AMRITA
VISHWA VIDYAPEETHAM
UNIVERSITY

CASE STUDY
WORKING OF THE INTERNET

COURSE CODE : 19CSE214

COURSE NAME: THEORY OF COMPUTATION

TEAM MEMBERS :

S.No	Name of the Student	Roll No.
1.	RAVELLA ABHINAV	CB.EN.U4CSE19453
2.	PENUGONDA KOUSHIK	CB.EN.U4CSE19449
3.	SINGADI SHANTHAN REDDY	CB.EN.U4CSE19459
4.	NUSUM KARTHIK	CB.EN.U4CSE19444

Problem Statement:

The Internet is a very large network of computers connected to each other serving and receiving billions of gigabytes of data every day. For this modern wonder to work, there are many background activities happening that are ensuring the smooth and safe operation of the internet. This is all abstracted out from the point of view of regular users. The aim of this case study is to examine some of the processes involved in the functioning of the Internet and apply concepts from Theory of Computation to it.

Scenarios:

1) The connection

2) Generating a HTTP request

3) The Firewall

4) Domain Name Resolution

5) The complete process

The connection

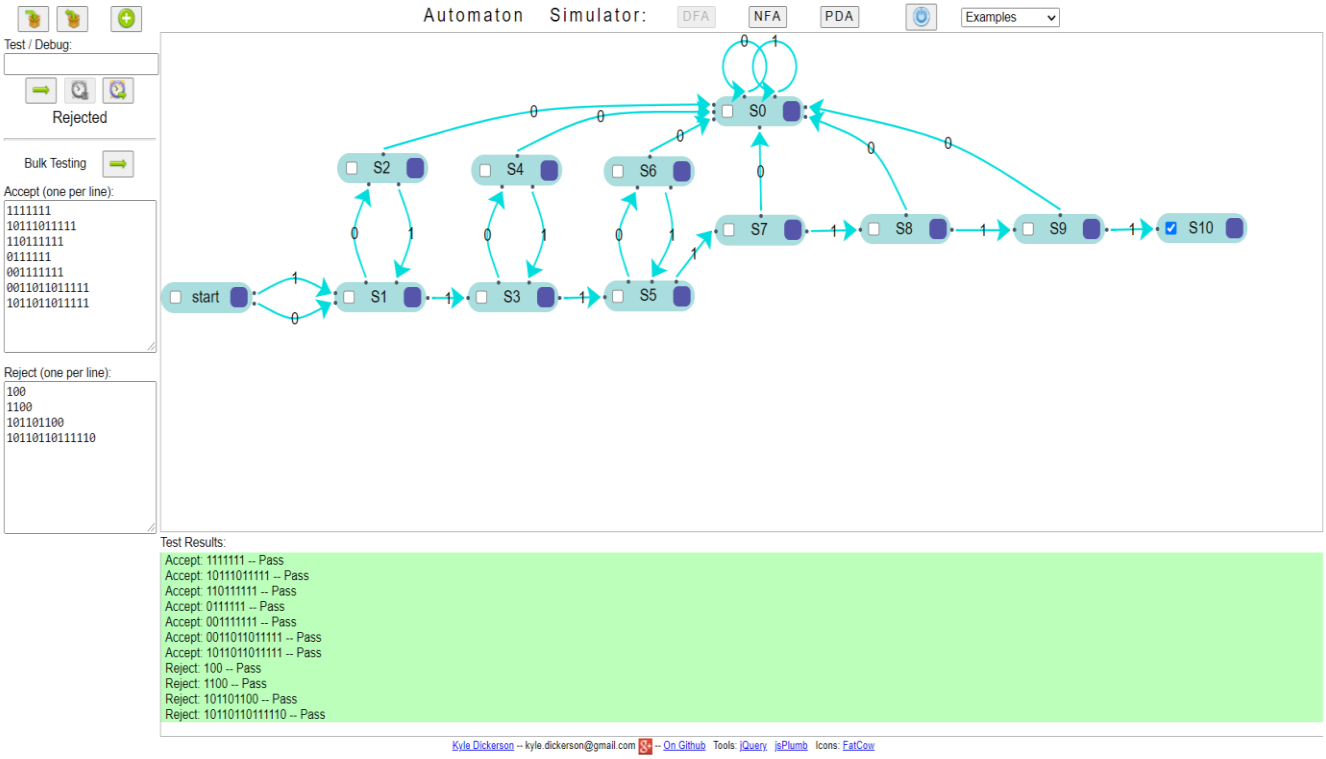
Connecting to and using the Internet involves a combination of hardware components and protocols that ensure that there is a standard format for the data that is being sent over it. The process begins with the NIC. NIC (Network Interface Controller) is a hardware interface card which is the part of the computer that communicates with the router when a request has been generated.

The request first reaches the router from the NIC. If the receiver is on the same Local Area Network (LAN) the router routes the request to that device. If the receiver is on another local area network, then the request reaches the routing switch from where it is forwarded to the DNS server. This system of servers is responsible for translating the domain name of the website specified in the URL into a valid numerical IP address that can be identified by computers on the network.

Once the DNS server has identified the IP address of the domain, the servers need to create a secure line for the transfer of packets. This is done to ensure that other services/middle men cannot access the same packets. The protocols involved here are TCP and TLS. The TCP handshake opens a connection between the client and the sever. The TLS handshake encrypts the connection to make sure that middlemen cannot

access unencrypted data. Packets are then delivered to the destination server through undersea cabling or satellite communication, and are processed at the endpoint. If the packets are lost somewhere in the middle then the replacement packets will be generated by the client that initiated the request.

DFA:



States and Transition rules:

S1 - NIC – It is a state where NIC presence and transmitting is checked.

- **Output '1':** If NIC is present and request to send to server is ready. Next state is S3
- **Output '0':** If there is Issue either with NIC or communicating with router. Next state is S2

S2 - NIC troubleshoot – It is a state where troubleshooting takes place.

- **Output '1':** If troubleshooting resolves issue with NIC. Next state is S1
- **Output '0':** If troubleshooting fails goes to TRAP STATE(S0) I.e., fails to connect and request will not be processed.

S3 - Router – Here checking for presence and working of the router is done.

- **Output '1':** Router is working fine and ready to transmit. Next state is S5
- **Output '0':** Any issues with router then goes to S4

S4 - Router troubleshoot – This state deals with troubleshooting of router.

- **Output '1':** If troubleshooting resolves issue with router. Next state is S3
- **Output '0':** If troubleshooting fails goes to TRAP STATE(S0) I.e., fails to connect and request will not be processed.

S5 - Router Switch – It checks for connectivity with DNS server.

- **Output '1':** Router switch connectivity is fine with DNS server. Next state is S7
- **Output '0':** If router switch has any issues it goes to S6

S6 - Router Switch troubleshoot – Troubleshoots router switch's connectivity.

- **Output '1':** If troubleshooting resolves issue with router switch. Next state is S5
- **Output '0':** If troubleshooting fails goes to TRAP STATE(S0) I.e., fails to connect and request will not be processed.

S7 - DNS server – This state gets IP address from the URL.

- **Output '1':** If there is an IP in the DNS server for the requested URL. Next state is S8
- **Output '0':** If not found it goes to TRAP STATE(S0) I.e., fails to connect and request will not be processed.

S8 - IP lookup – Here it checks of IP address related with the domain and finds if it actually exists.

- **Output '1':** If IP address is located on the internet. Next state is S9
- **Output '0':** If not found it goes to TRAP STATE(S0) I.e., fails to connect and request will not be processed.

S9 - Satellite/ Undersea cables – This state checks for connectivity with and availability of servers.

- **Output '1':** If connection is established with the found IP address. Next state is S10
- **Output '0':** If connection is failed then it goes to TRAP STATE(S0) I.e., fails to connect and request will not be processed.

S10 - (FINAL State) Server – Here the request generated by the client will be processed.

Regular Grammar:

Right linear grammar: (Intermediate stages)

- Start -> 0S1
- Start -> 1S1
- S1 -> 0S2
- S2 -> 0S0
- S2 -> 1S1
- S1 -> 1S3
- S3 -> 0S4
- S4 -> 0S0
- S4 -> 1S3
- S3 -> 1S5
- S5 -> 0S6
- S6 -> 0S0
- S6 -> 1S5
- S5 -> 1S7
- S7 -> 0S0
- S7 -> 1S8
- S8 -> 0S0
- S8 -> 1S9
- S9 -> 0S0
- S9 -> 1S10
- S10 -> λ

Regular language:

- 1111111
- 10111011111
- 110111111
- 0111111
- 001111111
- 0011011011111
- 1011011011111

Generating a HTTP request

Once the user wants something from the Internet, the browser generates a HTTP request to get that data from the server concerned. This request has different parameters in it, some of which are mentioned here:

This request has different parameters in it, some of which are mentioned here:

- **Verb – (GET/POST/PUT/Patch/Delete)**

The verb is the part of the request that decides the purpose of the request being sent. It is used to inform the receiver of the request the type of request it is getting and the service expected.

- **Target URL (Uniform Resource Locator)**

The target URL specifies the location of the requested file on the server from where information is being requested. This is preceded by the root domain of the server owner.

- **URL parameters**

The URL parameters hold strings which contain information needed by the server to service the request.

- **Target domain**

The target domain is the first part of the URL and is used to identify the IP address of the server using DNS (Domain Name Service) which returns an IP address in exchange for a string.

- **Browser information**

Different browsers use different methods to handle websites and sometimes outdated browsers or browsers lacking some features would be unable to server the request. Hence, browser information is attached to make sure that the server returns data that is accessible on the browser concerned.

Sample request:

GET https://developer.mozilla.org/en-US/search?q=client+server+overview&topic=apps&topic=html&topic=css&topic=js&topic=api&topic=webdev HTTP/1.1

Host: developer.mozilla.org

Connection: keep-alivePragma: no-cache

Cache-Control: no-cacheUpgrade-Insecure-Requests: 1User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8Referer: https://developer.mozilla.org/en-US/Accept-Encoding: gzip, deflate, sdch, br

Accept-Charset: ISO-8859-1,UTF-8;q=0.7,*;q=0.7Accept-Language: en-US,en;q=0.8,es;q=0.6

Cookie: sessionid=6ynxs23n521lu21b1t136rhbv7ezngie; csrftoken=zIPUJsAZv6pcgCBJSCj1zU6pQZbfMUAT; dwf_section_edit=False; dwf_sg_task_completion=False; _gat=1; _ga=GA1.2.1688886003.1471911953; ffo=true

STATES:

Start: The TCP connection is established; the client sends a HTTP request to the server to retrieve the webpage it should display.

S0: Recognize network protocol

- **Output '1':** If the browser successfully extracts the "http" part and recognizes that it is the name of the network protocol to use, it then moves to S2.
- **Output '0':** If the browser fails to extract or identify the protocol, goes to S1.

S1: Recognizing network protocol troubleshoot

- **Output '1':** When troubleshooting traces, the error to either the Internet Service Provider or the computer and resolves it, goes to state S0.
- **Output '0':** If troubleshooting fails goes to trap state, fails to connect and request will not be processed.

S2: Retrieve the IP address

- **Output '1':** After extraction and recognition of the network protocol, browser takes the domain name from the URL, and asks the internet Domain Name Server to return an Internet Protocol (IP) address. Moves to S4
- **Output '0':** IP address isn't returned, moves to S3.

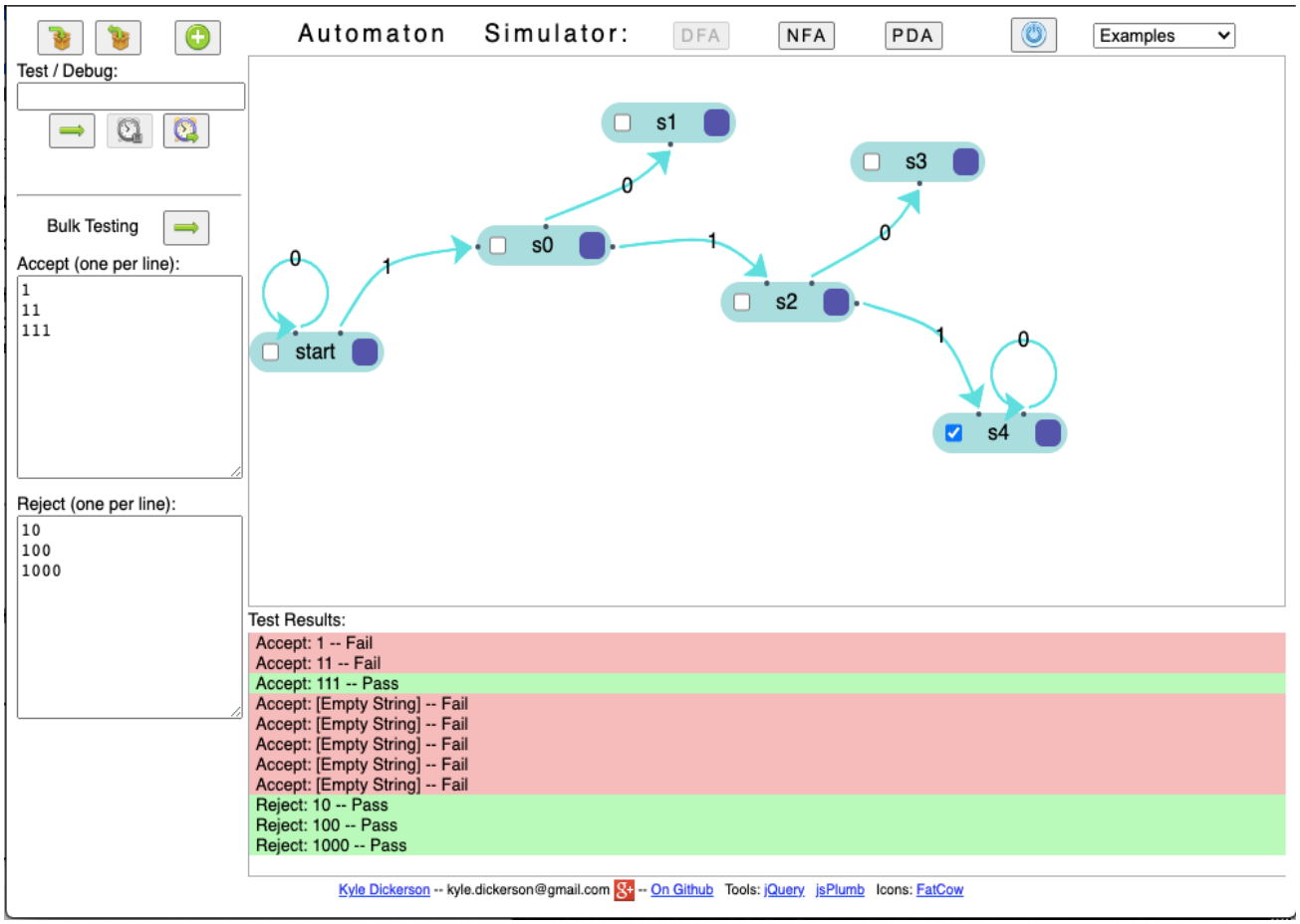
S3: Retrieving the IP address troubleshoot

- **Output '1':** If troubleshooting is successful, goes to state S2.

- **Output '0':** If troubleshooting fails due to a duplicate address/ no assigned address/system issues, fails to retrieve and does not proceed further.

S4 <final state>: Initiate request to locate the path requested; display.

- **Output '1':** If the server is able to locate the path requested, displays the http version, type of content an HTTP status code
- **Output '0':** If the server is not able to locate the path requested by the client, it will respond with the header: 404 NOT FOUND. A trap state since server couldn't trace the path, it cannot display the specific file the user requested.



The Firewall

In the process of the packets reaching their destination each of them must pass through two Firewalls:

1. Corporate internet firewall
2. Web server firewall (also called as packet filtering firewall)

Corporate internet firewall:

This corporate internet firewall mainly has two purposes:

1. It analyses each packet from the internet and prevent malicious or harmful packets from entering into the intranet.

2. It also analyses the data packets going out from the intranet and prevents harmful data from entering into the internet.

Web server firewall:

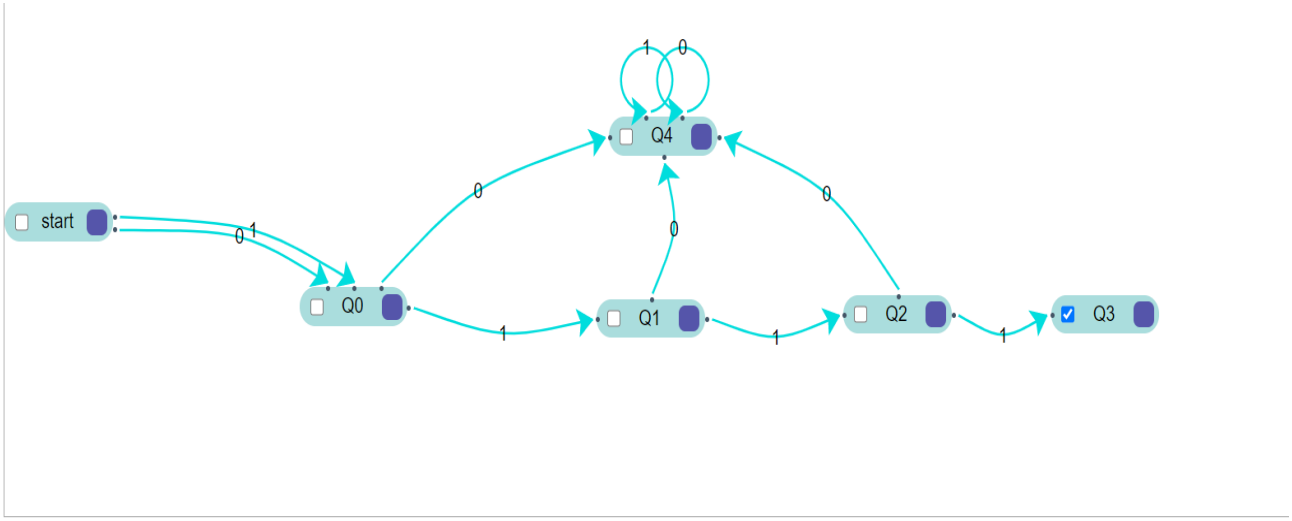
There are certain restrictions or set of rules that a web server firewall use to restrict data packets from entering into the web server. The types of information that the web server firewall basically look into is:

1. Port
2. Source address
3. Destination address

If the request is http, then only ports that are allowed are 25 and 80. Port 80 in http is the entrance from the internet to the webserver and port 25 usually deals with mail packets and all other ports are closed for private use.

So, only when the data packets pass through both the firewalls it will reach the webserver and then server handles the request.

DFA FOR THE CORPORATE INTERNET FIREWALL:



Test Results:

Accept: 1111 -- Pass
Accept: 1001 -- Fail
Accept: 0001 -- Fail
Accept: 0101 -- Fail
Accept: 0101 -- Fail
Accept: 0111 -- Pass
Accept: 1111 -- Pass
Accept: 10011 -- Fail

STATES:

Start – To Test and Debug the Deterministic Finite Automata

Q4 (Trap) - The trap state in this DFA Means the Packet will get destroyed.

Q0 – Packet Filter --Is a Packet Filter which gives an output of '1' when only rawIP, TCP, UDP, ICMP ...etc. i.e., valid type of packets are sent else will output '0' which leads to a trap state.

Q1 –checking for the malicious code - This state checks for the Malicious or Harmful code and gives an output of '1' if the packet is safe and '0' if this state finds some Malicious code and moves to the trap state (Q4) and the packet gets destroyed.

Q2 –checks the frequency to avoid DOS attacks - Will check for the frequency of each packet that is part of the request and if the frequency is less than 1ms then will result in an output of '0' and will move to the trap state (Q4).

Q3 –Final state which sends the packets reached here to DNR - And this is the final state of the DFA which will give the input to the initial state of the Domain name resolutor.

Regular Grammar:

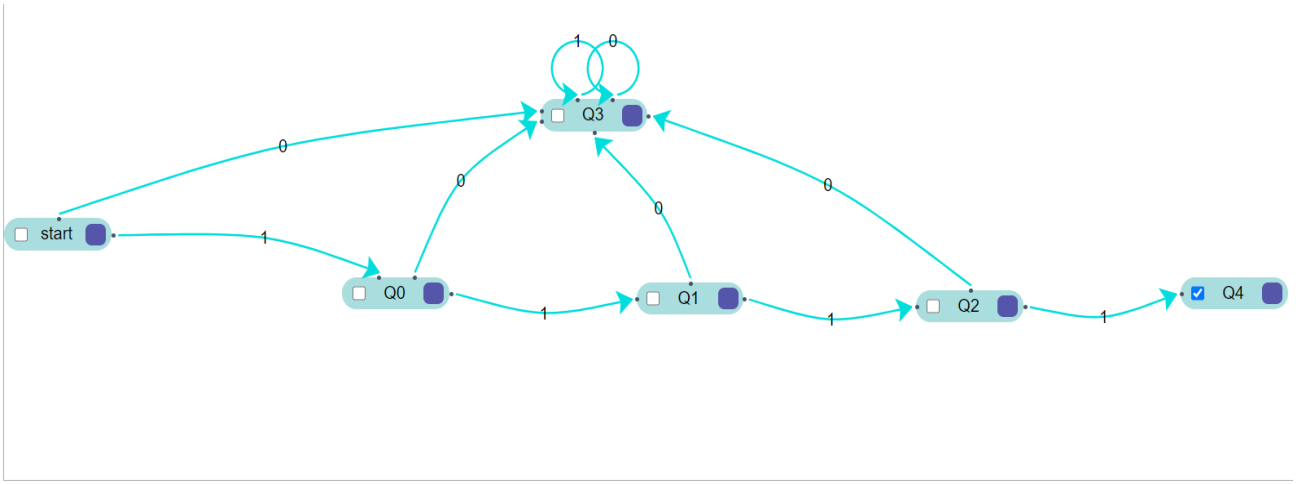
Right linear grammar: (Intermediate stages)

- Start -> 0Q0
- Start -> 1Q0
- Q0 - > 0Q4
- Q0 -> 1Q1
- Q1 -> 0Q4
- Q1 -> 1Q2
- Q2 -> 0Q4
- Q2 -> 1Q3
- Q3 -> λ

Final conclusive grammar results:

- S - > 0111A | 1111A
- A -> λ

DFA FOR THE WEB SERVER FIREWALL:



Test Results:

Accept: 1111 -- Pass
Accept: 0000 -- Fail
Accept: 1010 -- Fail
Accept: 1001 -- Fail
Accept: 0110 -- Fail
Reject: 10001 -- Pass
Reject: 0000 -- Pass
Reject: 1000 -- Pass
Reject: 1010 -- Pass
Reject: 1111 -- Fail
Reject: 1001 -- Pass

States:

Q0 –Port Filter - This state is basically a port filter and results in output '1' if the request is to access the webserver through the internet (Port 80) and if the request is regarding a mail (Port 25).

Q1 –checks for a valid source address- This state checks if the packets source address is valid and results in output '1' if it is valid and '0' if invalid

Q2 –checks for the valid destination address - This state checks if the packets destination address is valid and if valid results in '1' and if it's not valid output's '0'.

Q4 –Final state that sends the packets reached to the webserver - This state is the final state in the DFA. This state sends the reached packets to the webserver.

Q3 –Trap- Trap state which will destroy the packets in real time.

Regular Grammar:

Right linear grammar: (Intermediate stages)

- Start -> 0Q3
- Start -> 1Q0
- Q0 - > 0Q3
- Q0 -> 1Q1
- Q1 -> 0Q3
- Q1 -> 1Q2
- Q2 -> 0Q3
- Q2 -> 1Q3
- Q3 -> λ

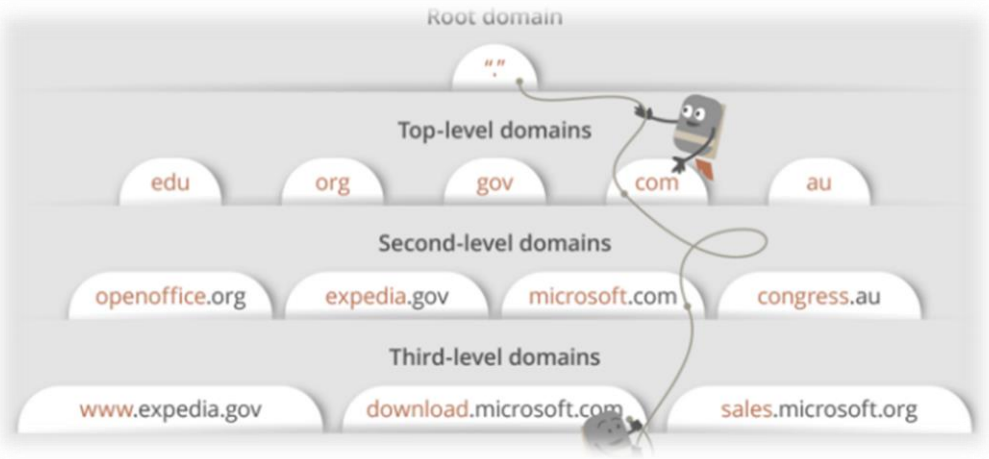
Final conclusive grammar results:

- S - > 1111A
- A -> λ

Domain Name Resolution

Since IP addresses which are used for identification of servers worldwide on the Internet are too hard for human beings to remember, we use a set of strings to identify these servers, which are then converted into IP addresses that can then be used to identify and contact the server/client in question.

Parts of a domain:

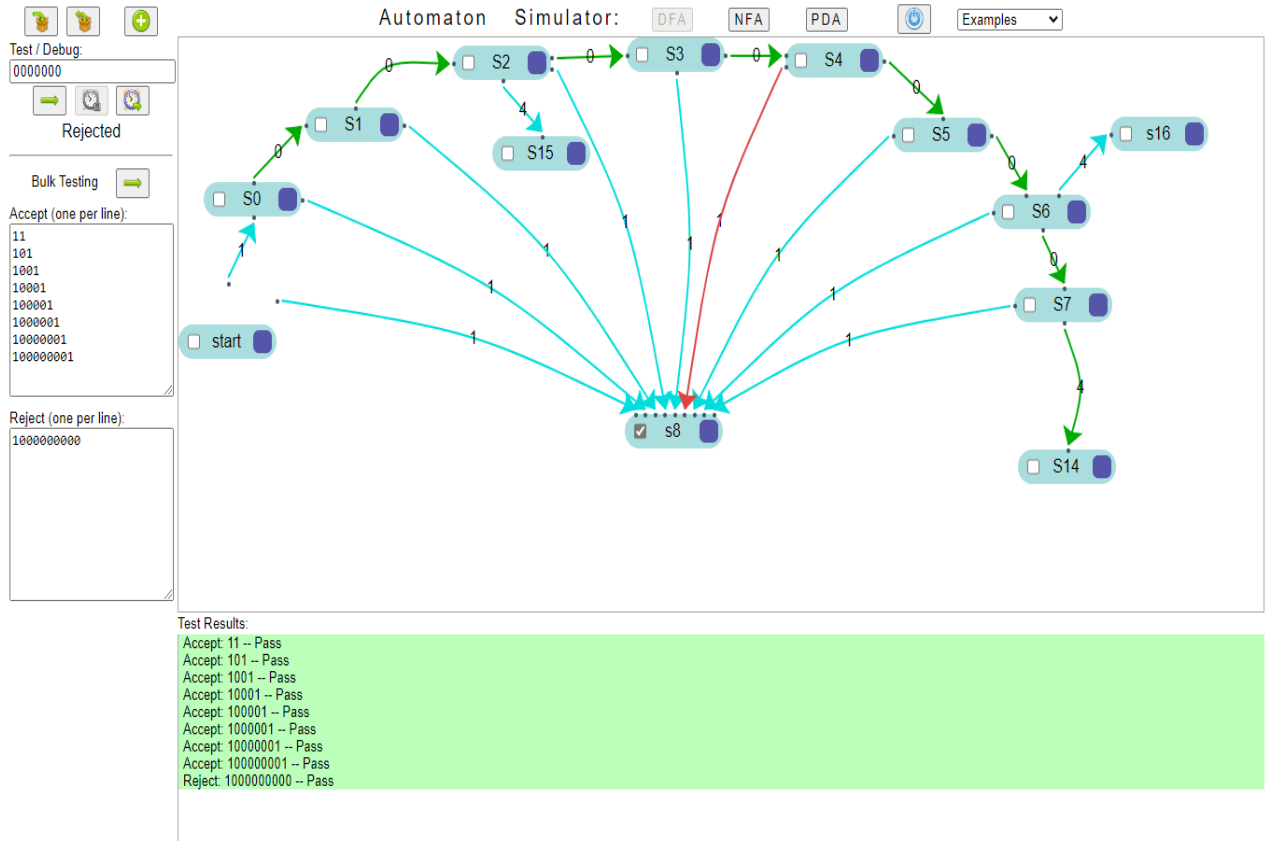


Identification Steps:

- The browser first checks its own DNS cache to see if the corresponding IP address is already available in the cache.
- If the IP address is not available in browser cache, the OS DNS cache is next queried for the IP address.
- Failing, the browser then contacts a resolving nameserver. This is a server, usually owned by the local ISP (Internet Service Provider) whose job is to resolve the given domain address and return an IP address.
- Cache of this server is checked, and if domain is found, IP address is returned.
- If the resolving nameserver also doesn't house the IP address, the root name resolver is called. These are 13 specially designated servers around the world which maintain a DNS database.
- If the domain is not found here either, the root name resolver provides the address of the root TLD (Top Level Domain) server.

- The TLD nameserver that corresponds to the TLD of the given domain might have the IP address, which is returned. If not, the TLD nameserver points to the authoritative name servers.
- When the domain is registered, the domain registrar assigns authoritative name servers which can connect to the servers with the IP addresses in question. These servers have addresses **ns1.domainname.tld**, **ns2.domainname.tld**, etc.
- These servers hold the IP address of the server in question and return them to the TLD root server which is returned to the ISP resolving nameserver and finally to the browser.
- If none of the DNS servers can resolve this domain, that means the domain doesn't exist.

DFA:



Link: Automation Simulator

To view: Copy link, go to Automaton Simulator, click on green 'Load' button in the top left corner and paste copied link in the 'Shareable URL' section.

States:

S0 – Browser cache -> On successful creation of HTTP response (from the Start state), the generated domain will be compared to the cache that is part of the browser to identify if the domain is available in the cache.

S1 – OS cache -> If the domain is not available in the browser cache, the domain is crosschecked with the cache maintained by the Operating System.

S2 – Contact resolving name server -> If the domain is not available in the OS cache, the next step is to contact the resolving name server, which is part of the local IPS infrastructure.

S3 – Resolving Name Server -> If the domain is present in the cache of the local resolving name server, it is returned.

S4 – Root name resolver -> If the resolver does not have the domain address, as the next step, the root name resolver is checked. This is a set of 13 unique servers duplicated all over the world which cache domains.

S5 – TLD Server -> If the domain is not available in the root name resolver, the next step is to check for existence in the TLD (Top Level Domain) server, which contains a cache of domain names associated with that particular top-level domain (.com, .org., .in, .co.uk, etc).

S6 – Identify authoritative name server -> If the domain is not available in the TLD cache, the next step is to identify the authoritative name server for that particular domain. When the domain is purchased from the domain name registrar, these authoritative servers are established and their purpose is to store the IP address of all sub domains of that particular domain. This server has to be identified from the TLD.

S7 – Authoritative name server -> If the authoritative name server has been successfully identified, the domain name is looked up in the database that is housed in this server and the correct IP address is returned.

Trap states:

S14 -> Failed to identify -> The entire stack of servers for DNS identification were checked and the provided domain was not available in the authoritative name server. This represents an error.

S15 -> Failed to contact resolver -> Attempts were made to contact the resolving server and were met with failure. This is an error.

S16 -> Failed to identify authoritative name server -> The authoritative name server for the entered domain was not available in the TLD server. This represents an error state.

Final states:

S8 -> IP has been returned -> The IP address of the given domain has been identified and returned to the process that requested it.

Regular Grammar:

Start -> 1 S0
S0 -> 0 S1
S1 -> 0 S2
S2 -> 0 S3
S3 -> 0 S4
S4 -> 0 S5
S5 -> 0 S6
S6 -> 0 S7
S7 -> 0S 14

- S0 -> 1 S8
- S1 -> 1 S8
- S2 -> 1 S8
- S3 -> 1 S8
- S4 -> 1 S8
- S5 -> 1 S8
- S6 -> 1 S8
- S7 -> 1 S8

- S2->4 S15
- S6 ->4 S16

Regular language:

- 11
- 101
- 1001
- 10001
- 100001
- 1000001
- 10000001

Regular Expression:

$$1.(\lambda+0)\ (\lambda+0)\ (\lambda+0)\ (\lambda+0)\ (\lambda+0)\ (\lambda+0).1$$

Request handling

Once a request has been received by the server, it has to be separated into different strings providing information about the source of the request and the service required, in order for the service to be successfully offered. Once the request has been handled, a response is generated in the same way as the request was generated and this is sent back to the requesting client, who will display the data accordingly.

Overall DFA

STATES:

Q0: Establishing the connection - Where the presence, working and troubleshooting of NIC and Router is checked. So is the connectivity with DNS server (Router switch) , IP address lookup and match and connectivity/ availability with servers (satellite/undersea cables)

- **Output '1':** When NIC is present, request to send to server is ready and the router is working, ready to transmit; so is the router switch, retrieval and lookup of the IP address and finally if a connection is established with the found IP address or when the troubleshooting is successful with the NIC/router/router switch and moves on to retrieve and lookup the IP address onto checking for connectivity

with and availability of servers. This indicates that the request generated by the client will be processed.

- **Output '0':** If an issue comes up with NIC/communicating with the router or the troubleshooting fails to go to the trap state or if the IP address is not located and the server connection fails. Creates a trap state.

Q1: Generating a HTTP request: Browser generates a HTTP request based on client's request in order to retrieve data from the concerned server. It is used to structure requests and responses over the internet.

- **Output '1 ':** When the request is broken down and it contains proper verb, a target URL, URL parameters, target domain and browser information which on identification by the server enables it to locate the path.
- **Output '0':** When request identification fails, and the server is unable to locate the path. Or when the specific piece of content requested was not found- throws the 404 NOT FOUND status code. Creates a trap state in order to resolve.

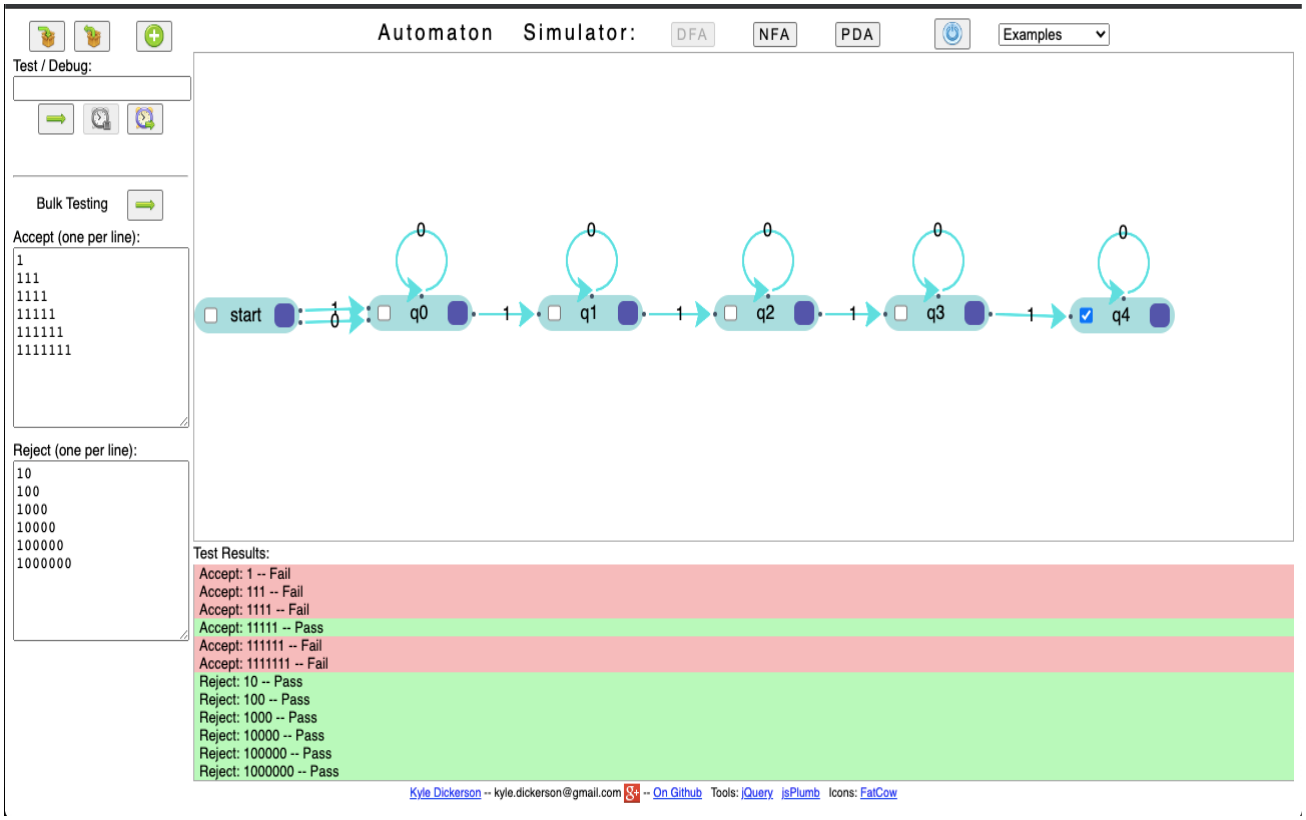
Q2: The Firewall: After generating a request, and before reaching their destinations, the packets go through firewalls.

- **Output '1':** When the packet passes through both the corporate internet firewall and the web server firewall, the server handles the request.
- **Output '0':** When the packet fails to let the server handle the request due to inconsistency in checking for malicious code and the frequency to avoid potential DOS attacks. Creates a trap state in order to troubleshoot the error.

Q3: Domain Name Resolution: Set of strings which are convertible to IP addresses for better and easier identification.

- **Output '1':** When the authoritative name server has been successfully identified, the domain name is traced and the corresponding IP address is returned.
- **Output '0':** When the domain is not available in the authoritative name server or when the attempts to contact the resolving server were met with failure or when there is a failure to identify the authoritative name server. Represents an error state.

Q4: Final state: Response is generated.


Automaton Simulator:
DFA
NFA
PDA
Examples
Test / Debug:
Bulk Testing
Accept (one per line):
1
111
1111
11111
111111
1111111
Reject (one per line):
10
100
1000
10000
100000
1000000
start
q0
q1
q2
q3
q4
Test Results:
Accept: 1 -- Fail
Accept: 111 -- Fail
Accept: 1111 -- Fail
Accept: 11111 -- Pass
Accept: 111111 -- Fail
Accept: 1111111 -- Fail
Reject: 10 -- Pass
Reject: 100 -- Pass
Reject: 1000 -- Pass
Reject: 10000 -- Pass
Reject: 100000 -- Pass
Reject: 1000000 -- Pass
Kyle Dickerson -- kyle.dickerson@gmail.com -- On Github Tools: jQuery jsPlumb Icons: FatCow

* * *