**콘솔 게임 설명서**

|  |  |  |  |
| --- | --- | --- | --- |
| **학번** | 1502, 1513 | **이름** | 김경민, 김휘림 |
| **게임 이름** | hack\_the\_dimi | | |
| **게임 개요[10점]** | | | |
| 1. 게임 소개   Hack The Dimi는 디미고를 배경으로 제작된 게임으로, 플레이어가 기말고사를 앞둔 학생이 되어 시험지를 얻기 위해 여러 해킹 미니게임을 해결해 나가는 게임입니다.   1. 제작 의도   진입 장벽이 높고, 어렵게만 느껴지는 해킹이라는 분야를 게임을 통해 자연스럽게 체험해 봄으로써 해킹의 진입장벽에 대한 막연한 부담감을 낮추고, 게임 엔딩을 통해 윤리의식의 필요성을 강조한다.   1. 게임 플레이 방법 2. 타이틀 화면 3. New   클릭시 새로운 게임을 시작한다.   1. Continue   클릭시 세이브 슬롯을 불러온다. 세이브 슬롯에는 마지막으로 저장한 시간, 스테이지 등의 정보가 기록되어있다.  키보드 위, 아래 화살표를 통해 불러올 세이브 슬롯을 선택할 수 있으며, 엔터 키를 통해 게임을 불러와 저장된 시점부터 시작할 수 있다. Esc 키를 누르면 저장하지 않고 다시 타이틀 화면으로 돌아온다.   1. Settings   중앙에 있는 버튼을 클릭하여 배경음악을 끄고 켤 수 있다.   1. Exit   클릭시 게임을 종료한다.   1. 튜토리얼 화면   New 버튼을 클릭하여 게임을 처음 시작하면 튜토리얼을 읽을 수 있다. 1,2번키 입력을 통해 튜토리얼 1,2를 볼 수 있습니다. S키 입력을 통해 튜토리얼을 종료하고 게임을 시작할 수 있다.   1. 이동 2. MainMap   텍스트, 스크린샷이(가) 표시된 사진  자동 생성된 설명  위의 그림이 MainMap이다. 건물들에 마우스를 올리면, 해당 건물의 일러스트가 중앙에 뜨며, 클릭시 해당 건물로 이동하거나 이동이 불가한 건물이면 대화창이 뜬다.   1. 학봉관, 본관   이동이 가능한 공간이면 마우스를 해당 공간 위에 올렸을 때, 해당 공간이 반투명 흰색으로 변하며, 클릭시 이동이 가능하다.   1. 상호작용, 저장    1. 상호작용   일정 조건에서 특정 장소를 클릭했거나, 엔딩을 보게 되면 대화창이 뜨게 된다. 대화창은 마우스를 클릭하면 종료하거나, 다음 대화로 넘어갈 수 있다.   * 1. 저장   게임 시작 후 MainMap에서 우측 상단의 노란색 Save 버튼을 클릭하면, 세이브 슬롯이 뜨며 타이틀 화면의 Continue와 마찬가지로 방향키, Enter, Esc를 통해 선택, 저장, 나가기가 가능하다. 저장에 성공하면 “Successfully Saved!”가 뜨고, 실패 시 “Save Failed” 화면이 뜬다. 저장은 오직 MainMap에서만 가능하다.   1. Buffer Overflow 스테이지    1. 입력   키보드를 통해 패스워드 입력이 가능하며, 엔터키를 누르면 비밀번호를 검사하여 입력값이 올바르면 진행되고, 틀리면 “Wrong”이라는 창이 뜨고 다시 입력해야한다.   1. Caesar Cipher 스테이지    1. 원판 회전   좌, 우 방향키를 통해 각각 원판을 왼쪽과 오른쪽으로 회전이 가능하며, 내부 원판과 외부 원판의 배열에 따라 질문 문자열의 배열이 바뀐다.   * 1. 입력   키보드를 통해 패스워드 입력이 가능하며, 엔터키를 누르면 비밀번호를 검사하여 입력값이 올바르면 진행되고, 틀리면 “Wrong”이라는 창이 뜨고 다시 입력해야한다.   1. Lock 스테이지    1. 자물쇠 다이얼 조작   숫자 위에 마우스를 올린 후 좌클릭 시 다이얼의 숫자가 1 만큼 증가하고, 우클릭 시 1 만큼 감소한다.   * 1. 자물쇠 해제   클릭하여 증감된 자릿수의 숫자와 비밀번호의 일치 여부에 따라 다른 소리가 난다. 만약 자물쇠의 비밀번호를 맞추었다면, 자물쇠가 해제되고, 해당 보관함의 시험지 보관 여부에 따라 시험지 획득 창 또는 시험지가 없다는 창이 뜬다.   1. 인게임 UI    1. 뱃지   문제를 해결하거나, 아이템을 얻으면 고양이 뱃지가 생긴다.   * 1. taskList   플레이어가 해야할 행동을 알려준다.   * 1. Item   플레이어가 지금까지 얻은 물품들이 표시된다. | | | |
| **게임의 기술적 포인트(5가지 이상 기술할 것)[각 2점]** | | | |
| 1. **실제 Buffer Overflow 취약점의 원리를 이용하여 Buffer Overflow(이하 BOF) 스테이지를 구현하였다.**   **BOF 공격은 연속된 메모리 공간을 사용하는 프로그램에서 할당된 메모리 범위를 넘어선 위치에 자료를 읽거나 쓸 때 발생한다. 이러한 BOF 공격의 특징을 살려 단순히 정해진 답이 있는 것이 아니라 실제로 사용자가 input값을 입력해서 비밀번호가 저장되어 있는 곳의 메모리를 덮어쓰면 그것을 인식하여 스테이지가 클리어 되도록 구현했다.** | | | |
| 1. **카이사르 암호의 특성을 고려하여 플레이어가 직접 원판을 조작하여 암호화 된 질문을 복호화 할 수 있도록 구현하였다.**   **카이사르 암호는 알파벳을 일정한 거리만큼 밀어서 다른 알파벳으로 치환하여 암호화하는 방식이다. 이러한 특징으로 인해 카이사르 암호는 두 개의 원판을 이용해 표현하면 쉽게 암호화와 복호화가 가능하다. 이와 같은 카이사르 암호의 특성을 반영하여 회전하는 원판이 기준이 되는 원판의 알파벳과 랜덤한 값만큼 밀려서 생성되도록 하였다. 플레이어는 해당 스테이지를 해결하며 직접 밀린 값만큼 원판을 회전시켜 복호화하는 과정에서 시저 암호에 대한 자연스러운 이해가 가능하다.** | | | |
| 1. **다이얼 자물쇠를 회전 소리를 통해 열 수 있는 기술을 구현하였다.**   **자전거 자물쇠와 같은 다이얼 자물쇠의 비밀번호를 까먹었을 때, 다이얼을 돌리는 소리에 집중하면 비밀번호를 알아 낼 수 있다고 한다. Lock 스테이지에서는 실제로 다이얼을 돌릴 때마다 소리가 나고, 각 자릿수가 비밀번호인지의 여부에 따라 다른 소리가 난다. 또한 실제로 자물쇠의 다이얼을 너무 빠르게 돌리면 정확한 소리의 차이를 파악할 수 없는 것처럼, 플레이어가 너무 빠르게 다이얼을 돌릴 시(동시에 너무 빠르게 마우스를 클릭할 시) 제대로 소리를 파악할 수 없도록 구현하여 현실성을 높혔다.** | | | |

|  |  |  |
| --- | --- | --- |
| 1. **이미지 사용에 있어 발생하는 어려움을 해결하였다.**   **이미지를 콘솔 화면에 원하는데로 띄우려면 띄워줄 비트맵에서의 좌표값을 구해주거나 파일 타입을 bmp형태로 변환해주어야 한다. 그러나 사용하는 이미지의 개수가 많아질수록 이런 작업을 직접 하는 것은 한계가 있었다. 그리하여 이미지 타입 변환을 위해 특정 폴더에 이미지를 넣어두면 해당 이미지들을 모두 bmp 타입으로 변환시켜주는 프로그램을 작성하여 자동화했다. 또한 마우스 입력 처리를 편하게 하기 위해 이미지의 x,y 좌표값을 자동으로 구해주는 프로그램을 작성하여 해당 좌표를 구조체에 입력해주면 그 값을 통해 마우스 입력을 처리하도록 하여 편의성과 생산성을 향상하였다.** | | |
| 1. **세이브 파일의 슬롯을 여러 개로 하고, 마지막으로 저장한 시각과 플레이 중이던 스테이지를 세이브 슬롯에 표시해주어 편의성을 향상했다.**   **세이브 기능을 구현할 때 플레이어가 보다 편하게 게임을 플레이 할 수 있도록 하기위해 세이브 슬롯을 3개 만들었다.**  **그러나 세이브 슬롯을 여러 개로 구현했다고 하더라도 플레이어 입장에서는 세이브 파일을 구분할 수 없다면 불편할 것이다. 이러한 불편함을 줄이기 위해 마지막으로 세이브한 시각과 플레이 중이던 스테이지를 슬롯에 표기해주어 플레이어가 어떤 세이브 파일을 로드해야 할지 더욱 명확하게 알 수 있도록 해주었다.** | | |
| **게임의 재미 요소 2가지[각 3점]** | | |
| 1. hacking을 굉장히 쉽게 접할 수 있다는게 재미 요소라고 생각한다.   보통, 해킹이라고 하면 굉장히 많은 지식이 필요하고 엄청난 양의 코드를 작성해서 exploit하는 것이라고 생각하는 사람들이 많다.  하지만, 본 게임의 buffer overflow stage를 예로 들어서 보면, 이 stage에서는 엄청난 전문 지식을 사용하지 않고, 간단한 memory overwrite 취약점을 게임 형식으로 구현함으로써 현실에서도 충분히 일어날 수 있는 취약점에 대해 설명해준다.  이를 통해 이 게임을 하는 사람들이 해킹이라는 것이 엄청 어렵고, 힘든 분야가 아니라는 것을 깨달을 수 있다고 생각한다. —> 이게 곧 사람들에게 재미 요소, 흥미 요소가 될 수 있다고 생각한다. (해킹을 시작하려고 했지만, 너무 어려울 것 같아서 못하고 있는 사람들에게도 좋은 흥미거리가 되지 않을까 생각한다.)  또한, bof 뿐만 아니라, caesar cipher, 자물쇠 뚫기 같은 stage를 추가하여 뭔가 비정상적인 결과를 도출해내고, 공격자가 자신의 의도대로 프로그램의 흐름을 변경할 수 있다는 해킹의 기본적인 개념들에 대해서도 드러내고 있어서, 조금 더 본질적으로 사람들이 해킹에 대해 접근할 수 있게 한다.  결론적으로 이 게임의 가장 핵심적인 재미 요소는 여러 해킹 기법들을 게임으로 구현함으로써 이를 플레이하는 사용자로 하여금 “뭔가 하고 있다!, 해킹을 하고 있다!”라는 멋진(?) 느낌이 들게 하는 것이다.   1. 해킹뿐만 아니라 스토리가 함께 곁들여져 있다는 점이 두번째 재미 요소라고 생각한다. 만약 현재 만들어진 게임이 스토리 없이 단순히 해킹 문제만 푸는 게임이었다면 단순히 문제를 푸는 해킹대회나 워게임과 다를 바가 없다. 그러나 게임과 디미고를 배경으로 하는 스토리가 곁들여지면서 마침내 재미와 교육적 요소가 결합하여 이 게임의 중요한 재미 요소가 된 것 같다. | | |
| **게임의 핵심 화면 및 화면 설명[각 5점]** | | |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 | | 1. 화면 설명   왼쪽 사진은 세이브나 로드 기능을 사용시 나타나는 세이브 슬롯이다. 테두리가 초록색인 슬롯은 현재 선택된 슬롯을 나타내며, Stage에는 저장된 세이브 파일을 로드하면 어느 스테이지부터 시작하게 되는지 알려준다. 또한 Last Saved는 해당 세이브 파일의 정보가 언제 마지막으로 업데이트되었는지 알려준다. 화면 4개의 귀퉁이에 있는 “Esc Back”은 Esc를 누르면 이전 화면으로 돌아갈 수 있음을 알려준다.   1. 구현 방법    1. 세이브 슬롯 그리기   Save 또는 Load 기능을 수행해야 하는 경우 세이브 파일의 존재 여부를 확인하여 파일이 존재하면 파일을 열어 먼저 세이브 슬롯에 표시할 2가지 정보를 변수에 저장해둔다.  그 후 세이브 슬롯 이미지와 슬롯에 표시해줄 정보를 띄워준다.   * 1. 세이브 슬롯 선택하기   무한반복문에서 \_kbhit함수를 통해 키보드 입력의 여부를 확인한다. 그리고 현재 선택된 세이브 슬롯 번호를 저장할 변수를 선언한다. 세이브 슬롯 변수는 0은 1번 슬롯 1은 2번 슬롯 2는 3번 슬롯을 의미한다.  위쪽 화살표를 입력한 경우에 현재 세이브 슬롯 번호 변수가 0이 아닌 경우 변수의 값을 1 감소시키고, 아래쪽 화살표를 입력한 경우 현재 세이브 슬롯 번호 변수가 2가 아닌 경우 1증가시킨다.  그 후 이 변수의 값에 따라 이미지를 띄워준다.   * 1. 저장하기, 불러오기   저장하는 경우 : 저장해주어야 하는 값들을 sprintf함수를 통해 이차원 문자열 배열에 일정한 형식대로 저장한다.  그 후 배열에 저장된 문자열을 세이브 파일에 써준다.  불러오는 경우 : 세이브 파일에 저장된 데이터들을 저장할 변수들을 선언한다. 그 후 세이브 파일에서 문자열을 파싱해준 후 데이터를 읽어와 세이브 파일 데이터를 복사해두는 변수에 저장해준다.  변수에 저장된 데이터들을 다시 실제 게임 FLAG역할을 하는 변수들에 복사해준다. |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 | | . 1. 기존의 console 입력 처리 방식과 다르게 모든 것을 kbhit, getch로 받음으로써 정해진 자리에 정해진 글자를 쓸 수 있게 하였다. → 이로써 whitelist filtering 방식을 사용하여 사용자의 입력값을 필터링 할 수 있게 되었고, 폰트문제나 이런 문제가 발생하지 않도록, 모든 문자를 미리 이미지로 변환해 놔서 조금 더 빠르고, 안전한 접근이 가능하게 만들었다.  2. buffer overflow는 말 그대로 사용자가 자신이 접근 가능한 영역 이외에 영역에 잘못된 입력 범위 지정으로 값을 쓸 수 있게 될 때 생기는 취약점이다.  따라서, 우리는 본 기법의 정의를 그대로 살리기 위해 input, pw 영역 두개로 입력받는 곳을 나눠서 input값은 사용자 입력값이, pw 값에는 미지의 값이 들어가게 세팅했다.  이후, 사용자가 값을 씀에 따라 input 영역에 있는 값과, pw 영역에 있는 값이 맞는지 비교하게 되고, bof를 통해 pw 부분에 값이 덮어써지면서 input, pw 영역에 쓰여저 있는 값이 똑같게 되면, 통과가 되는 식으로 위 취약점을 구현하였다.  조금 더 현실적인 예를 들자면, master canary 라는 stack smashing detector에 관련된 함수 스택 카나리 검증 우회 취약점에서도 이와 같은 것을 이용한다.  물론 직접적인 bof가 일어나는 것은 아니지만, master canary에 쓰여진 값을 우리가 변조할 수 있다면, 다른 모든 canary 값을 우리가 쓴 값으로 변경했을 때, 모든 루틴이 통과되게 된다.  (이런식으로 최대한 현실에 있을 법한 취약점을 쉽게 쉽게 가공해서 위 stage를 제작하였다.)  3. 옆에 있는 것들에 대해 설명하자면, 맨 위에 다섯 개 polygon은 badge가 있을 자리를 표시한 것이고, badge의 개수를 전역 변수에서 읽어와 출력하는 함수를 매 실행마다 호출하여 현재 클리어한 스테이지의 개수를 알 수 있게 한다.  4. 밑에 tasklist에는 해야될 것들이 나열해 있게 되는데, 그냥 tasklist, cleared\_tasklist로 char배열을 만들어서 flag로 저장하는 방식을 통해 tasklist또한 badge와 같이 매 실행마다 tasklist를 출력하는 함수로 출력하게 된다.  5. 밑에 item에는 현재 소유하고 있는 아이템에 대해 출력하게 되는데, 본관 열쇠, 교무실 열쇠, 시험지 를 얻게 되면 얻은 아이템이 저 부분에 뜨게 된다. |
| **프로젝트 진행하면서 느낀점-팀원1[2점]** | 1. 나는 예전부터 많은 프로그래밍 언어 중 유독 C언어를 좋아했다. 그래서 디미고에 와서 C언어를 배운다는 소식을 듣고 굉장히 기뻤다. 그러나 내가 늘 느꼈던 아쉬운 점은 예전에 C언어를 공부해 보았지만 정작 실제 프로그램다운 프로그램은 만들어 본 적이 없어서 문법만 알고 활용해 볼 기회가 없었던 것이다. 그런데 이번 프로그래밍 수행평가를 하면서 게임을 만들어 볼 수 있어서 굉장히 기뻤던 것 같다. 또한 이번에 친구들과 함께 게임을 제작해보면서 다른 친구의 코드를 읽어보고, 함께 코드를 작성하고, 협업하는 과정도 처음 경험해 보아서 더욱 좋았던 것 같다. 다음에도 이런 기회가 있으면 좋겠다. | |
| **프로젝트 진행하면서 느낀점-팀원2[2점]** | 1. 윈도우 콘솔 게임을 만들어 본적도 처음이였고, 이미지를 콘솔에 출력한 것도 처음이였다. 구현하면서 “이게 된다고?”하는 마음으로 신기해 하면서 개발했던 것 같다.  오랜만에 개발 + 디자인도 같이 했던 프로젝트였던 것 같고, 주제를 해킹으로 잡은 만큼 제대로 구현해보고 싶었는데, bof나 caesar cipher의 경우에 굉장히 잘 구현된 것 같아서 기분이 좋다.  2. 개발을 하면서 중간중간 오류도 있었는데, 특히 기억에 나는 것은 이미지 출력이였다.  선생님이 주신 코드로 실행해 봤는데, 실행이 안 됐어서 새로운 이미지 처리 함수를 만들었던게 흥미로웠다. 또한, 내 마음대로 구현한 코드에 이미지 처리에 도움이 되는 구조체까지 만들고, 이렇게 했던 과정에서 “뭔가 만든다!”라는 느낌을 팍팍받았던 것 같다.  인터넷에서 긁어오고, 내 마음대로 이리저리 손 본 코드여서 제대로 실행이 안될 줄 알았는데, 잘 돼서 편하게 개발했던 것 같다.  이 외에도 필요에 의해 “png to bmp”, “console size converter” 등등의 기능을 수행하는 파이썬 파일들을 만들어서 개발을 했던 것이 기억에 남는다. 필요에 의해 디자인도 하고, 추가 툴도 제작하고, 모쪼록 재밌었던 개발이였다. | |