1. Cloud strike breach scenario

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define MAX_LINE_LENGTH 256
#define MAX_USERS 100

typedef struct {
    char username[MAX_LINE_LENGTH];
    char password[MAX_LINE_LENGTH];
} UserCredential;

void evaluatePasswordStrength(const char *password, char *strength) {
    int length = strlen(password);
    int hasUpper = 0, hasLower = 0, hasDigit = 0, hasSpecial = 0;

    if (length < 8) {
        strcpy(strength, "Weak");
        return;
    }

    for (int i = 0; i < length; i++) {
        if (isupper(password[i])) hasUpper = 1;
        else if (islower(password[i])) hasLower = 1;
        else if (isdigit(password[i])) hasDigit = 1;
        else if (ispunct(password[i])) hasSpecial = 1;
    }

    if (hasUpper && hasLower && hasDigit && hasSpecial) {
        strcpy(strength, "Strong");
    } else if ((hasUpper || hasLower) && (hasDigit || hasSpecial)) {
        strcpy(strength, "Moderate");
    } else {
        strcpy(strength, "Weak");
    }
}

void identifyVulnerabilities(UserCredential users[], int userCount) {
    printf("\nPotential Vulnerabilities:\n");
    for (int i = 0; i < userCount; i++) {
        if (strcmp(users[i].password, "password123") == 0 ||
            strcmp(users[i].password, "12345678") == 0 ||
            strcmp(users[i].password, "weakpassword") == 0) {
            printf("- %s: Weak password (common pattern)\n", users[i].username);
        }
        if (strlen(users[i].password) < 8) {
            printf("- %s: Weak password (too short)\n", users[i].username);
        }
    }
}

int main() {
    UserCredential users[MAX_USERS];
    int userCount = 0;
```

```
printf("Enter the number of users: ");
scanf("%d", &userCount);
getchar();

for (int i = 0; i < userCount; i++) {
    printf("Enter username for user %d: ", i + 1);
    fgets(users[i].username, MAX_LINE_LENGTH, stdin);
    users[i].username[strcspn(users[i].username, "\n")] = 0;

    printf("Enter password for user %d: ", i + 1);
    fgets(users[i].password, MAX_LINE_LENGTH, stdin);
    users[i].password[strcspn(users[i].password, "\n")] = 0;
}


printf("\nCompromised Usernames and Passwords:\n");
for (int i = 0; i < userCount; i++) {
    printf("%s:%s\n", users[i].username, users[i].password);
}


identifyVulnerabilities(users, userCount);


printf("\nPassword Strength Evaluation:\n");
for (int i = 0; i < userCount; i++) {
    char strength[10];
    evaluatePasswordStrength(users[i].password, strength);
    printf("%s: %s\n", users[i].username, strength);
    }

    return EXIT_SUCCESS;
}
```