Byrne Seminar

Internet of Things Things

kulikows@cs.rutgers.edu
mcgrew@cs.rutgers.edu
<table>
<thead>
<tr>
<th>Week 1:</th>
<th>Definitions, mapping out the course</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week 2:</td>
<td>Composition of an IOT Thing. Programming it.</td>
</tr>
<tr>
<td></td>
<td>The IOT before the “I”: Apollo AGC (1961-71)</td>
</tr>
<tr>
<td>Week 3:</td>
<td>Introduction to our processor: The Arduino</td>
</tr>
<tr>
<td></td>
<td>Distribution of hardware <strong>WHICH I WANT BACK</strong></td>
</tr>
<tr>
<td>Week 4:</td>
<td>Security of IOT things. Fiddling with devices</td>
</tr>
<tr>
<td>Week 5:</td>
<td>Getting used to the Arduino and building devices</td>
</tr>
<tr>
<td>Week 6:</td>
<td>Show-offs; turn in your code</td>
</tr>
<tr>
<td></td>
<td>Uses of IOTs: medicine, &quot;smart cities&quot;, political oppression</td>
</tr>
<tr>
<td>Week 7:</td>
<td>Research description: you pick an IOT, and research it for presentation to the class</td>
</tr>
<tr>
<td>Week 8:</td>
<td>Security of IOT things #2: security, and liability</td>
</tr>
<tr>
<td>Week 9:</td>
<td>Presentations of research projects part 1</td>
</tr>
<tr>
<td>Week 10:</td>
<td>Presentations of research projects part 2</td>
</tr>
</tbody>
</table>
Arduino software install

https://www.arduino.cc/en/Main/Software

… select and install for whatever OS you’ve got.

Power: either power cable OR USB cable.

Program: runs most recently downloaded program… Forever

Check out

How things really, really work

- Obtain host name (e.g. www.cnn.com)

- Obtain host IP address

  www.cnn.com is a nickname for turner-tls.map.fastly.net
  turner-tls.map.fastly.net has address 151.101.201.67

- Discover “port” for the service (e.g. http = 80, https = 443)

- Try to connect to the port on that server

- read-write to the server

- done!
An example: the network guts of the temperature sensor stuff

- “sensor1.lcsr.rutgers.edu” runs a program that interrogates sensors, saves in /tmp/new_temp_file.save

- “report.cs.rutgers.edu” interrogates “sensor1”, sensor by sensor, to find out the temps and displays them
  
  TEST
  TEMP
  TIME
# define SERVER_PORT 1500
# define MAX_MSG 1024

/* bind server port */
struct sockaddr servAddr;
servAddr.sin_family = AF_INET;
servAddr.sin_addr.s_addr = htonl(INADDR_ANY);
servAddr.sin_port = htons(SERVER_PORT);

if(bind(sd, (struct sockaddr*)&servAddr, sizeof(servAddr))<0) {
    perror("cannot bind port ");
    return ERROR;
}
listen(sd,5);
while(1)
    { /* forever */
        cliLen = sizeof(cliAddr);
        newSd = accept(sd, (struct sockaddr *)&cliAddr, &cliLen);
        if(newSd<0) {
            perror("cannot accept connection ");
            return ERROR;
        }

        if (((strcmp(inet_ntoa(cliAddr.sin_addr), "128.6.168.251") != 0) &&
        (strcmp(inet_ntoa(cliAddr.sin_addr), "128.6.168.240") != 0) &&
        (strcmp(inet_ntoa(cliAddr.sin_addr), "128.6.168.241") != 0))
            { fprint(stderr, 
            "I won't talk to you, %s\n", inet_ntoa(cliAddr.sin_addr));
                    close(newSd);
                    goto restart;
            };

 Server side
while(read_line(newSd, line)!=ERROR) {
  line[strlen(line)-1] = '\0'; /* get rid of last \n */

  if (strlen(line) < 5) { /* bogus - no second argument */
    send_reply(newSd, "Bad Command\n");
    printf("server saw bad command\n");
  } /* bogus - no second argument */

  else { /* non-bogus -- maybe ok */
    bzero(action_arg, MAX_MSG);
    strcpy(action_arg, &line[5]);
    if (strlen(action_arg) == 0) { /* bogus - no second argument */
      send_reply(newSd, "Bad Command\n");
      printf("server saw bad command\n");
    } /* bogus - no second argument */
    else { /* non-bogus - maybe second test */
      if (strncmp(line, "TEST", 4) == 0) { /* test command */
        found = search_for(action_arg, return_arg);
        if (found == SUCCESS) send_reply(newSd, "0\n");
        else send_reply(newSd, "1\n");
      } /* test command */
      else if (strncmp(line, "READ", 4) == 0) { /* read command */
        found = search_for(action_arg, return_arg);
        if (return_arg[strlen(return_arg)-2] == ' ') {
          /* get rid of last blank */
          return_arg[strlen(return_arg)-2] = '\n';
          return_arg[strlen(return_arg)-1] = '\0';
        } /* get rid of last blank */
        s = rindex(return_arg, ' ');
        if (found == SUCCESS)
          send_reply(newSd, return_arg); /* send_reply(newSd, &s[1]);
          else send_reply(newSd, "0\n");
        } /* read command */
    } /* non-bogus - maybe second test */
  } /* non-bogus -- maybe ok */
} /* test command */
else if (strncmp(line, "TIME", 4) == 0)
    { /* time command */
        printf("server: time argument is *%s*\n", action_arg);
        statval = stat(DATAFILE, &statbuf);
        if (statval == 0)
            { /* got the info */
                sprintf(stat_str, "%d\n", statbuf.st_mtime);
                send_reply(newSd, stat_str);
            } /* got the info */
        else
            send_reply(newSd, "1\n");
    } /* time command */
else
    { /* invalid command */
        send_reply(newSd, "Bad Command\n");
    } /* invalid command */

} /* non-bogus -- maybe second test */

}; /* non-bogus -- maybe */

/* init line */
memset(line,0x0,MAX_MSG);

} /* while(read_line) */
close(newSd);
} /* while (1) */
Client side networking guts

```c
#define SERVER_PORT 1500
#define MAX_MSG 100

int main (int argc, char *argv[]) {
    if(argc < 3) { printf("usage: %s <server> <data1> <data2>
argv[0]); exit(1); }
    if (argc > 4) { printf("usage: %s <server> <data1> <data2>
argv[0]); exit(1); }

    h = gethostbyname(argv[1]);
    if(h==NULL) {printf("%s: unknown host %s\n",argv[0],argv[1]); exit(1);}

    servAddr.sin_family = h->h_addrtype;
    memcpy((char *)&servAddr.sin_addr.s_addr, h->h_addr_list[0], h->h_length);
    servAddr.sin_port = htons(SERVER_PORT);

    /* create socket */
    sd = socket(AF_INET, SOCK_STREAM, 0);
    if(sd<0) { perror("cannot open socket "); exit(1); }

    /* bind any port number */
    localAddr.sin_family = AF_INET;
    localAddr.sin_addr.s_addr = htonl(INADDR_ANY);
    localAddr.sin_port = htons(0);

    rc = bind(sd, (struct sockaddr *)&localAddr, sizeof(localAddr));
    if(rc<0) {
        printf("%s: cannot bind port TCP %s\n",argv[0],SERVER_PORT);
        perror("error ");
        exit(1);
    }
```
/* connect to server */
rc = connect(sd, (struct sockaddr *) &servAddr, sizeof(servAddr));
if(rc<0) {
    perror("cannot connect ");
    printf("Connection Refused\n");
    exit(1);
}

bzero(send_str, BUFFERSIZE);
strcpy(send_str, argv[2]);
strcat(send_str, " ");
strcat(send_str, argv[3]);
strcat(send_str, "\n");

rc = send(sd, send_str, strlen(send_str)+1, 0);
if(rc<0) {
    perror("client: cannot send data ");
close(sd);
    exit(1);
};

fd = fdopen(sd, "r+ ");
if(fd == NULL)
    {printf("can't reopen socket to read reply\n"); exit(1); }
num_chars = BUFFERSIZE;
tmp_s = fgets(reply_str, num_chars, fd);

reply_str[strlen(reply_str)-1] = '\0'; /* get rid of \n */
printf("%s\n", reply_str);

return 0;
Client side do things with the data

- A perl script that calls “client”, then reads back what “client” gets back
- Uses that information to build a (small) database of sensors-temps (including in the past)
- Uses that information to build the temperature graphs
Research Machine Room

Main Machine Room
DIMACS Machine Room
Supplemental Cooling

Building Supply Air

Last updated on Tuesday, September 26, 2017 at 11:03:47 AM (current)
Security: why does it matter?
Security: why does it matter?

• Maintain control of our devices
Security: why does it matter?

- Maintain control of our devices
- Maintain believability in our devices
Security: why does it matter?

- Maintain control of our devices
- Maintain believability in our devices
- Maintain control of our data
Security: why does it matter?

- Maintain control of our devices
- Maintain believability in our devices
- Maintain control of our data
- Control access to our data
- Limit who can see our data
Security: why does it matter?

- Maintain control of our devices
- Maintain believability in our devices
- Maintain control of our data
- Control access to our data
- Limit who can see our data
- Control of sensitive data
Security: why does it matter?

- Maintain control of our devices
- Maintain believability in our devices
- Maintain control of our data
- Control access to our data
- Limit who can see our data
- Control of sensitive data
- Protect users FROM our devices
Security: why does it matter?

- Maintain control of our devices
- Maintain believability in our devices
- Maintain control of our data
- Control access to our data
- Limit who can see our data
- Control of sensitive data
- Protect users FROM our devices
- Protect the world FROM our devices
Why do vendors seem to not care much for security?

- Which is more important, working-ness, or security?
- Which is more important, getting the product to market, or security?
- Which is more expensive, programming in security or not?
- Do buyers worry about security, or features?
- Security can be added in later, as we “beta-test” our products on our customers, and we look like caring vendors for providing “free updates”
Attack mechanisms

- Internet (wired)
- Internet (wireless)
- Bluetooth
- Physical access
- Software access
- Operating System access
Internet (wired)
Getting Started with the Aircrack-Ng Suite of Wi-Fi Hacking Tools

Welcome back, my fledgling hackers!

In the first part of my series on Wi-Fi hacking, we discussed the basic terms and technologies associated with Wi-Fi. Now that you have a firm grip on what Wi-Fi is exactly and how it works, we can start diving into more advance topics on how to hack Wi-Fi.
Bluetooth

SEP 12, 2017 @ 09:23 AM  5,992 👍

Critical Bluetooth Flaws Put Over 5 Billion Devices At Risk Of Hacking

Lucian Constantin, CONTRIBUTOR
I cover malware, vulnerabilities, data breaches and security research.
FULL BIO

Opinions expressed by Forbes Contributors are their own.

Bluetooth is one of the most popular short-range wireless communications technologies in use today and is built into many types of devices, from phones, smartwatches and TVs to medical equipment and car infotainment systems. Many of those devices are now at risk of being hacked due to critical flaws found in the Bluetooth protocol.
10.5: Reset a user's password in single user mode

We needed to reset the password on a Leopard system, but we didn't have the OS X install DVD available. After a few minutes of playing around, I came up with this solution:

1. Boot into single user mode (press Command-S at power on)
2. Type `fsck -fy`
3. Type `mount -uw /`
5. Type `dscl . -passwd/Users/username password`, replacing `username` with the targeted user and `password` with the desired password.
6. Reboot

This allows you to reset the password in single user mode without booting from the install media.

[robg adds: For everyone about to comment about this massive security hole, please don't do so. We ran a similar hint at the time of the OS X 10.0 release, and you can read the comments there for some of the give and take on the security issue. The bottom line is that someone with physical access has full access to your machine, regardless of whether or not they happened to bring a boot DVD with them. If you're truly worried about such things, then you'll want to use a combination of File Vault, a firmware password, and a case lock to minimize the chances that your machine is accessed.]
Physical Access

Google response to:

boot windows machine from usb stick

To boot from a CD or USB drive:
Restart your computer and wait for that first screen to pop up. Often, it'll say something like "Press F12 to Choose Boot Device" somewhere on the screen. ...
Give it a moment to continue booting, and you should see a menu pop up with a list of choices on it.

At a public Rutgers computer lab, I discovered every machine was configured to allow this. When I pointed it out, I was told, “yes. Please don’t tell anybody.”
Apple's Chinese App Store Has Come Under a Malware Attack

Apple's iOS App Store in China has been attacked for the first time by malware, multiple sources report. Internet security company Palo Alto Networks says that approximately 39 applications have been compromised.

According to the Wall Street Journal, hackers planted an outwardly normal version of an Apple software called Xcode, used to develop iOS applications, on a Chinese cloud service called Baidu Pan. Developers began using it because it was faster to download than the Xcode software from Apple's U.S. servers, the CBC reports, citing Palo Alto Networks.
Google Is Fighting A Massive Android Malware Outbreak -- Up To 21 Million Victims

Another month, another bunch of Android malware that's found its way onto Google Play. That's according to researchers from Check Point, who claimed to have found the second-biggest outbreak to ever hit Google's platform, with as many as 21.1 million infections from one malware family.

The malware's been dubbed ExpensiveWall after hiding...
Software/OS access

APPLE

This Nasty New Malware Can Infect Your Apple iPhone or iPad

Jonathan Vanian
Mar 16, 2016

Apple’s mobile operating system just got a nasty bug.

Researchers from the cybersecurity firm Palo Alto Networks (PANW, -1.40%) said on Wednesday that they discovered new malware that can infect Apple iOS devices even if they aren’t jailbroken.

A jailbroken device generally refers to an iPhone or iPad that has been modified without approval by Apple so the user can install software and apps that aren’t available on the Apple App Store. Apple discourages its customers to jailbreak their devices and says that doing so puts the devices at greater risk of security vulnerabilities.

Get Data Sheet, Fortune's technology newsletter.

The fact that the newly discovered malware, dubbed AceDeceiver, affects non-tampered iOS devices is noteworthy because it shows that hackers are “getting around Apple’s security measures,” the researchers explained. Although hackers are still primarily
Software/OS access

PSA: If you downloaded Handbrake last week, your Mac may be seriously compromised

Ben Lovejoy - May 8th 2017 3:51 am PT @benlovejoy
Software/OS access

Hundreds of apps infected by fake Xcode tools, Apple removing known malicious software from App Store

Benjamin Mayo - Sep. 20th 2015 4:20 pm PT  @lbzamayo

Apple has admitted that it is App Store integrity was compromised as apps were secretly infected by fake Xcode tools before submission to the App Store. The company has now officially acknowledged the problem and is now removing apps affected by this ‘hack’ from the App Store.
Things we can do

if ((strcmp(inet_ntoa(cliAddr.sin_addr), "128.6.168.251") != 0) &&
    (strcmp(inet_ntoa(cliAddr.sin_addr), "128.6.168.240") != 0) &&
    (strcmp(inet_ntoa(cliAddr.sin_addr), "128.6.168.241") != 0))
{
    fprintf(stderr,
        "I won't talk to you, %s\n", inet_ntoa(cliAddr.sin_addr));
    close(newSd);
    goto restart;
Things we can do

while(1) {
    if(rcv_ptr==0) {
        /* read data from socket */
        memset(rcv_msg,0x0,MAX_MSG); /* init buffer */
        n = recv(newSd, rcv_msg, MAX_MSG, 0); /* wait for data */
        if (n<0) {
            perror(" server: cannot receive data ");
            return ERROR;
        } else if (n==0) {
            printf(" connection closed by client\n");
            close(newSd);
            return ERROR;
        } else if (n==0) {
            printf(" connection closed by client\n");
            close(newSd);
            return ERROR;
        }
    }
}

/* copy line into 'line_to_return' */
while(*rcv_msg+rcv_ptr!=END_LINE && rcv_ptr<n) {
    memcpy(line_to_return+offset,rcv_msg+rcv_ptr,1);
    offset++;
    rcv_ptr++;
}
Buffer overflow

```c
#include <stdio.h>
#include <string.h>
int main(void)
{
    char buff[15];
    int pass = 0;
    printf("\n Enter the password : \n");
    gets(buff);

    if(strcmp(buff, "thegeekstuff"))
    {
        printf("\n Wrong Password \n");
    }
    else
    {
        printf("\n Correct Password \n");
        pass = 1;
    }
    if(pass)
    {
        /* Now Give root or admin rights to user*/
        printf("\n Root privileges given to the user \n");
    }

    return 0;
}
```
Enter the password:

```
hhhhhhhhhhhhhhhhhhhh
```

Wrong Password

Root privileges given to the user

**WHY?**

https://www.rapid7.com/resources/buffer-overflow-exploit-explained/

https://www.welivesecurity.com/2016/05/10/exploiting-1-byte-buffer-overflows/
An old vulnerability from days gone by

- CGI program included with original distribution of httpd – ‘finger.cgi’
- User presented with a text-box to type a user’s name into.
- System would take that, and run “finger whatyoutyped”

-bash-3.2$ finger mcgrew
Login: mcgrew       Name: Charles McGrew
Directory: /res/users/mcgrew       Shell: /bin/bash
On since Tue Sep 26 11:32 (EDT) on pts/2 from farside.rutgers.edu
Mail forwarded to mcgrew@farside.rutgers.edu
No mail.
No Plan.
If you typed

“whatyoutype; chpasswd –stdin root:anewpassword”

Since the web server ran as superuser (ah, those were the days), and didn’t look at the input it got, it would run change the superuser’s password.