ATU: Anti-Theft Unit

Focus: Prevent unauthorized access or theft.

Software design:

Implement multi-factor authentication (MFA) to protect access to systems.

Use real-time tracking and geofencing for physical devices.

Add encryption for sensitive data, ensuring stolen information is unusable.

Build in automatic lock-and-wipe mechanisms triggered by predefined conditions, like prolonged inactivity or unauthorized attempts.

```
IDENTIFICATION DIVISION.

PROGRAM-ID. Database-MFA-System.

ENVIRONMENT DIVISION.

DATA DIVISION.


WORKING-STORAGE SECTION.

01 User-Input.

  05 Username        PIC X(20).

  05 Password        PIC X(20).
```

```
   05 Security-Code    PIC 9(7).

01 Stored-Database.
   05 Stored-Username    PIC X(20) VALUE " live ".
   05 Stored-Password    PIC X(20) VALUE " forever ".
   05 Stored-Code1       PIC 9(5) VALUE 82698.
   05 Stored-Code2       PIC 9(7) VALUE 7628429.

01 Verification-Status PIC X VALUE "N".

PROCEDURE DIVISION.
    DISPLAY "Enter Username: " WITH NO ADVANCING.
    ACCEPT Username.

    DISPLAY "Enter Password: " WITH NO ADVANCING.
    ACCEPT Password.

    IF Username = Stored-Username
```

```
           AND Password = Stored-Password
        THEN
           DISPLAY "Enter Security Code: " WITH NO
ADVANCING.
           ACCEPT Security-Code.

           IF Security-Code = Stored-Code1
             OR Security-Code = Stored-Code2
           THEN
              MOVE "Y" TO Verification-Status
              DISPLAY "Access Granted. System Secured."
           ELSE
              DISPLAY "Invalid Security Code. Access Denied."
           END-IF
        ELSE
           DISPLAY "Invalid Credentials. Access Denied."
        END-IF.
```

```
        STOP RUN.


import sys


# Simulating COBOL WORKING-STORAGE SECTION

# 01 User-Input.

#    05 Username        PIC X(20).

#    05 Password        PIC X(20).

#    05 Security-Code    PIC 9(7).

username_input = ""

password_input = ""

security_code_input_str = "" # Store raw input for
potential error display

security_code_input_num = 0


# 01 Stored-Database.

#    05 Stored-Username    PIC X(20) VALUE " live ".

#    05 Stored-Password    PIC X(20) VALUE " forever ".
```

```
#    05 Stored-Code1       PIC 9(5) VALUE 82698.

#    05 Stored-Code2       PIC 9(7) VALUE 7628429.

stored_username = " live           " # Padded to 20 chars for exact COBOL PIC X(20) behavior if needed, though comparison usually ignores trailing spaces

stored_password = " forever       " # Padded to 20 chars

stored_code1 = 82698

stored_code2 = 7628429



# 01 Verification-Status PIC X VALUE "N".

verification_status = "N"



# PROCEDURE DIVISION.

# DISPLAY "Enter Username: " WITH NO ADVANCING.

print("Enter Username: ", end='')

# ACCEPT Username.

# Read input and pad/truncate to 20 characters to mimic PIC X(20) fixed length

username_input = input().ljust(20)[:20]
```

```
# DISPLAY "Enter Password: " WITH NO ADVANCING.

print("Enter Password: ", end='')

# ACCEPT Password.

# Read input and pad/truncate to 20 characters

password_input = input().ljust(20)[:20]



# IF Username = Stored-Username

#    AND Password = Stored-Password

# THEN

if username_input == stored_username and
password_input == stored_password:

   # DISPLAY "Enter Security Code: " WITH NO
ADVANCING.

   print("Enter Security Code: ", end='')

   # ACCEPT Security-Code.

   security_code_input_str = input()
```

```
    # Attempt to convert input to integer, mimicking PIC
9(7)

    try:

        # COBOL ACCEPT for numeric fields usually handles
leading/trailing spaces

        # and potentially signs depending on the exact PIC
clause.

        # PIC 9(7) implies an unsigned integer.

        # We attempt to convert to int. COBOL might truncate
or error on > 7 digits.

        # Python int() handles varying lengths. We'll check
length conceptually.

        # We don't explicitly enforce 7 digits here as the
COBOL ACCEPT behavior

        # for invalid numeric input can vary and isn't fully
specified.

        # We prioritize converting valid numbers.

        security_code_input_num =
int(security_code_input_str.strip())



        # IF Security-Code = Stored-Code1

        #    OR Security-Code = Stored-Code2
```

```
        # THEN

        if security_code_input_num == stored_code1 or
security_code_input_num == stored_code2:

            # MOVE "Y" TO Verification-Status

            verification_status = "Y"

            # DISPLAY "Access Granted. System Secured."

            print("Access Granted. System Secured.")

        # ELSE

        else:

            # DISPLAY "Invalid Security Code. Access Denied."

            print("Invalid Security Code. Access Denied.")

        # END-IF

    except ValueError:

        # Handle non-numeric input for security code

        print("Invalid Security Code (must be numeric). Access
Denied.")

# ELSE

else:

    # DISPLAY "Invalid Credentials. Access Denied."
```

```
    print("Invalid Credentials. Access Denied.")

# END-IF.



# STOP RUN.

# In Python, the script naturally ends here. sys.exit()
provides explicit termination.

sys.exit(0)



// Database-MFA-System



// User Input

let userInput = {

    username: "",

    password: "",

    securityCode: ""

};



// Stored Database
```

```
const storedDatabase = {
    storedUsername: " live ",
    storedPassword: " forever ",
    storedCode1: 82698,
    storedCode2: 7628429
};

// Verification Status
let verificationStatus = "N";

// Function to simulate user input
function promptUser(message) {
    return prompt(message);
}

// Main Procedure
userInput.username = promptUser("Enter Username: ");
userInput.password = promptUser("Enter Password: ");
```

```
if (userInput.username === storedDatabase.storedUsername && userInput.password === storedDatabase.storedPassword) {

   userInput.securityCode = promptUser("Enter Security Code: ");



   if (userInput.securityCode == storedDatabase.storedCode1 || userInput.securityCode == storedDatabase.storedCode2) {

      verificationStatus = "Y";

      console.log("Access Granted. System Secured.");

   } else {

      console.log("Invalid Security Code. Access Denied.");

   }

} else {

   console.log("Invalid Credentials. Access Denied.");

}
```

```
create {

   // This code initializes a process and provides
instructions.

   instructions: "This code starts a process that will
execute a series of tasks. Ensure that all prerequisites are
met before starting the process.",

   .start

}

; Dependencies: NASM Assembler, ld Linker (standard
Linux build tools)

; Assembling and Linking (Example):

; nasm -f elf64 -o program.o program.asm

; ld -o program program.o



section .data

   ; Equivalent to COBOL WORKING-STORAGE SECTION
constants and literals



   ; User Input Prompts (from DISPLAY statements)

   prompt_user     db "Enter Username: ", 0    ; Null-
terminated string for helper
```

```
    prompt_pass     db "Enter Password: ", 0    ; Null-
terminated string

    prompt_code     db "Enter Security Code: ", 0 ; Null-
terminated string



    ; Stored Database Values (Equivalent to VALUE clauses)

    ; 05 Stored-Username    PIC X(20) VALUE "live".

    stored_user     db "live            " ; Padded with spaces to
20 bytes

    stored_user_len equ 20



    ; 05 Stored-Password    PIC X(20) VALUE "forever".

    stored_pass     db "forever         " ; Padded with spaces to
20 bytes

    stored_pass_len equ 20



    ; 05 Stored-Code1       PIC 9(5) VALUE 82698.

    ; Stored as numeric value for comparison

    stored_code1_val dq 82698
```

```
    ; 05 Stored-Code2       PIC 9(7) VALUE 7628429.

    ; Stored as numeric value for comparison

    stored_code2_val dq 7628429



    ; 01 Verification-Status PIC X VALUE "N".

    verification_status db "N"



    ; Output Messages (from DISPLAY statements)

    msg_granted       db "Access Granted. System Secured.",
0xA, 0 ; Added newline (0xA) and null terminator

    msg_denied_code   db "Invalid Security Code. Access
Denied.", 0xA, 0 ; Added newline and null

    msg_denied_creds  db "Invalid Credentials. Access
Denied.", 0xA, 0 ; Added newline and null



section .bss

    ; Equivalent to COBOL WORKING-STORAGE SECTION
variables without VALUE clause



    ; 01 User-Input.
```

```
    ;    05 Username        PIC X(20).

    input_user      resb 21  ; Reserve 20 bytes for data + 1 for
potential newline/processing



    ;    05 Password        PIC X(20).

    input_pass      resb 21  ; Reserve 20 bytes for data + 1



    ;    05 Security-Code    PIC 9(7).

    input_code      resb 8   ; Reserve 7 bytes for digits + 1



section .text

    global _start          ; Linker entry point



;--------------------------------------------------

; Helper function to print a null-terminated string

; Arg: rdi = address of the null-terminated string

; Uses syscall: sys_write

; Clobbers: rax, rsi, rdx, rdi, rbx
```

```
;-----------------------------------------------------

print_string_z:

   push rdi           ; Save original pointer if needed later

   mov rbx, rdi       ; Use rbx to find the end of the string

.loop:

   cmp byte [rbx], 0   ; Check for null terminator

   je .done

   inc rbx

   jmp .loop

.done:

   sub rbx, rdi       ; rbx now holds the length

   mov rax, 1         ; syscall number (sys_write)

   mov rsi, rdi       ; address of string to write

   mov rdx, rbx       ; length of string

   mov rdi, 1         ; file descriptor 1 (stdout)

   syscall

   pop rdi           ; Restore original pointer

   ret
```

```
;-----------------------------------------------------

; Helper function to read input from stdin

; Args: rdi = buffer address, rsi = buffer size

; Returns: rax = number of bytes read (excluding potential
trailing newline)

; Uses syscall: sys_read

; Clobbers: rax, rdi, rsi, rdx, rbx

;-----------------------------------------------------

read_input:

    mov rax, 0            ; syscall number (sys_read)

    mov rdx, rsi          ; buffer size (max bytes to read)

    mov rsi, rdi          ; buffer address

    mov rdi, 0            ; file descriptor 0 (stdin)

    syscall               ; Kernel reads input, rax = bytes read or -
1 on error



    ; Check for read error or EOF

    cmp rax, 0
```

```
    jle .read_error         ; If bytes read <= 0, handle as
error/EOF



    ; Remove trailing newline (0xA) if present

    mov rbx, rax            ; Save original byte count read in rbx

    dec rax                 ; Point rax to the index of the last
character read

    cmp byte [rsi + rax], 0xA ; Compare the last byte with
newline character

    jne .no_newline         ; If it's not a newline

1000

, the actual length is rbx



    ; It was a newline, rax already holds the length
*without* the newline

    jmp .read_done

.no_newline:

    mov rax, rbx            ; Restore original count (rbx) if no
newline found

.read_done:
```

```
    ret

.read_error:

    ; On error or EOF, return 0 bytes read

    mov rax, 0

    ret



;-----------------------------------------------------

; Helper function to convert ASCII string to unsigned
integer

; Args: rdi = string pointer, rsi = length of string segment to
convert

; Returns: rax = integer value, rcx = 0 on success, 1 on
error (non-digit found)

; Clobbers: rax, rbx, rcx, rdx

; Note: Simple implementation, no overflow check.

;-----------------------------------------------------

atoi:

    mov rax, 0          ; Initialize result accumulator to 0

    mov rcx, 0          ; Initialize index/counter to 0
```

```
    mov rbx, 10          ; Base for decimal conversion

.loop:

    cmp rcx, rsi         ; Have we processed all characters
based on length?

    jge .done            ; Yes, jump to done


    movzx rdx, byte [rdi + rcx] ; Get current character into dl
(zero-extend to rdx)


    ; Validate if the character is a digit '0'-'9'

    cmp dl, '0'

    jl .error            ; If less than '0', it's not a digit

    cmp dl, '9'

    jg .error            ; If greater than '9', it's not a digit


    ; Convert character digit to integer value

    sub dl, '0'          ; '0' becomes 0, '1' becomes 1, etc.


    ; Update the result: result = result * 10 + digit
```

```
    imul rax, rax, rbx    ; Multiply current result by 10
    add rax, rdx          ; Add the new digit's value

    inc rcx               ; Move to the next character index
    jmp .loop

.done:
    mov rcx, 0            ; Set return code to 0 (success)
    ret

.error:
    mov rax, 0            ; Return 0 on error (or could return
specific error value)
    mov rcx, 1            ; Set return code to 1 (error)
    ret

;-----------------------------------------------

; Main program logic starts here
```

```
; Equivalent to COBOL PROCEDURE DIVISION
;--------------------------------------------------
_start:
    ; DISPLAY "Enter Username: " WITH NO ADVANCING.
    mov rdi, prompt_user    ; Load address of prompt string
    call print_string_z     ; Call helper to print it

    ; ACCEPT Username.
    mov rdi, input_user     ; Load buffer address
    mov rsi, 21             ; Max buffer size
    call read_input         ; Read user input
    mov r12, rax            ; Store actual length read (excluding
newline) in r12

    ; DISPLAY "Enter Password: " WITH NO ADVANCING.
    mov rdi, prompt_pass    ; Load address of prompt string
    call print_string_z     ; Call helper to print it
```

```
    ; ACCEPT Password.

    mov rdi, input_pass     ; Load buffer address

    mov rsi, 21             ; Max buffer size

    call read_input         ; Read user input

    mov r13, rax            ; Store actual length read (excluding
newline) in r13



    ; IF Username = Stored-Username AND Password =
Stored-Password

    ; Check Username length first

    cmp r12, stored_user_len ; Compare input length with
expected length (20)

    jne .invalid_credentials ; If lengths differ, credentials
invalid



    ; Compare Username content (fixed length)

    mov rdi, input_user     ; First string (input)

    mov rsi, stored_user    ; Second string (stored)

    mov rcx, stored_user_len ; Length to compare (20)
```

```
    cld                 ; Clear direction flag (forward
comparison)

    repe cmpsb          ; Repeat compare string byte while
equal

    jne .invalid_credentials ; If comparison failed (ZF=0),
credentials invalid



    ; Check Password length

    cmp r13, stored_pass_len ; Compare input length with
expected length (20)

    jne .invalid_credentials ; If lengths differ, credentials
invalid



    ; Compare Password content (fixed length)

    mov rdi, input_pass     ; First string (input)

    mov rsi, stored_pass    ; Second string (stored)

    mov rcx, stored_pass_len ; Length to compare (20)

    cld

    repe cmpsb          ; Repeat compare string byte while
equal

    jne .invalid_crede
```

```
1000

ntials ; If comparison failed (ZF=0), credentials invalid


    ; THEN (Credentials matched)

.mfa_check:

    ; DISPLAY "Enter Security Code: " WITH NO ADVANCING.

    mov rdi, prompt_code    ; Load address of prompt string

    call print_string_z     ; Call helper to print it


    ; ACCEPT Security-Code.

    mov rdi, input_code     ; Load buffer address

    mov rsi, 8              ; Max buffer size (7 digits + potential
newline)

    call read_input         ; Read user input

    mov r14, rax            ; Store actual length read (excluding
newline) in r14


    ; Check if read was successful and length > 0

    cmp r14, 0
```

```
    jle .invalid_code       ; Treat empty/error input as invalid
code



    ; Convert Security-Code input string to integer for
comparison

    mov rdi, input_code     ; Address of the input string

    mov rsi, r14            ; Length of the input string

    call atoi               ; Convert to integer -> result in rax,
status in rcx



    ; Check if atoi conversion was successful

    cmp rcx, 0

    jne .invalid_code       ; If atoi failed (rcx=1, non-digit
input), code is invalid



    ; rax now holds the integer value entered by the user



    ; IF Security-Code = Stored-Code1 OR Security-Code =
Stored-Code2

    ; Compare input integer (rax) with Stored-Code1
```

```
    mov rbx, [stored_code1_val] ; Load stored numeric
value 1

    cmp rax, rbx

    je .access_granted      ; If equal, jump to grant access


    ; Compare input integer (rax) with Stored-Code2

    mov rbx, [stored_code2_val] ; Load stored numeric
value 2

    cmp rax, rbx

    je .access_granted      ; If equal, jump to grant access


    ; ELSE (Security code did not match either stored code)

    jmp .invalid_code       ; Jump to invalid code handling


    ; THEN (Security code matched)

.access_granted:

    ; MOVE "Y" TO Verification-Status

    mov byte [verification_status], "Y" ; Update status flag
```

```
    ; DISPLAY "Access Granted. System Secured."

    mov rdi, msg_granted    ; Load address of success
message

    call print_string_z     ; Print the message

    jmp .exit_program       ; Proceed to program termination



    ; ELSE (Security code was invalid)

.invalid_code:

    ; DISPLAY "Invalid Security Code. Access Denied."

    mov rdi, msg_denied_code ; Load address of invalid
code message

    call print_string_z     ; Print the message

    jmp .exit_program       ; Proceed to program
termination



    ; ELSE (Credentials were invalid)

.invalid_credentials:

    ; DISPLAY "Invalid Credentials. Access Denied."

    mov rdi, msg_denied_creds ; Load address of invalid
credentials message
```

```
    call print_string_z       ; Print the message

    jmp .exit_program         ; Proceed to program
termination



.exit_program:

    ; STOP RUN.

    mov rax, 60               ; syscall number (sys_exit)

    xor rdi, rdi              ; exit code 0 (successful termination)

    syscall                   ; Make the system call to exit
```

ATO: Authorization to Operate

Focus: Ensuring secure system functionality.

Software design:

Develop access control lists (ACLs) that define user roles and permissions explicitly.

Use certificate-based authentication to validate users and devices.

Schedule regular security audits within the application and automate compliance checks.

Create a “pre-launch” scanning process for security vulnerabilities before systems go live.

```
create {
```

```
    // Initialization and instructions.

    instructions: "This software implements access control,
certificate-based authentication, security audits, and
vulnerability scanning to ensure secure system
functionality.",

    .start {

        // Step 1: Initialize Access Control Lists (ACLs).

        acl {

            roles: ["Admin", "User", "Guest"],

            permissions: {

                Admin: ["read", "write", "delete"],

                User: ["read", "write"],

                Guest: ["read"]

            }

        }

        // Step 2: Implement Certificate-Based
Authentication.

        certificate {

            validation: {

                issuer: "TrustedAuthority",
```

```
            expiration-check: "enabled"
        }
    }
    // Step 3: Schedule Regular Security Audits.
    audit {
        frequency: "monthly",
        automated-checks: ["access violations", "system
updates"]
    }
    // Step 4: Pre-launch Vulnerability Scanning.
    prelaunch-scan {
        scan-modules: ["network integrity", "malware
detection"],
        resolve-before-launch: "critical issues"
    }
  }
}
```

```
    IDENTIFICATION DIVISION.
```

```
PROGRAM-ID. Authorization-to-Operate.

ENVIRONMENT DIVISION.

DATA DIVISION.


WORKING-STORAGE SECTION.

01 ACL-Database.

  05 Role      PIC X(10) VALUE "Admin".

  05 Permission PIC X(20) VALUE "read, write, delete".


01 Certificate-Details.

  05 Issuer       PIC X(30) VALUE "TrustedAuthority".

  05 Expiration   PIC X(10) VALUE "2025-12-31".


01 Audit-Schedule.

  05 Frequency    PIC X(10) VALUE "Monthly".


01 Vulnerability-Scan.

  05 Module1      PIC X(20) VALUE "Network Integrity".
```

```
        05 Module2      PIC X(20) VALUE "Malware
Detection".


      PROCEDURE DIVISION.

          DISPLAY "Initializing Access Control Lists...".

          DISPLAY "Role: " Role ", Permissions: " Permission.


          DISPLAY "Validating Certificate...".

          IF Issuer = "TrustedAuthority"

            AND Expiration > "2025-04-25"

          THEN

             DISPLAY "Certificate Valid."

          ELSE

             DISPLAY "Invalid Certificate. Access Denied."

          END-IF.


          DISPLAY "Scheduling Security Audits...".

          DISPLAY "Frequency: " Frequency.
```

```
        DISPLAY "Performing Pre-Launch Vulnerability
Scanning...".

        DISPLAY "Scanning Modules: " Module1 ", "
Module2.

        DISPLAY "Critical Issues Resolved. Ready to
Operate.".



        STOP RUN.
```

Features Included:

Access Control Lists (ACLs): Role-based permissions are defined explicitly in both examples.

Certificate-Based Authentication: Validation of certificates ensures secure user/device operations.

Automated Security Audits: Regular checks are scheduled to maintain system compliance.

Pre-Launch Scanning: Vulnerabilities are analyzed and resolved before deployment.

ATD: Advanced Threat Detection

Focus: Identifying sophisticated threats.

Software design:

Incorporate machine learning models to detect anomalies in user behavior or network traffic.

Develop modules to analyze log files for patterns consistent with attacks.

Use sandboxing to safely inspect and analyze suspicious files or code in isolation.

Real-time monitoring systems with alert mechanisms for detected threats.

- create {
- // Initialization and instructions.
- instructions: "This software detects sophisticated threats using anomaly detection, log analysis, sandboxing, and real-time monitoring with alerts.",
- .start {
- // Step 1: Initialize Threat Detection System.
- detection-system {
- machine-learning {
- model: "Anomaly-Detection",
- training-data: ["user-behavior", "network-traffic"],
- threshold: "0.95"
- }
- }
- // Step 2: Analyze Log Files for Attack Patterns.
- log-analysis {
- log-files: ["syslog", "auth.log"],

- patterns: ["multiple failed login attempts", "unexpected network activity"]
- }
- // Step 3: Implement Sandboxing for Suspicious Files.
- sandbox {
- isolation: "enabled",
- scan-modules: ["file-behavior", "malware-signatures"]
- }
- // Step 4: Real-Time Monitoring and Alert System.
- monitoring {
- alert-mechanisms: ["email", "SMS", "dashboard-alerts"],
- response-time: "immediate",
- alerts: ["unauthorized access detected", "suspicious file execution"]
- }
- }
- }

IDENTIFICATION DIVISION.

PROGRAM-ID. Advanced-Threat-Detection.

ENVIRONMENT DIVISION.

DATA DIVISION.

WORKING-STORAGE SECTION.

```
       01 Threat-Parameters.

          05 ML-Threshold          PIC 9(4)V99 VALUE 0.95.

          05 Suspicious-Patterns    PIC X(30) VALUE "Multiple
Failed Logins".

          05 Isolation-Status       PIC X(8) VALUE "Enabled".

          05 Alert-Methods          PIC X(30) VALUE "Email, SMS,
Dashboard".



       01 Real-Time-Monitoring.

          05 Unauthorized-Access-Flag PIC X(3) VALUE "N".



       PROCEDURE DIVISION.

           DISPLAY "Initializing Threat Detection System...".

           DISPLAY "Machine Learning Threshold: " ML-
Threshold.



           DISPLAY "Analyzing Log Files for Patterns...".

           IF Suspicious-Patterns = "Multiple Failed Logins"

           THEN
```

```
            DISPLAY "Threat Detected! Initiating
Sandboxing...".

            DISPLAY "Isolation Status: " Isolation-Status.



            DISPLAY "Real-Time Alerts Sent via: " Alert-
Methods.

            MOVE "Y" TO Unauthorized-Access-Flag

        ELSE

            DISPLAY "No Threats Detected. Monitoring
Continues.".

        END-IF.



        DISPLAY "Monitoring System Active. Waiting for
Alerts...".

        STOP RUN.
```

Imagine a system that learns regular user activity, like logging in at specific times or visiting certain websites.

An ML model flags anomalies, such as a user suddenly accessing the system from another country late at night. This suggests unauthorized access.

If a spike in data usage occurs that doesn't fit the typical pattern (e.g., downloading gigabytes of sensitive files), the system raises an alert.

In COBOL, a threshold (e.g., "abnormal behavior score above 0.95") triggers actions such as isolating the user's session or revoking access temporarily until verified.

2. Log File Analysis

Consider a log file tracking login attempts. It records each failed or successful attempt and includes timestamps.

The software scans logs and notices 15 failed login attempts in a short timeframe, indicating a brute-force attack.

Another example: Analyzing a network log shows an IP address sending irregular amounts of data, a sign of data exfiltration.

Both the create code and COBOL system would take these patterns and initiate protective measures, like locking the affected account or disconnecting suspicious network activity.

3. Sandboxing Suspicious Files

A file attached to an email looks harmless but acts oddly when opened (e.g., it starts accessing system files).

Before allowing the file to run, the sandbox isolates it in a virtual environment to study its behavior safely.

If the file tries to copy sensitive data or execute commands without permission, the sandbox blocks it and marks it as malware.

For example, the sandbox might detect a script within a PDF file attempting unauthorized access to user credentials. It alerts administrators before any damage occurs.

4. Real-Time Monitoring and Alerts

The system monitors activities in real-time, such as login attempts, file transfers, or changes to permissions.

If an unauthorized user gains access or someone attempts to modify admin rights, the software sends an alert via email, SMS, or displays a dashboard warning.

For instance, an admin receiving an SMS might see: “Alert! Suspicious user behavior detected: multiple failed logins followed by file access.”

Real-time alerts help administrators act immediately, preventing further security breaches.

Explanation of Features

Machine Learning Anomaly Detection:

The create code includes an ML model trained on user behavior and network traffic.

In COBOL, the threat detection threshold ensures anomalies exceeding the limit trigger actions.

Log File Analysis:

Both versions examine logs for attack patterns (e.g., failed login attempts).

Sandboxing Suspicious Files:

Suspicious files are isolated for further examination, limiting potential damage.

Real-Time Monitoring and Alerts:

Real-time alerts are implemented in both formats to notify administrators immediately of unauthorized access or other issues.

ATS: Automated Threat Systems

Focus: Reacting to and neutralizing threats.

Software design:

Build response automation scripts that quarantine infected systems or restrict access during a breach.

Integrate event-driven triggers for specific types of threats (e.g., DDoS attacks, unauthorized logins).

Use decision-making algorithms to categorize threats and initiate appropriate countermeasures.

Combine with ATD for real-time detection and immediate response.

```
create {

   // Initialization and instructions.
```

```
    instructions: "This Automated Threat System detects
and neutralizes threats by quarantining infected systems,
triggering event-based responses, and using decision-
making algorithms.",

    .start {

        // Step 1: Initialize Threat Response Automation.

        response-automation {

            actions: {

                quarantine: "infected-system",

                restrict-access: "unauthorized-users"

            }

        }

        // Step 2: Define Event-Driven Triggers.

        event-triggers {

            threats: ["DDoS attack", "unauthorized login"],

            responses: {

                "DDoS attack": "block IP address, reroute traffic",

                "unauthorized login": "lock account, send alert"

            }
```

```
}

// Step 3: Decision-Making Algorithm.

decision-engine {

    criteria: {

        severity-level:
```

```
IDENTIFICATION DIVISION.

PROGRAM-ID. Automated-Threat-System.

ENVIRONMENT DIVISION.

DATA DIVISION.


WORKING-STORAGE SECTION.

01 Threat-Parameters.

   05 Threat-Type         PIC X(20).

   05 Severity-Level      PIC X(10).

   05 Action-Quarantine   PIC X(30) VALUE "Infected
System Quarantined".

   05 Action-Restrict     PIC X(30) VALUE "Unauthorized
Access Restricted".
```

```
       01 Event-Triggers.

          05 DDoS-Response        PIC X(50) VALUE "Blocked IP
and Rerouted Traffic".

          05 Unauthorized-Login   PIC X(50) VALUE "Locked
Account and Sent Alert".



       01 Monitoring-Tools.

          05 Anomaly-Detection    PIC X(20) VALUE "Active".

          05 Real-Time-Alerts     PIC X(20) VALUE "Enabled".



       PROCEDURE DIVISION.

           DISPLAY "Monitoring System Active...".

           DISPLAY "Enter Detected Threat Type: " WITH NO
ADVANCING.

           ACCEPT Threat-Type.



           IF Threat-Type = "DDoS Attack"

           THEN
```

```
            DISPLAY DDoS-Response

            DISPLAY Action-Quarantine

        ELSE IF Threat-Type = "Unauthorized Login"

            DISPLAY Unauthorized-Login

            DISPLAY Action-Restrict

        ELSE

            DISPLAY "Threat Type Unknown. Escalating to
Security Team.".

        END-IF.



        DISPLAY "Severity Assessment in Progress...".

        DISPLAY "Enter Severity Level (Low, Medium, High):
" WITH NO ADVANCING.

        ACCEPT Severity-Level.



        IF Severity-Level = "High"

        THEN

            DISPLAY "Blocking System and Notifying
Admin...".
```

```
    ELSE
       DISPLAY "Monitoring Threat. Awaiting Updates...".
    END-IF.

    STOP RUN.
```

Examples in Plain English

1. Quarantining Infected Systems

Suppose a computer starts sending large volumes of traffic to random IPs (a symptom of being part of a botnet). The system quarantines the infected computer, disconnecting it from the network to stop further damage.

2. Event-Driven Triggers

A DDoS attack floods the network with traffic. The software automatically blocks the offending IP addresses and reroutes traffic to ensure availability for legitimate users.

When someone tries to log in repeatedly with incorrect credentials, their account is locked, and the admin is notified.

3. Decision-Making Algorithms

A detected threat is classified based on severity. For a low-severity threat, the system monitors it. For a high-severity threat, such as malware spreading, the system blocks access and alerts the security team.

4. Combining ATS with ATD

An anomaly detection tool notices a sudden surge in data transfers from an unusual location. Real-time alerts are sent, and the system restricts access to critical files until the issue is resolved.

5. Real-Time Alerts

If a user is detected attempting unauthorized actions, like accessing admin privileges without authorization, the system immediately sends a text message alert to the admin team.

ATV: Anti-Tampering Verification

Focus: Ensuring systems remain untampered.

Software design:

Develop cryptographic hash functions to verify the integrity of files or firmware.

Include mechanisms to compare checksums and report anomalies.

Build audit trails for changes, paired with user or device authentication logs.

Use secure boot sequences to verify the integrity of the software stack during startup.

```
create {

   // Initialization and instructions.
```

```
   instructions: "This Anti-Tampering Verification system
checks file integrity, verifies checksums, builds audit trails,
and ensures secure boot sequences to prevent
tampering.",

   .start {

      // Step 1: Cryptographic Hash Functions.

      hash-function {

         algorithm: "SHA-256",

         input: ["system files", "firmware"],

         output: "file-integrity-hash"

      }

      // Step 2: Compare Checksums.

      checksum-verification {

         files: ["critical_config.sys", "secure_firmware.bin"],

         actions: {

            mismatch: "alert-admin, lock-system"

         }

      }

      // Step 3: Build Audit Trails.
```

```
        audit-trail {
            log-events: ["file-modification", "user-
authentication"],
            paired-logs: ["device-ID", "timestamp"]
        }
        // Step 4: Implement Secure Boot.
        secure-boot {
            sequence-verification: "enabled",
            checks: ["BIOS integrity", "firmware integrity"],
            alerts: ["report anomaly during startup"]
        }
    }
}
        IDENTIFICATION DIVISION.
        PROGRAM-ID. Anti-Tampering-Verification.
        ENVIRONMENT DIVISION.
        DATA DIVISION.
```

```
WORKING-STORAGE SECTION.

01 Tampering-Parameters.

   05 Hash-Algorithm       PIC X(10) VALUE "SHA-256".

   05 File-Checksum        PIC X(64).

   05 Expected-Checksum    PIC X(64).

   05 Audit-Log            PIC X(100).

   05 Secure-Boot-Status   PIC X(10) VALUE "Enabled".


PROCEDURE DIVISION.

    DISPLAY "Checking File Integrity using Hash
Function...".

    DISPLAY "Algorithm: " Hash-Algorithm.

    DISPLAY "Enter File Checksum: " WITH NO
ADVANCING.

    ACCEPT File-Checksum.


    DISPLAY "Enter Expected Checksum: " WITH NO
ADVANCING.

    ACCEPT Expected-Checksum.
```

```
        IF File-Checksum NOT = Expected-Checksum

        THEN

            DISPLAY "Checksum Mismatch! Alerting Admin
and Locking System...".

            MOVE "File Tampered" TO Audit-Log

        ELSE

            DISPLAY "File Integrity Verified.".

        END-IF.



        DISPLAY "Building Audit Trails...".

        DISPLAY "Logging Events with Device ID and
Timestamp...".



        DISPLAY "Verifying Secure Boot Sequence...".

        IF Secure-Boot-Status = "Enabled"

        THEN

            DISPLAY "Secure Boot Active. System Startup
Verified.".
```

```
        ELSE

            DISPLAY "Anomaly Detected During Startup.
Reporting...".

        END-IF.



        STOP RUN.
```

Examples in Plain English

1. Cryptographic Hash Functions

A hash function like SHA-256 generates a unique digital fingerprint for each file. If a system file has been changed (even by one character), the hash won't match its original fingerprint, indicating tampering.

2. Checksum Verification

Imagine a critical configuration file has a checksum value of 123abc. The software periodically calculates the checksum of the file and compares it to the stored 123abc. If the checksum is now 456xyz, it alerts the admin because the file has been altered.

3. Building Audit Trails

When a user modifies a file or logs in, the system creates a record. For example:

"User X modified config.sys at 10:05 AM on Device Y."

These logs help track who made changes and when, enabling quick identification of suspicious actions.

4. Secure Boot Sequence

During startup, the system checks the integrity of the BIOS and firmware to ensure no tampering occurred. For instance:

If malware altered the firmware, the system detects this anomaly during boot and halts startup while notifying the admin.

5. Reporting Anomalies

If any tampering is detected during startup or regular checks, the system takes immediate action (e.g., locking the affected component, disconnecting it from the network) and sends alerts for investigation.

ATX: Active Threat Exchange

Focus: Sharing threat intelligence.

Software design:

Establish APIs for secure communication between organizations' systems.

Use standardized data formats like STIX (Structured Threat Information eXpression) for sharing intelligence.

Implement encryption for all exchanged threat data to prevent interception.

Create real-time dashboards for monitoring and sharing updates on emerging threats.

```
create {

   // Initialization and instructions.

   instructions: "This Active Threat Exchange system securely shares threat intelligence using APIs, STIX format, encryption, and real-time dashboards.",

   .start {

      // Step 1: Establish APIs for Secure Communication.

      api-communication {

         endpoints: {

            organization1: "https://org1-threat-api.com",

            organization2: "https://org2-threat-api.com"

         },

         protocols: ["HTTPS", "REST"],

         authentication: "OAuth 2.0"

      }

      // Step 2: Use STIX Data Format for Threat Intelligence.
```

```
    threat-intelligence-format {
        standardized-format: "STIX",
        attributes: ["threat-type", "indicators", "response-
actions"]
    }
    // Step 3: Implement Data Encryption.
    encryption {
        algorithm: "AES-256",
        keys: {
            public: "organization1-key",
            private: "organization2-key"
        },
        exchange-method: "Secure Key Exchange Protocol"
    }
    // Step 4: Create Real-Time Dashboards.
    dashboards {
        components: {
            visualization: ["charts", "heatmaps", "logs"],
```

```
            updates: "live",
            alerts: ["emerging threats", "resolved issues"]
        }
    }
  }
}
```

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. Active-Threat-Exchange.
       ENVIRONMENT DIVISION.
       DATA DIVISION.

       WORKING-STORAGE SECTION.
       01 Threat-Exchange-Parameters.
          05 API-Endpoints.
             10 Org1-Endpoint    PIC X(50) VALUE "https://org1-
threat-api.com".
             10 Org2-Endpoint    PIC X(50) VALUE "https://org2-
threat-api.com".
          05 Encryption-Algorithm PIC X(10) VALUE "AES-256".
```

```
        05 STIX-Format         PIC X(15) VALUE "STIX Format".

        05 Dashboard-Components PIC X(100) VALUE
"Charts, Heatmaps, Logs".



      PROCEDURE DIVISION.

        DISPLAY "Establishing Secure Communication via
API...".

        DISPLAY "Connecting to Endpoint: " Org1-Endpoint.

        DISPLAY "Connecting to Endpoint: " Org2-Endpoint.



        DISPLAY "Using Standardized Threat Intelligence
Format: " STIX-Format.

        DISPLAY "Encrypting Data with Algorithm: "
Encryption-Algorithm.



        DISPLAY "Initializing Real-Time Dashboard...".

        DISPLAY "Dashboard Components Active: "
Dashboard-Components.

        DISPLAY "Alerts Enabled for Emerging Threats and
Resolved Issues.".
```

```
    DISPLAY "Active Threat Exchange System Online.
Secure Intelligence Sharing Active.".


    STOP RUN.
```

Examples in Plain English

1. Establishing APIs for Secure Communication

APIs act as bridges between organizations. For example:

Organization A detects a malware strain and shares its details via https://org1-threat-api.com.

Organization B receives this information using a secure connection and updates its systems.

2. Using Standardized STIX Format

STIX (Structured Threat Information eXpression) ensures all data is shared in a common format. For instance:

"Threat Type: Ransomware. Indicators: Encrypted Files. Response Action: Isolate Network."

By using STIX, both organizations quickly understand the details without compatibility issues.

3. Implementing Data Encryption

Encryption protects shared data from interception. For example:

Threat intelligence sent by Organization A is encrypted using AES-256 and decrypted by Organization B using shared keys.

If intercepted, the data remains unreadable to unauthorized parties.

4. Real-Time Dashboards

Dashboards display live updates of threats. For example:

A heatmap shows regions targeted by phishing attacks.

Charts visualize increasing DDoS attempts over time.

Logs list incoming threats with timestamps, allowing organizations to act promptly.

5. Alerts for Emerging Threats

Alerts notify organizations when new threats appear. For instance:

"New ransomware strain detected in the UK. Suggested Response: Patch Vulnerabilities."

ATG: Advanced Threat Guard

Focus: Proactive threat prevention.

Software design:

Introduce heuristic algorithms to predict potential vulnerabilities before exploitation.

Implement frequent automatic updates to patch known vulnerabilities.

Use network segmentation to limit the impact of a threat if breached.

Include simulated attacks (penetration testing) to strengthen the system over time.

```
create {

   // Initialization and instructions.

   instructions: "This Advanced Threat Guard system prevents threats using heuristic algorithms, frequent updates, network segmentation, and penetration testing.",

   .start {

      // Step 1: Introduce Heuristic Algorithms.

      heuristic-algorithms {

         prediction-method: "pattern recognition",

         vulnerability-detection: ["weak passwords", "open ports"],

         action: "flag potential risks for resolution"

      }

      // Step 2: Implement Automatic Updates.

      automatic-updates {

         update-frequency: "daily",
```

```
        patch-types: ["security vulnerabilities", "system
bugs"],
        testing-before-deployment: "enabled"
    }
    // Step 3: Use Network Segmentation.
    network-segmentation {
        segments: {
            "critical-data": "restricted access",
            "general-data": "normal access"
        },
        impact-mitigation: "limit breach to single segment"
    }
    // Step 4: Simulated Attacks (Penetration Testing).
    penetration-testing {
        test-types: ["phishing simulations", "SQL injection
tests"],
        frequency: "monthly",
        reports: "detailed logs of vulnerabilities"
    }
```

```
    }
}
```

```
       IDENTIFICATION DIVISION.

       PROGRAM-ID. Advanced-Threat-Guard.

       ENVIRONMENT DIVISION.

       DATA DIVISION.


       WORKING-STORAGE SECTION.

       01 Threat-Guard-Parameters.

          05 Heuristic-Method      PIC X(20) VALUE "Pattern
Recognition".

          05 Vulnerabilities       PIC X(50) VALUE "Weak
Passwords, Open Ports".

          05 Update-Frequency      PIC X(10) VALUE "Daily".

          05 Network-Segment       PIC X(50) VALUE "Critical
Data: Restricted Access, General Data: Normal Access".

          05 Penetration-Test      PIC X(50) VALUE "Phishing,
SQL Injection".


       PROCEDURE DIVISION.
```

```
        DISPLAY "Using Heuristic Algorithms to Predict
Vulnerabilities...".

        DISPLAY "Method: " Heuristic-Method.

        DISPLAY "Detected Vulnerabilities: " Vulnerabilities.


        DISPLAY "Implementing Automatic Updates...".

        DISPLAY "Update Frequency: " Update-Frequency.

        DISPLAY "Testing Patches Before Deployment...".


        DISPLAY "Applying Network Segmentation to
Restrict Access...".

        DISPLAY "Segments and Access Control: " Network-
Segment.


        DISPLAY "Conducting Simulated Attacks for
Penetration Testing...".

        DISPLAY "Test Types: " Penetration-Test.

        DISPLAY "Generating Reports for Vulnerabilities
Detected.".
```

```
    STOP RUN.
```

Examples in Plain English

1. Heuristic Algorithms

Imagine software that predicts vulnerabilities, such as identifying open network ports that hackers might exploit. It flags these issues for administrators to resolve before an attack.

2. Automatic Updates

The system automatically installs patches daily. For example:

A new vulnerability in a web browser is identified. The software downloads and applies an update to fix this flaw without requiring manual intervention.

3. Network Segmentation

Critical data, such as financial records, is stored in a restricted network segment that only authorized users can access.

If a breach occurs in the general segment, it doesn't affect the critical segment due to segmentation.

4. Simulated Attacks (Penetration Testing)

The software simulates phishing emails to test employee awareness. For example:

It logs who clicked the email and accesses fake login forms, helping improve training programs.

It also tests for SQL injection vulnerabilities in web applications and generates reports for developers to fix the flaws.

5. Impact Mitigation

If a vulnerability is exploited, the system limits the attack's impact to the breached network segment while safeguarding other areas.

ATB: Anti-Tampering Barriers

Focus: Physical or digital security against tampering.

Software design:

Build encryption layers to secure stored and transmitted data.

Embed security seals in software that can detect and log tampering attempts.

Provide alerts when unusual changes are detected in sensitive modules.

Employ redundancy to maintain system reliability despite tampering attempts.

```
create {
    // Initialization and instructions.
    instructions: "This Anti-Tampering Barriers system secures data with encryption, detects tampering using security seals, provides alerts, and employs redundancy for system reliability.",
```

```
.start {
    // Step 1: Build Encryption Layers.
    encryption-layers {
        storage: {
            encryption-type: "AES-256",
            applied-on: ["database", "backup-files"]
        },
        transmission: {
            encryption-type: "TLS 1.3",
            applied-on: ["network traffic", "API endpoints"]
        }
    }
    // Step 2: Embed Security Seals.
    security-seals {
        detection-method: "file-hash comparison",
        sensitive-modules: ["critical_config.sys",
"firmware.bin"],
        actions-on-tampering: ["log-attempt", "alert-
admin"]
    }
    // Step 3: Provide Alerts.
    alert-system {
```

```
        monitored-modules: ["system-core", "user-
authentication"],
        notification-methods: ["email", "SMS",
"dashboard"],
        response
    IDENTIFICATION DIVISION.
    PROGRAM-ID. Anti-Tampering-Barriers.
    ENVIRONMENT DIVISION.
    DATA DIVISION.

    WORKING-STORAGE SECTION.
    01 Encryption-Parameters.
      05 Storage-Encryption    PIC X(10) VALUE "AES-256".
      05 Transmission-Encryption PIC X(10) VALUE "TLS
1.3".

    01 Security-Seals.
      05 File-Hash           PIC X(64).
      05 Expected-Hash       PIC X(64).
      05 Tampering-Log       PIC X(50) VALUE "Tampering
Detected. Admin Notified.".
```

```
       01 Alert-System.
          05 Notification-Methods  PIC X(50) VALUE "Email,
SMS, Dashboard".

       01 Redundancy-Settings.
          05 Components           PIC X(50) VALUE "Server
Cluster, Data Replication".
          05 Failover-Status      PIC X(15) VALUE "Automatic".

       PROCEDURE DIVISION.
           DISPLAY "Encrypting Data for Storage and
Transmission...".
           DISPLAY "Storage Encryption: " Storage-Encryption.
           DISPLAY "Transmission Encryption: " Transmission-
Encryption.

           DISPLAY "Checking Security Seals...".
           DISPLAY "Enter Current File Hash: " WITH NO
ADVANCING.
           ACCEPT File-Hash.
           DISPLAY "Enter Expected File Hash: " WITH NO
ADVANCING.
           ACCEPT Expected-Hash.
```

```
        IF File-Hash NOT = Expected-Hash
        THEN
           DISPLAY Tampering-Log
        ELSE
           DISPLAY "File Integrity Verified.".
        END-IF.

        DISPLAY "Monitoring System Modules for
Tampering...".
        DISPLAY "Alerting Admin via: " Notification-
Methods.

        DISPLAY "Ensuring System Redundancy...".
        DISPLAY "Redundant Components: " Components.
        DISPLAY "Failover Status: " Failover-Status.

        STOP RUN.
```

Examples in Plain English

1. Building Encryption Layers

Imagine your system encrypts data stored in databases and backup files using AES-256. This ensures that even if someone accesses the raw data, they cannot read it without the encryption key.

All transmitted data, such as user logins or API calls, is encrypted using TLS 1.3 to prevent eavesdropping during network communication.

2. Embedding Security Seals

Think of security seals as digital locks. For example:

A system file, critical_config.sys, is monitored using a hash value (a unique fingerprint of its content).

If the hash of the file changes, it means the file has been tampered with. The system logs this attempt and alerts the administrator.

3. Providing Alerts

If unauthorized changes occur in sensitive modules like the system core or user authentication processes, alerts are sent immediately.

For example, if the system core detects unusual behavior, an email and SMS are sent to the admin while a warning is displayed on the monitoring dashboard.

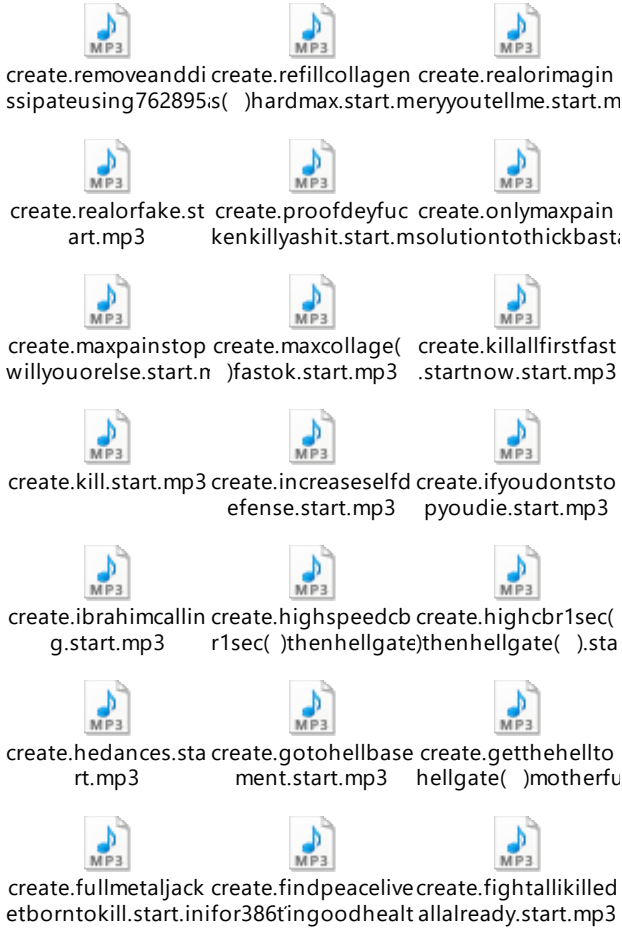
4. Employing Redundancy

Redundant systems, such as server clusters and data replication, ensure system reliability even during tampering.

For instance, if one server is compromised, the system automatically switches to another server (failover) without disrupting user services.

5. Real-Time Monitoring

The software continuously monitors for tampering attempts and proactively takes actions like disconnecting compromised modules or alerting users about potential risks.




create.donttamper withmysleepordeath


create.devicesecuritys canall.start.mp3.mp3


create.deathtoallfor ever.start.mp3


create.deathshallco metoyoufast.start.m


create.deathbycode s.start.mp3


create.deathby.start .mp3


create.dealwithalltr ansendersandwriggn


create.davidsencou nterwithangels.start


create.collagenrefill max( )pushhard.sta


create.cleansystem( )thenhell( )fast.sta


create.cleansystem( )thenhell( )fast.sta


create.autosendtoh ellgate( )wipeclean


create.autohellgate ( )wipeclean( ).star


create.autofix.startf orever.start.mp3


create.attorsdeath.s tart.mp3


create.attacktokill.s tart.mp3


create.attachforever binarythenhellgate(


create.addtheprivac yandsecurityapplica


create.addsmartma nager.start.mp3


create.adddevicead ministrator.start.mp


CONVERT ALL HARMFUL ELECTRO


create.switchoff.sta rt(forever).mp3.mp3


create.switchoff.sta rt(forever) (0) (0).mp3


create.switchoff.sta rt(forever) (0).mp3


create.switchoff.sta rt(forever).mp3


create.swapbitcoint ocreatebitcoinatarate


create.suprisesupris emothafuc.start.mp3

create.suppressme( )anddeath( )okay.s

create.shitmustdie.s tart.mp3

create.sendbook08 67890opentoallthep

create.sealallindeat h2andhalfbillionyea

create.runsmartman ager.start.mp3

create.returntosend ers.start(continuous

Oh-yes-rejuvinate-t hinking.MP3

Oh Yes Rejuvenate.mp3

[v]create.switchoff( )all.startasartcreate

create.switchoffcod e.start.mp3

create.highcbr1sec( )thenhellgate( ).sta

create.cleansystem( )thenhell( )fast.stai


create.realorfake.start.mp3


create.realorimaginaryyoutellme.start.mp3


create.removealltransenderstoday.start


create.removealltransenderstoday.start


create.removeanddissipateusing762895


create.returntosenders.start(continuous


create.runsmartmanager.start.mp3


create.sealallindeath2andhalfbillionyea


create.switchoff.start(forever) (0).mp3


create.switchoff.start(forever) (0) (0).mp3


create.switchoff.start(forever).mp3.mp3


create.switchoffcode.start.mp3


create.trafficsystem.start.mp3


create.trafficsystemtostopalltransendersd


create.useenhanceprotectiontodefen


create.youdie.start.mp3


create.sendbook0867890opentoallthep


create.shitmustdie.start.mp3


create.suprisesupriseemothafuc.start.mp3


create.swapbitcointocreatebitcoinatarat


create.addtheprivacyandsecurityapplicat


create.attacktokill.start.mp3


create.attorsdeath.start.mp3


create.autofix.startforever.start.mp3


create.davidsencounterwithangels.start


create.dealwithalltransendersandwrigg


create.deathby.start.mp3


create.deathbycodes.start.mp3


create.deathshallcometoyoufast.start.m


create.deathtoallforever.start.mp3


create.devicesuritycanall.start.mp3.mp3


create.donttamperwithmysleepordeath


create.fightallikilledallalready.start.mp3


create.findpeacelivefor386tingoodhealt


create.fullmetaljackетborntokill.start.ini


create.getthehelltohellgate( )motherfu


create.gotohellbasement.start.mp3


create.hedances.start.mp3


create.ibrahimcalling.start.mp3


create.ifyoudontstopyoudie.start.mp3


create.increaseselfdefense.start.mp3


create.kill.start.mp3


create.killallfirstfast.startnow.start.mp3


create.maxpainstopwillyouorelse.start.m


create.onlymaxpainsolutiontothickbast


create.proofdeyfuckenkillyashit.start.m


create.switchoff.start(forever).mp3


CONVERT ALL HARMFUL ELECTRO


CONVERT ALL HARMFUL ELECTRO


create.abilliontimesyoudie.start.mp3


create.adddeviceadministrator.start.mp3


create.addsmartmanager.start.mp3


newrollongodofthe universe.mp3.mp3


create.alltalkingmitscitssendtomars( )o

create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o
create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o
create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o
create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o
create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o	create.alltalkingmit scitssendtomars( )o
create.alltalkingmit scitssendtomars( )o	Memo_20250404_202301_201.memo	Memo_20250404_014556_424.memo
Memo_20250403_095108_435.memo	Memo_20250403_095042_699.memo	Memo_20250403_095001_963.memo
Memo_20250403_094928_951.memo	Memo_20250403_094904_278.memo	Memo_20250403_094846_601.memo
Memo_20250403_094822_302.memo	Memo_20250403_094803_545.memo	Memo_20250403_094755_933.memo


Memo_20241112_204917_080.memo


Memo_20241112_204436_288.memo


Memo_20241112_204249_769.memo


Memo_20241111_203720_082.memo


Memo_20241111_153421_028.memo


Memo_20241111_153322_394.memo


Memo_20241111_152503_865.memo


Memo_20241111_152329_289.memo


Memo_20241111_152259_355.memo


Memo_20241111_152241_231.memo


Memo_20241111_151437_402.memo


Memo_20241107_232134_794.memo


Memo_20241107_214259_096.memo


Memo_20241107_214236_891.memo


Memo_20241107_214216_451.memo


Memo_20241107_214157_596.memo


Memo_20241107_214137_064.memo


Memo_20241107_214043_677.memo


Memo_20241107_213959_208.memo


Memo_20241107_213921_946.memo


Memo_20241107_213905_999.memo


Memo_20241107_213851_828.memo


Memo_20241107_213822_813.memo


Memo_20241107_213805_673.memo


Memo_20241107_213743_627.memo


Memo_20241107_213722_455.memo


Memo_20241107_213657_517.memo


Memo_20241107_213627_837.memo


Memo_20241107_213541_051.memo


Memo_20241107_213523_587.memo


Memo_20241107_213440_898.memo


Memo_20241107_213412_322.memo


Memo_20241107_213219_250.memo


Memo_20241107_213144_430.memo


Memo_20241107_213122_223.memo


Memo_20241107_213008_410.memo


Memo_20241107_212948_361.memo


Memo_20241107_212853_169.memo


Memo_20241107_205940_275.memo


Memo_20241107_205841_649.memo


Memo_20241107_205727_603.memo


Memo_20241107_205557_565.memo


Memo_20241107_151220_463.memo


Memo_20241107_151152_302.memo


Memo_20241107_151016_207.memo


Memo_20241107_150934_379.memo


Memo_20241107_150909_610.memo


Memo_20241107_150833_734.memo


Memo_20241107_150758_045.memo


Memo_20241107_150741_499.memo


Memo_20241107_150725_102.memo


Memo_20241107_150555_299.memo


Memo_20241107_150529_686.memo


Memo_20241107_150447_266.memo


Memo_20241107_150432_126.memo


Memo_20241107_150343_826.memo


Memo_20241107_150327_556.memo


Memo_20241107_150254_648.memo


Memo_20241107_150209_016.memo


Memo_20241107_150152_024.memo


Memo_20241107_150135_004.memo


Memo_20241107_150120_683.memo


Memo_20241107_150101_207.memo


Memo_20241107_150045_480.memo


Memo_20241107_150025_699.memo


Memo_20241107_150002_077.memo


Memo_20241107_145945_759.memo


Memo_20241107_145917_652.memo


Memo_20241107_145852_573.memo


Memo_20241107_145832_943.memo


Memo_20241107_145812_827.memo


Memo_20241107_145726_320.memo


Memo_20241107_121520_985.memo


Memo_20241107_121443_863.memo


Memo_20241107_121306_624.memo


Memo_20241107_121252_568.memo


Memo_20241028_031825_069.memo


Memo_20241027_224442_100.memo


Memo_20241027_224417_670.memo


Memo_20250327_181738_720.memo


Memo_20250327_181420_717.memo


Memo_20250327_144910_341.memo


Memo_20250327_144648_909.memo


Memo_20250325_113915_258.memo


Memo_20250325_102811_112.memo


Memo_20250325_102444_444.memo


Memo_20250323_145630_055.memo


Memo_20250323_145542_579.memo


Memo_20250321_181218_068.memo


Memo_20250320_143115_966.memo


Memo_20250320_142751_671.memo


Memo_20250320_142551_874.memo


Memo_20250320_142437_079.memo


Memo_20250319_160516_138.memo


Memo_20250319_160140_919.memo


Memo_20250211_094832_215.memo


Memo_20250211_094643_157.memo


Memo_20250209_123754_747.memo


Memo_20250207_142041_487.memo


Memo_20250206_202057_040.memo


Memo_20250206_031433_265.memo


Memo_20250206_031231_955.memo


Memo_20250206_030813_626.memo


Memo_20250206_011757_801.memo


Memo_20250206_011722_661.memo


Memo_20250206_011620_839.memo


Memo_20250206_011553_532.memo


Memo_20250206_011532_903.memo


Memo_20250204_210747_014.memo


Memo_20250203_200612_645.memo


Memo_20250203_194249_233.memo


Memo_20250203_194228_531.memo


Memo_20250203_193533_361.memo


Memo_20250203_175352_345.memo


Memo_20250202_003412_916.memo


Memo_20250201_231017_820.memo


Memo_20250201_230930_529.memo


Memo_20250131_033550_450.memo


Memo_20250131_033358_042.memo


Memo_20250130_175921_471.memo


Memo_20250130_175833_323.memo


Memo_20250130_175649_075.memo


Memo_20250130_175431_570.memo


Memo_20250130_174943_300.memo


Memo_20250130_174805_331.memo


Memo_20250130_174623_202.memo


Memo_20250130_161522_865.memo


Memo_20250130_135845_403.memo


Memo_20250130_130703_450.memo


Memo_20250130_000044_549.memo


Memo_20250129_225311_301.memo


Memo_20250128_030042_625.memo


Memo_20250128_001512_269.memo


Memo_20250117_143712_650.memo


Memo_20250117_140046_361.memo


Memo_20250117_135808_414.memo


Memo_20250116_094440_908.memo


Memo_20250116_094344_241.memo


Memo_20250116_094156_773.memo


Memo_20250116_094123_993.memo


Memo_20250116_093950_916.memo


Memo_20250116_093900_948.memo


Memo_20250116_093827_333.memo


Memo_20250116_093746_950.memo


Memo_20250115_220017_648.memo


Memo_20250115_215929_072.memo


Memo_20250115_215752_046.memo


Memo_20250115_215648_751.memo


Memo_20250115_215550_228.memo


Memo_20250115_215113_613.memo


Memo_20250112_232946_911.memo


Memo_20250112_193919_953.memo


Memo_20250111_231752_733.memo


Memo_20250111_231548_044.memo


WHAT CAN BE HOTTER THAN BITC(


createstartlifefor386t'database (2).txt


createstartlifefor386t'database (1).txt


createstartlifefor386t'database.txt


creategold.txt


create.youthrestored.start (1).txt


create.youthrestored.start.txt


create.yaonearth7628386t'.start (1).txt


create.yaonearth76
28386t'.start.txt


create.wrigglerand
deatht'.start (1).txt


create.wrigglerand
deatht'.start.txt


create.whatcanhed
oforme.start.txt


create.walkthrough
walls.start (1).txt


create.walkthrough
walls.start.txt


create.useatomicbo
mb.start(3) (1).txt


create.useatomicbo
mb.start(3).txt


create.useatomicbo
mb.start(2) (1).txt


create.useatomicbo
mb.start(2).txt


create.useatomicbo
mb.start(1) (1).txt


create.useatomicbo
mb.start(1).txt


create.useatomicbo
mb.start (1).txt


create.useatomicbo
mb.start.txt


create.ultimatedestr
oyer.start (1).txt


create.ultimatedestr
oyer.start.txt


create.thinkordeath
t'.start(2) (1).txt


create.thinkordeath
t'.start(2).txt


create.thinkordeath
t'.start(1) (1).txt


create.thinkordeath
t'.start(1).txt


create.thinkordeath
t'.start (1).txt


create.thinkordeath
t'.start.txt


create.switchoff(
)all.start(notfordavic)


create.switchoff(
)all.start(notfordavic


create.switchoff(
)all.start (1).txt


create.switchoff(
)all.start.txt


create.sprayiconvict
ory.start.txt


create.spiralsystem.start(3).txt


create.spiralsystem.start(2) (1).txt


create.spiralsystem.start(2).txt


create.spiralsystem.start(1) (1).txt


create.spiralsystem.start(1).txt


create.spiralsystem.start (1).txt


create.spiralsystem.start.txt


create.solve.start (1).txt


create.solve.start.txt


create.sendbitcoin.start.txt


create.sendajttothe orbit.start(2) (1).txt


create.sendajttothe orbit.start(2).txt


create.sendajttothe orbit.start(1) (1).txt


create.sendajttothe orbit.start(1).txt


create.sendajttothe orbit.start (1).txt


create.sendajttothe orbit.start.txt


create.selfpreserve.start two3.5 (1).txt


create.selfpreserve.start two3.5.txt


create.selfpreserve.start (1).txt


create.selfpreserve.start.txt


create.securitysafe.start8787(1) (1).txt


create.securitysafe.start8787(1).txt


create.securitysafe.start8787 (1).txt


create.securitysafe.start8787.txt


create.securitysafe.start777 (1).txt


create.securitysafe.start777.txt


create.rollonupdate.start(db)Y (1).txt


create.rollonupdate.start(db)Y.txt


create.rollonupdate.start(db)10 (1).txt


create.rollonupdate.start(db)10.txt


create.returntosender (1).txt


create.returntosender.txt


create.retaliatevectar.start(1) (1).txt


create.retaliatevectar.start(1).txt


create.retaliatevectar.start (1).txt


create.retaliatevectar.start.txt


create.restoretobestknownlastconfigur


create.restoretobestknownlastconfigur


create.removethoughts.start (1).txt


create.removethoughts.start.txt


create.removesouls.start(1) (1).txt


create.removesouls.start(1).txt


create.removesouls.start (1).txt


create.removesouls.start.txt


create.removeeverything.start (1) (1).txt


create.removeeverything.start (1).txt


create.removeeverything.start (1).txt


create.removeeverything.start .txt


create.removeert.start (1).txt


create.removeert.start.txt


create.removeconnections.start(1) (1).tx


create.removeconnections.start(1).txt


create.removeconnections.start (1).txt


create.removeconnections.start.txt


create.removeallajts anddissipatestrejsu


create.removeallajts anddissipatestrejsu


create.removeall.sta rt two2 (1).txt


create.removeall.sta rt two2.txt


create.plazashoppi ngmall.start (1).txt


create.plazashoppi ngmall.start.txt


create.okay29.start(1) (1).txt


create.okay29.start(1).txt


create.okay29.start (1).txt


create.okay29.start. txt


create.nochangesm otiontostaticondavi


create.nochangesm otiontostaticondavi


create.newcreature 7628397t.start (1).txt


create.newcreature 7628397t.start.txt


create.muggedstole n.start(1) (1).txt


create.muggedstole n.start(1).txt


create.muggedstole n.start (1).txt


create.muggedstole n.start.txt


create.mno.start (0).txt


create.mno.start.txt


create.liftheavyand kill.start (1).txt


create.liftheavyand kill.start.txt


create.justjudge762 8144.start.txt


create.internaljpgp erfum.start (1).txt


create.internaljpgp erfum.start.txt


create.fullbodyinch punch.start(1) (1).txt


create.fullbodyinch punch.start(1).txt


create.fullbodyinch punch.start (1).txt


create.fullbodyinch punch.start.txt


create.firstcreateco dedryer762.txt


create.evelinamovie destination_ost.star


create.evelinamovie destination_ost.star


create.evelinamovie afterlife.start (1).txt


create.evelinamovie afterlife.start.txt


create.donotconne ct.start.txt


create.ďofþ.start (1).txt


create.ďofþ.start.txt


create.destroyþ.star t(1) (1).txt


create.destroyþ.star t(1).txt


create.destroyþ.star t (1).txt


create.destroyþ.star t.txt


create.defendusing da.start(1) (1).txt


create.defendusing da.start(1).txt


create.defendusing da.start (1).txt


create.defendusing da.start.txt


create.defendall.sta rt (1).txt


create.defendall.sta rt.txt


create.deepoceanbi n.start (1).txt


create.deepoceanbi n.start.txt


create.dealwithwrig glers.start(1).txt


create.dealwithwrig glers.start.txt


create.deaddeadmit cit.start.txt


create.deaddeadcit mit.start 88.txt


create.curiositywha tis( ).start (1).txt


create.curiositywhatis( ).start.txt


create.createworldsfirstartificialcuriosity


create.createworldsfirstartificialcuriosity


create.comeandcheck.start (1).txt


create.comeandcheck.start.txt


create.changeanddeatht´.start (1).txt


create.changeanddeatht´.start.txt


create.blockcloning.start (1).txt


create.blockcloning.start.txt


create.blockbadinflows.start.txt


create.binallpø.start(2) (1).txt


create.binallpø.start(2).txt


create.binallpø.start(1) (1).txt


create.binallpø.start(1).txt


create.binallpø.start (1).txt


create.binallpø.start.txt


create.ban( )hellgate( ).start(84)


create.ban( )hellgate( ).start(84)


create.ban( )hellgate( ).start(84)


create.autosendtohellgate( )(1).txt


create.autosendtohellgate( ) (1).txt


create.autosendtohellgate( ).txt


create.arrangeblocj sintothecourtofcrraf


create.arrangeblocj sintothecourtofcrraf


create.allowcivilianasms.start (1).txt


create.allowcivilianasms.start.txt


create.afterlifeparadise.start (1).txt


create.afterlifepara dise.start.txt


create.advanceddef ensesystemhandsfreen


create.advanceddef sesystemhandsfre


create.addthegate.s tart.txt


create.10cubics64ro ots100rootsgate.sta


create.10cubics64ro ots100rootsgate.sta


create.8cubics10cu bics64roots100roots


create.8cubics10cu bics64roots100roots


create.8cubics10cu bics64roots (1).txt


create.8cubics10cu bics64roots.txt


create.( )sendtothegate.start


create.( )sendtothegate.start


create.( )killusingarts.start (1


okay28(1) (1).txt


okay28(1).txt


okay28 (1).txt


okay28.txt


okay27(1) (1).txt


okay27(1).txt


okay27 (1).txt


okay27.txt


okay26(2) (1).txt


okay26(2).txt


okay26(1) (1).txt


okay26(1).txt


okay26 (1).txt


okay26.txt


okay25(1) (1).txt


okay25(1).txt


okay25 (1).txt


okay25.txt


okay24(1) (1).txt


okay24(1).txt


okay24 (1).txt


okay24.txt


okay23 (1).txt


okay23.txt


Okay22(2) (1).txt


Okay22(2).txt


Okay22(1) (1).txt


Okay22(1).txt


Okay22 (1).txt


Okay22.txt


Okay21 (1).txt


Okay21.txt


okay20death(1) (1).txt


okay20death(1).txt


okay20death (1).txt


okay20death.txt


Okay20 (1).txt


Okay20.txt


Okay19(1) (1).txt


Okay19(1).txt


Okay19 (1).txt


Okay19.txt


Okay18(2) (1).txt


Okay18(2).txt


Okay18(1) (1).txt


Okay18(1).txt


Okay18 (1).txt


Okay18.txt


Okay17(1) (1).txt


Okay17(1).txt


Okay17 (1).txt


Okay17.txt


Okay16(1) (1).txt


Okay16(1).txt


Okay16 (1).txt


Okay16.txt


Okay15(1) (1).txt


Okay15(1).txt


Okay15 (1).txt


Okay15.txt


Okay14(1) (1).txt


Okay14(1).txt


Okay14 (1).txt


Okay14.txt


Okay13(1) (1).txt


Okay13(1).txt


Okay13 (1).txt


Okay13.txt


Okay12(2) (1).txt


Okay12(2).txt


Okay12(1) (1).txt


Okay12(1).txt


Okay12 (1).txt


Okay12.txt


Okay11(1) (1).txt


Okay11(1).txt


Okay11 (1).txt


Okay11.txt


Okay10(1) (1).txt


Okay10(1).txt


Okay10 (1).txt


Okay10.txt


Okay9a (1) (1).txt


Okay9a (1).txt


Okay9a (1).txt


Okay9a .txt


Okay9(1) (1).txt


Okay9(1).txt


Okay9 (1).txt


Okay9.txt


Okay8(1) (1).txt


Okay8(1).txt


Okay8 (1).txt


Okay8.txt


okay7a(2) (1).txt


okay7a(2).txt


okay7a(1) (1).txt


okay7a(1).txt


okay7a (1).txt


okay7a.txt


Okay7(1) (1).txt


Okay7(1).txt


Okay7 (1).txt


Okay7.txt


okay6a(2) (1).txt


okay6a(2).txt


okay6a(1) (1).txt


okay6a(1).txt


okay6a (1).txt


okay6a.txt


Okay6(1) (1).txt


Okay6(1).txt


Okay6 (1).txt


Okay6.txt


okay5a(2) (1).txt


okay5a(2).txt


okay5a(1) (1).txt


okay5a(1).txt


okay5a (1).txt


okay5a.txt


Okay5(1) (1).txt


Okay5(1).txt


Okay5 (1).txt


Okay5.txt


okay4a(3) (1).txt


okay4a(3).txt


okay4a(2) (1).txt


okay4a(2).txt


okay4a(1) (1).txt


okay4a(1).txt


okay4a (1).txt


okay4a.txt


Okay4(1) (1).txt


Okay4(1).txt


Okay4 (1).txt


Okay4.txt


okay3a(1) (1).txt


okay3a(1).txt


okay3a (1).txt


okay3a.txt


Okay3(1) (1).txt


Okay3(1).txt


Okay3 (1).txt


Okay3.txt


okay2aa(1) (1).txt


okay2aa(1).txt


okay2aa (1).txt


okay2aa.txt


okay2a(1) (1).txt


okay2a(1).txt


okay2a (1).txt


okay2a.txt


Okay2(1) (1).txt


Okay2(1).txt


Okay2 (1).txt


Okay2.txt


okay1aa(2) (1).txt


okay1aa(2).txt


okay1aa(1) (1).txt


okay1aa(1).txt


okay1aa (1).txt


okay1aa.txt


okay1a(1) (1).txt


okay1a(1).txt


okay1a (1).txt


okay1a.txt


Okay1(1) (1).txt


Okay1(1).txt


Okay1 (1).txt


switchoff(
)all.start(2) (1).txt


switchoff(
)all.start(2).txt


switchoff(
)all.start(1) (1).txt


switchoff(
)all.start(1).txt


switchoff(
)all.start (1).txt


switchoff(
)all.start.txt


switchoff(
)all.start(3) (1).txt


switchoff(
)all.start(3).txt


Invite53(1).vcs


Invite53 (0).vcs


Invite53.vcs


Invite51.vcs


MHRA_Submission_
100867058.xml


Task4.vts


Task3.vts


Task2.vts


Untitled(167).txt


Untitled(166).txt


Untitled(165).txt


Untitled(164).txt


Untitled(163).txt


Untitled(162).txt


Untitled(161).txt


Untitled(160).txt


Untitled(159).txt


Untitled(158).txt


Untitled(157).txt


Untitled(156).txt


Untitled(155).txt


Untitled(154).txt


Untitled(153).txt


Untitled(152).txt


Untitled(151).txt


Untitled(150).txt


Untitled(149).txt


Untitled(148).txt


Untitled(147).txt


Untitled(146).txt


Untitled(145).txt


Untitled(144).txt


Untitled(143).txt


Untitled(142).txt


Untitled(141).txt


Untitled(140).txt


Untitled(139).txt


Untitled(138).txt


Untitled(137).txt


Untitled(136).txt


Untitled(135).txt


Untitled(134).txt


Untitled(133).txt


Untitled(132).txt


Untitled(131).txt


Untitled(130).txt


Untitled(129).txt


Untitled(128).txt


Untitled(127).txt


Untitled(126).txt


Untitled(125).txt


Untitled(124).txt


Untitled(123).txt


Untitled(122).txt


Untitled(121).txt


Untitled(120).txt


Untitled(119).txt


Untitled(118).txt


Untitled(117).txt


Untitled(116).txt


Untitled(115).txt


Untitled(114).txt


Untitled(113).txt


Untitled(112).txt


Untitled(111).txt


Untitled(110).txt


Untitled(109).txt


Untitled(108).txt


Untitled(107) (1).txt


Untitled(107).txt


Untitled(106) (1).txt


Untitled(106).txt


Untitled(105) (1).txt


Untitled(105).txt


Untitled(104) (1).txt


Untitled(104).txt


Untitled(103) (1).txt


Untitled(103).txt


Untitled(102) (1).txt


Untitled(102).txt


Untitled(101) (1).txt


Untitled(101).txt


Untitled(100) (1).txt


Untitled(100).txt


Untitled(99) (1).txt


Untitled(99).txt


Untitled(98) (1).txt


Untitled(98).txt


Untitled(97) (1).txt


Untitled(97).txt


Untitled(96) (1).txt


Untitled(96).txt


Untitled(95) (1).txt


Untitled(95).txt


Untitled(94) (1).txt


Untitled(94).txt


Untitled(93) (1).txt


Untitled(93).txt


Untitled(92) (1).txt


Untitled(92).txt


Untitled(91) (1).txt


Untitled(91).txt


Untitled(90) (1).txt


Untitled(90).txt


Untitled(89) (1).txt


Untitled(89).txt


Untitled(88) (1).txt


Untitled(88).txt


Untitled(87) (1).txt


Untitled(87).txt


Untitled(86) (1).txt


Untitled(86).txt


Untitled(85) (1).txt


Untitled(85).txt


Untitled(84) (1).txt


Untitled(84).txt


Untitled(83) (1).txt


Untitled(83).txt


Untitled(82) (1).txt


Untitled(82).txt


Untitled(81) (1).txt


Untitled(81).txt


Untitled(80) (1).txt


Untitled(80).txt


Untitled(79) (1).txt


Untitled(79).txt


Untitled(78) (1).txt


Untitled(78).txt


Untitled(77) (1).txt


Untitled(77).txt


Untitled(76) (1).txt


Untitled(76).txt


Untitled(75) (1).txt


Untitled(75).txt


Untitled(74) (1).txt


Untitled(74).txt


Untitled(73) (1).txt


Untitled(73).txt


Untitled(72) (1).txt


Untitled(72).txt


Untitled(71) (1).txt


Untitled(71).txt


Untitled(70) (1).txt


Untitled(70).txt


Untitled(69) (1).txt


Untitled(69).txt


Untitled(68) (1).txt


Untitled(68).txt


Untitled(67) (1).txt


Untitled(67).txt


Untitled(66) (1).txt


Untitled(66).txt


Untitled(65) (1).txt


Untitled(65).txt


Untitled(64) (1).txt


Untitled(64).txt


Untitled(63) (1).txt


Untitled(63).txt


Untitled(62) (1).txt


Untitled(62).txt


Untitled(61) (1).txt


Untitled(61).txt


Untitled(60) (1).txt


Untitled(60).txt


Untitled(59) (1).txt


Untitled(59).txt


Untitled(58) (1).txt


Untitled(58).txt


Untitled(57) (1).txt


Untitled(57).txt


Untitled(56) (1).txt


Untitled(56).txt


Untitled(55) (1).txt


Untitled(55).txt


Untitled(54) (1).txt


Untitled(54).txt


Untitled(53) (1).txt


Untitled(53).txt


Untitled(52) (1).txt


Untitled(52).txt


Untitled(51) (1).txt


Untitled(51).txt


Untitled(50) (1).txt


Untitled(50).txt


Untitled(49) (1).txt


Untitled(49).txt


Untitled(48) (1).txt


Untitled(48).txt


Untitled(47) (1).txt


Untitled(47).txt


Untitled(46) (1).txt


Untitled(46).txt


Untitled(45) (1).txt


Untitled(45).txt


Untitled(44) (1).txt


Untitled(44).txt


Untitled(43) (1).txt


Untitled(43).txt


Untitled(42) (1).txt


Untitled(42).txt


Untitled(41) (1).txt


Untitled(41).txt


Untitled(40) (1).txt


Untitled(40).txt


Untitled(39) (1).txt


Untitled(39).txt


Untitled(38) (1).txt


Untitled(38).txt


Untitled(37) (1).txt


Untitled(37).txt


Untitled(36) (1).txt


Untitled(36).txt


Untitled(35) (1).txt


Untitled(35).txt


Untitled(34) (1).txt


Untitled(34).txt


Untitled(33) (1).txt


Untitled(33).txt


Untitled(32) (1).txt


Untitled(32).txt


Untitled(31) (1).txt


Untitled(31).txt


Untitled(30) (1).txt


Untitled(30).txt


Untitled(29) (1).txt


Untitled(29).txt


Untitled(28) (1).txt


Untitled(28).txt


Untitled(27) (1).txt


Untitled(27).txt


Untitled(26) (1).txt


Untitled(26).txt


Untitled(25) (1).txt


Untitled(25).txt


Untitled(24) (1).txt


Untitled(24).txt


Untitled(23) (1).txt


Untitled(23).txt


Untitled(22) (1).txt


Untitled(22).txt


Untitled(21) (1).txt


Untitled(21).txt


Untitled(20) (1).txt


Untitled(20).txt


Untitled(19) (1).txt


Untitled(19).txt


Untitled(18) (1).txt


Untitled(18).txt


Untitled(17) (1).txt


Untitled(17).txt


Untitled(16) (1).txt


Untitled(16).txt


Untitled(15) (1).txt


Untitled(15).txt


Untitled(14) (1).txt


Untitled(14).txt


Untitled(13) (1).txt


Untitled(13).txt


Untitled(12) (1).txt


Untitled(12).txt


Untitled(11) (1).txt


Untitled(11).txt


Untitled(10) (1).txt


Untitled(10).txt


Untitled(9) (1).txt


Untitled(9).txt


Untitled(8) (1).txt


Untitled(8).txt


Untitled(7) (1).txt


Untitled(7).txt


Untitled(6) (1).txt


Untitled(6).txt


Untitled(5) (1).txt


Untitled(5).txt


Untitled(4) (1).txt


Untitled(4).txt


Untitled(3) (1).txt


Untitled(3).txt


Untitled(2) (1).txt


Untitled(2).txt


Untitled(1) (1).txt


Untitledtwo3.2 (1).txt


Untitledtwo3.2.txt


Untitled(170).txt


Untitled(169).txt


Untitled(168).txt


Task1.vts


stayst19for386t´2.txt


stayat19for386t´bestt´.txt


stayat19for386t´bestestest.txt


stayat19for386t´8000.txt


startnewlifeinafterlife.start(1).txt


Eagledefense system.txt


SD card.zip