player's firewall configuration. It creates ACE objects for each line of the configuration and stores them in the config database table.

Action.php

Object to hold the information about an action one player can take against another. These objects are created in defActionList.php and referenced by go4Blook.php.

defActionList.php

Defines the actionList array of Action objects which details the parameters of all possible actions one player can take against another.

go4Blood.php

Invoked by playerhome.html when the player wants to take an action against another player. It uses ACE and Action objects to determine the outcome of an action. Messages are sent to the players indicating the result of the action. The scores are updated in the players database table if appropriate.

# Utility programs

refreshPlayers.php

Invoked by playerhome.html and adminhome.html to refresh the list of players. It sends a JSON formatted list to the requester containing the names and scores of all the players.

adminSendMsg.php

Called from adminHome.html when the administrator wants to send a message to one or all of the players.

retrieveMsg.php

Invoked periodically by playerhome.html to get all of the messages sent to this player. Messages are taken from the message database table and deleted after they are sent to the player.

# Online player documentation

FireSimHelp.html

Documentation for the player on how to use FireSim. It gives a list of port numbers.

Config.html

Displays a diagram of the simulated network, config.JPG, giving the IP addresses of the computers in the simulate network.

# Error messages

usedgroupID.html

Error message displayed when the administrator enters a group ID that exists in the database.

Sent by adminStart.php

usedname.html

Error message displayed when a student enters a name that exists in the database. Sent by playerStart.php

nogroupID.html

Error message displayed when a student enters a group ID not found in the database. Sent by playerStart.php

badgroupID.html

Error message displayed when an administrator enters a group ID that does not pass the validity check. Sent by adminStart.php to avoid security problems such as cross site scripting.

badname.html

Error message displayed when a student enters a name that does not pass the validity check. Sent by playerStart.php to avoid security problems such as cross site scripting.