



제출일	2023.06.13	학과	컴퓨터공학전공
과목	컴퓨터보안	학번	2018112007
담당교수	김영부 교수님	이름	이승현



1) 실습 환경

(1)

운영 체제: Microsoft Windows 11 Home 64bit

프로세서 : Intel(R) Core(TM) i7-10510U @ 1.80GHz (8 CPUs), ~ 2.3GHz

메모리 : DDR4 16GB 2,667MHz

그래픽 카드 : Intel UHD Graphics

(2)

운영 체제: Microsoft Windows 10 Home 64bit

프로세서 : Intel(R) Core(TM) i7-7700HQ @ 2.80GHz (8 CPUs), ~ 2.8GHz

메모리 : DDR4 8GB 2,133MHz

그래픽 카드 : Intel HD Graphics 630, NVIDIA GeForce GTX 1050

2) 실습 진행

1. 문제 분석

i. 대칭키 암호/복호 AxCrypt 설치 및 사용

- 환경에서 널리 사용되는 공개용 파일 암호화 소프트웨어로 파일들의 암호화 복호화 그리고 저장 및 전송을 할 수 있도록 통합서비스를 제공한다. 또한 200만 명이 넘는 사용자를 자랑하니 프로그램의 안정성은 어느 정도 검증되었다고 볼 수 있다.
- AxCrypt는 하나의 독립된 프로그램에 의해 작동하지 않고 탐색기와 통합된 형태로 실행된다. 기본적으로 탐색기에서 파일이나 폴더를 우클릭함으로써 AxCrypt의 사용 메뉴를 불러올 수 있다. 특히 암호화하기, 암호화된 파일 열기, 복호화하기 등등의 메뉴 사용법을 익힌다.
- 일반적으로 [Encrypt]를 위한 메뉴에서 암호를 위해 passphrase를 사용하는 것 이외에 [Make Key-File]을 실행시켜 랜덤하게 생성된 'Key-File' 텍스트 문서를 이용하여 암호화하면 대상 파일을 AES -128 정도의 매우 강력한 암호로 보호할 수 있다.
- 다른 암호화 메뉴로 [Encrypt] 이외에 원본 파일을 유지한 채 추가로 암호화된 파일을 생성하는 기능인 [Encrypt a copy]와 AxCrypt가 설치되지 않은 컴퓨터의 사용자도 복호화가 가능하도록 Self-Decrypter 파일을 생성하는 [Encrypt copy to .EXE]가 있다.

ii. 공개키 암호 복호 GPG4Win 설치 및 사용

- GnuPG (GNU Privacy Guard)는 GNU재단에서 개발한 이메일 및 파일 내용의 암호화와 복호화 기능을 제공하는 대표적인 프로그램이다. GPG4Win은 GnuPG의 윈도우 버전이라 할 수 있다. 본 과제에서는 GPG4Win을 활용하여 사용자가 작성한 평문을 암호화한 후 암호화된 글을 다시 평문으로 복호화하는 과정에 대해 공부한다.
- GPG4Win 프로그램에서 1) GPA를 이용하여 비밀키와 공개키 생성하기, 2) 공개키 추출 및 공유하기, 3) 상대의 공개키 등록 및 암호문 작성, 4) 암호문 전달 및 복호화 과정, 5) 전자서명, 6) 암호화 및 전자서명을 동시에 하기, 7) 공개키를 교환하는 방식 등등을 자세히 익힌다.

iii. CrypTool

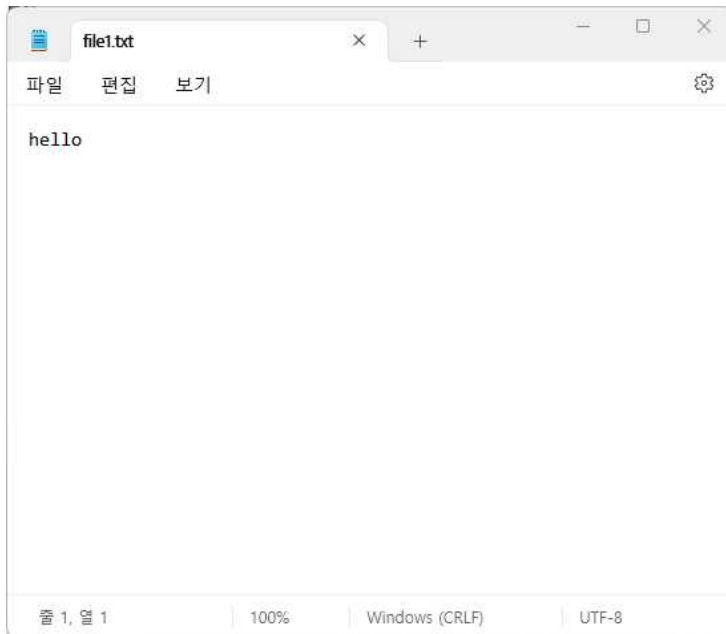
- CrypTool을 <https://www.cryptool.org/en/ct1/>에서 다운로드 하여 키 길이와 크래킹 횟수의 연관성을 확인한다.

iv. 이메일 사용자 이름과 패스워드 추출

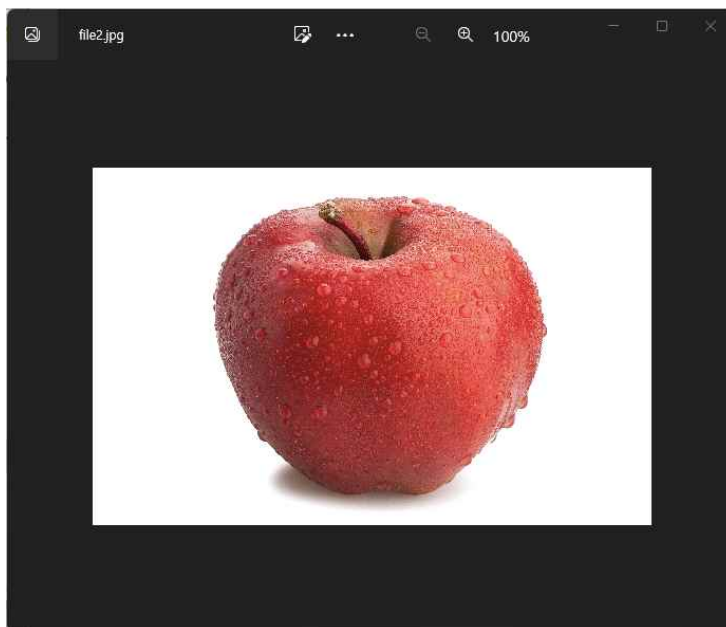
- Wireshark에서 SMTP.pcap 파일을 통해 SMTP 캡처에서 사용한 이름과 패스워드를 추출하고 DECODE를 수행한다.

2. 실습

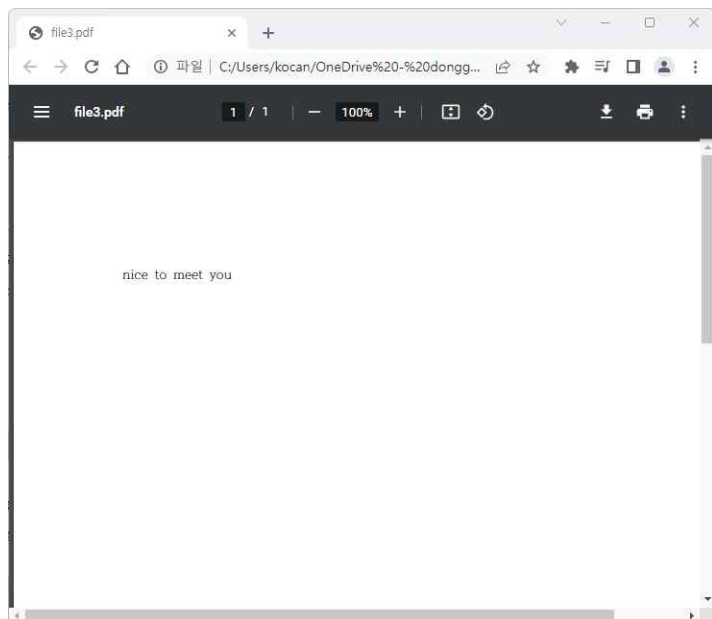
i. 대칭키 암호/복호 AxCrypt 설치 및 사용



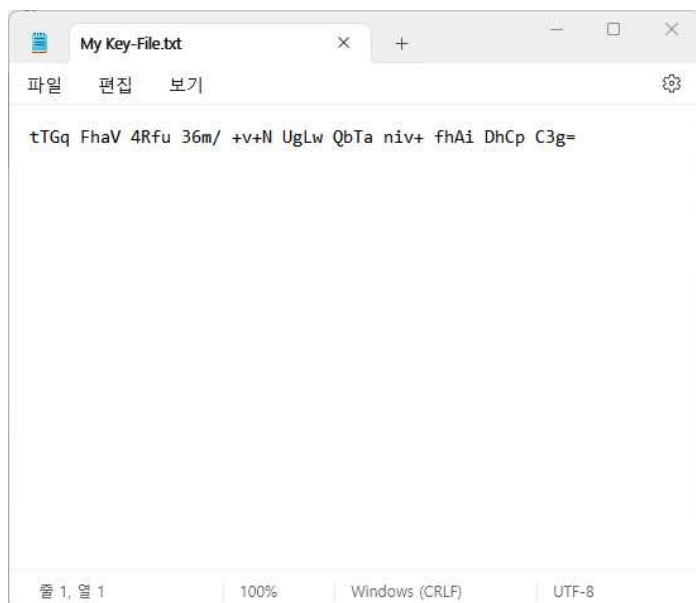
- 암호화할 첫 번째 파일이다. (file1.txt)
- 텍스트 파일이며, 'hello'가 기재되어있다.



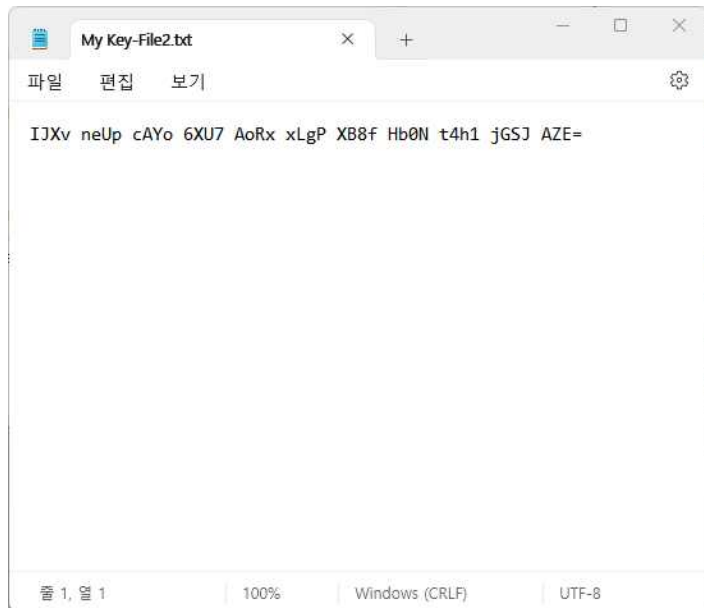
- 암호화할 두 번째 파일이다. (file2.jpg)
- 이미지 파일이며, 사과 이미지가 담겨있다.



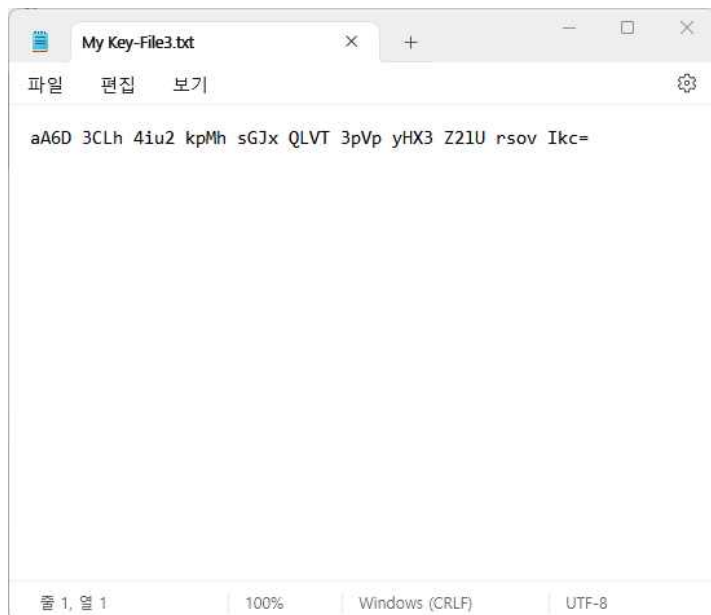
- 암호화할 세 번째 파일이다. (file3.pdf)
- pdf 파일이며, 'nice to meet you'라고 기재되어있다.



- file1.txt로 key-file을 생성했다. (My Key-File.txt)
- key 내용은 위 이미지와 같다.

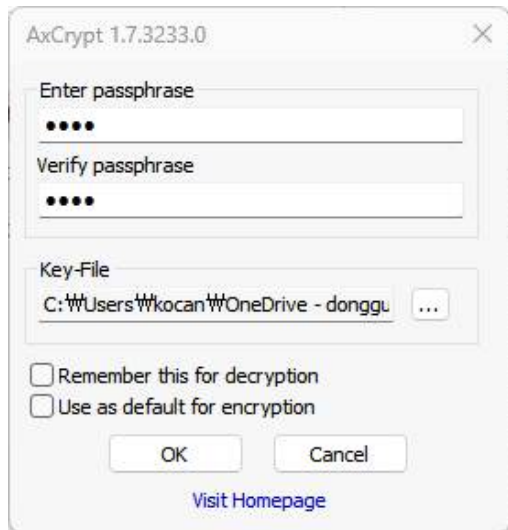


- file2.jpg로 key-file을 생성했다. (My Key-File2.txt)
- key 내용은 위 이미지와 같다.



- file3.pdf로 key-file을 생성했다. (My Key-File3.txt)
- key 내용은 위 이미지와 같다.
- 3개의 key 모두 다른 모습을 확인할 수 있다.

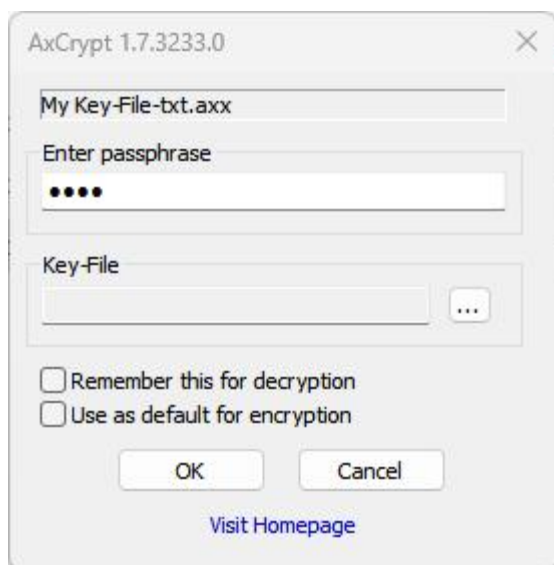
1) 하나의 Key-File을 자기 자신을 Key-File로 하여 [Encrypt]했을 경우 어떤 일이 발생하는지 설명하라.



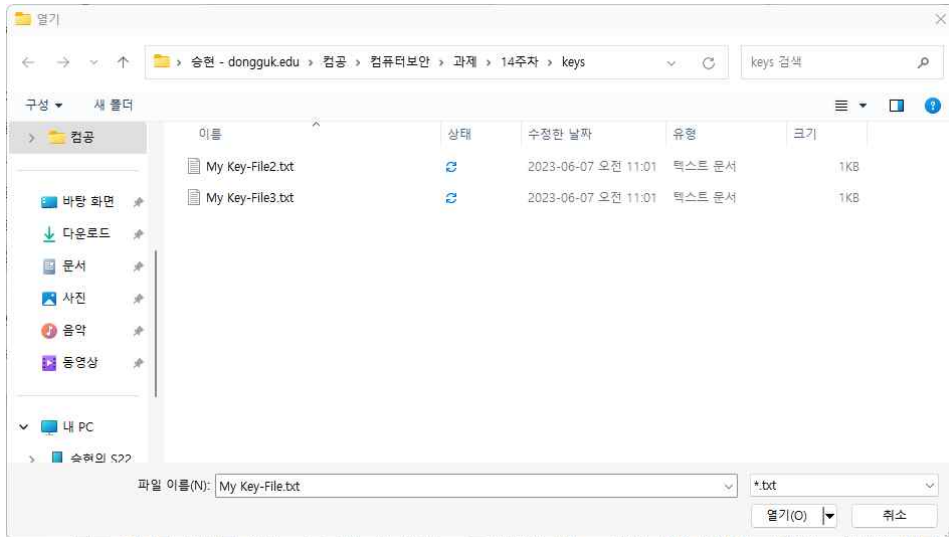
- My Key-File.txt를 암호화하려고 한다.
- 파일에서 오른쪽 클릭 -> AxCrypt -> Encrypt 메뉴를 클릭하여 위 이미지와 같은 창을 띄운다.
- passphrase를 입력한다. 이 값은 나중에 decrypt 할 때 필요하다.
- key-file을 선택해야 한다. key-file의 경우 My Key-File.txt 자기 자신을 선택한다.
- key-file까지 선택하면 OK 버튼을 클릭하여 암호화한다.

My Key-File-txt.axx

- My Key-File.txt 자기 자신이 암호화된 모습을 확인할 수 있다.

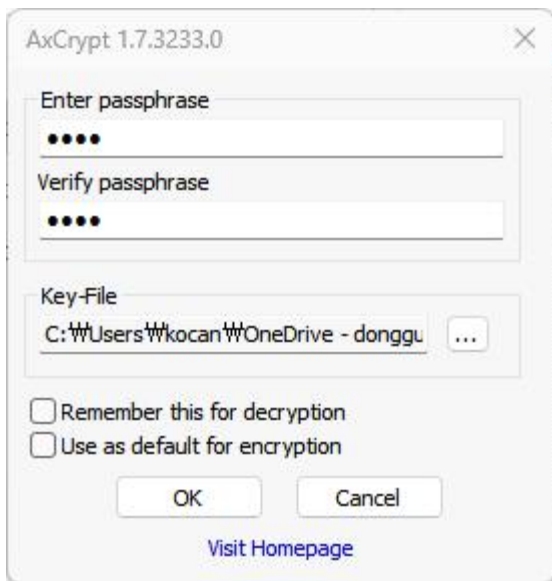


- 암호화된 파일을 열게 되면 위 이미지와 같은 창이 나타난다.
- 복호화를 위해서 암호화할 때 입력한 passphrase와 key-file을 입력해야 한다.
- key-file의 경우 자기 자신을 암호화했기에 선택할 수 있는 파일이 없어 복호화할 수 없다.



- key-file을 선택하려고 하면 My Key-File.txt가 존재하지 않아 복호화를 진행할 수 없다는 문제가 발생한다.

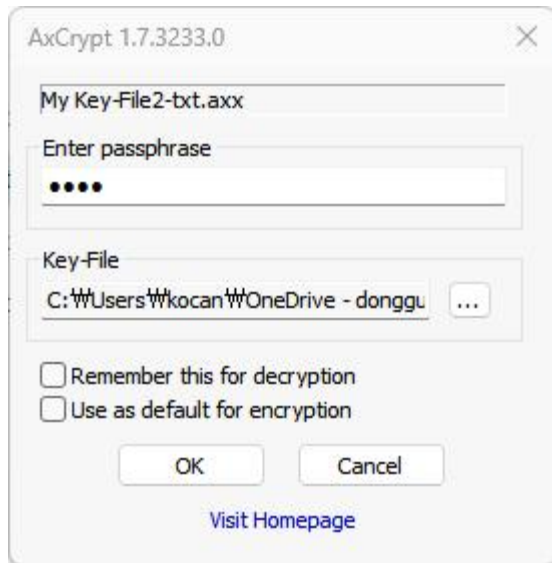
2) 하나의 Key-File을 자기 자신을 Key-File로 하여 [Encrypt a copy]를 실행시켰을 경우는 1번과 어떻게 달라지는지 설명하라.



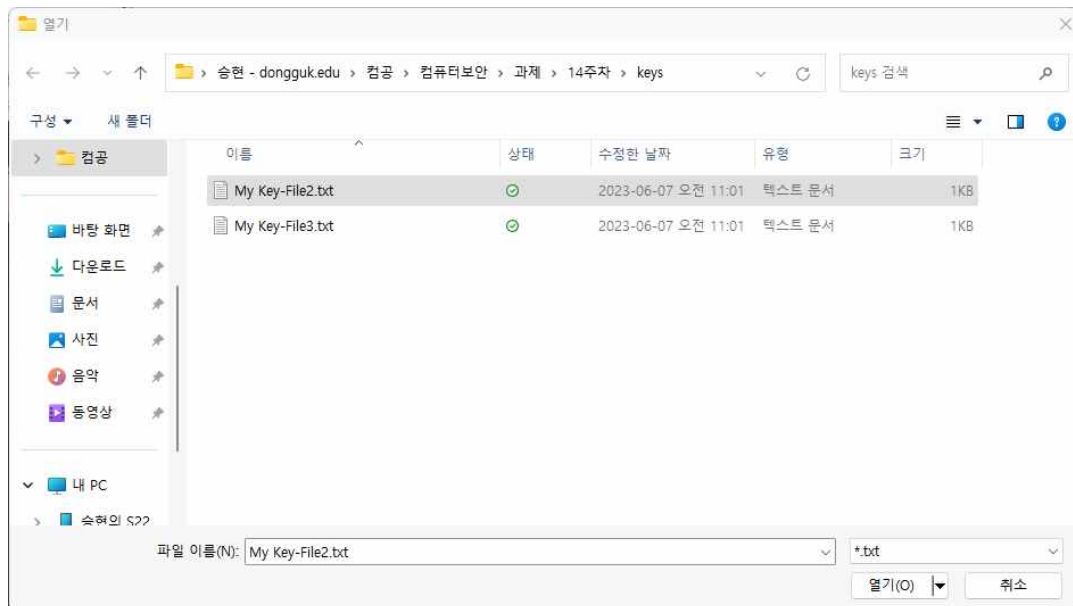
- My Key-File2.txt를 암호화하려고 한다.
- 파일에서 오른쪽 클릭 -> AxCrypt -> Encrypt a copy 메뉴를 클릭하여 위 이미지와 같은 창을 띄운다.
- passphrase를 입력한다. 이 값은 나중에 decrypt 할 때 필요하다.
- key-file을 선택해야 한다. key-file의 경우 My Key-File2.txt 자기 자신을 선택한다.
- key-file까지 선택하면 OK 버튼을 클릭하여 암호화한다.



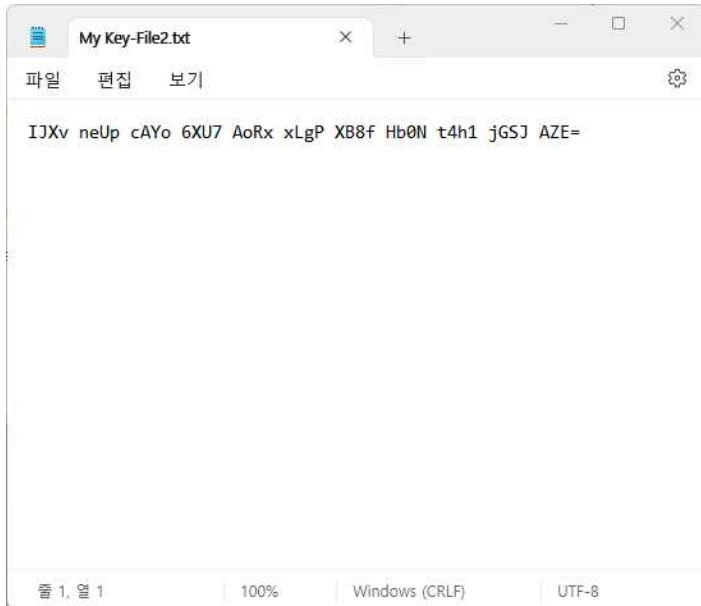
- 1번에서는 My Key-File.txt 자기 자신이 암호화가 되었지만, 이 문제에서는 My Key-File2.txt 원본이 암호화되지 않고 파일을 복사한 다음에 암호화되기에 원본 파일이 그대로 존재한다.



- 암호화된 파일을 열게 되면 위 이미지와 같은 창이 나타난다.
- 복호화를 위해서 암호화할 때 입력한 passphrase와 key-file을 입력해야 한다.
- passphrase와 key-file을 입력하면 복호화가 진행되어 원본 파일이 출력된다.



- 1번과 다르게 My Key-File2.txt 원본이 남아있기에 복호화할 때 선택할 수 있었다.

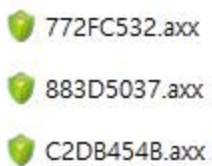


- 복호화되어 My Key-File2.txt가 실행한 모습이다.
- 암호화되기 전 My Key-File2.txt와 내용이 일치함을 볼 수 있다.

3) 각각의 파일을 [Encrypt a copy]를 이용하여 암호화된 사본 파일을 만들고 이들을 하나의 폴더(폴더명은 Test라 하자)에 따로 저장한 뒤 그 폴더 전체에 [Rename]을 실행시켜라. 폴더를 열어 암호화된 파일들이 어떻게 변하였는지 설명하라.



- 암호화된 파일을 Test 폴더에 저장한 모습이다.
- Test 폴더에서 오른쪽 클릭 -> AxCrypt -> Rename 메뉴를 클릭하여 암호화된 파일명을 변경한다.

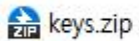


- Rename을 수행한 모습이다.
- Test 폴더 내 암호화된 파일의 이름이 무작위로 설정된 모습을 확인할 수 있다.
- 어떤 파일이 암호화된 것인지 구분이 어렵기에 복호화를 진행할 수 없었다.

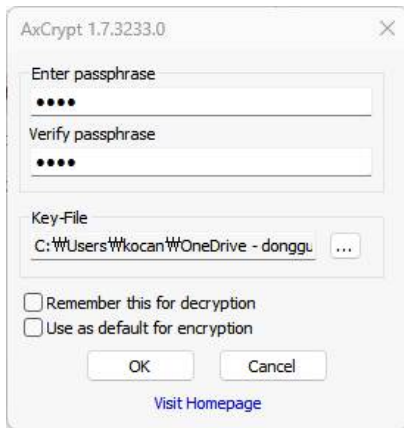
4) 3번 문제에서 각각의 파일마다 암호화 passphrase나 Key-File이 다를 경우 파일들을 어떻게 구분해 낼 수 있는지 설명하라.

- Rename 하기 전 원본 파일의 수정한 날짜와 Rename 한 후 파일의 수정한 날짜를 비교하여 어떤 파일이 Rename 되어 바뀌었는지 확인하는 방법이 있다.
- 각각의 파일마다 알고 있는 passphrase와 가지고 있는 key-file을 이용해 복호화를 시도하여 어떤 파일이 암호화되었는지 구분한다.

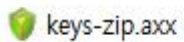
- 5) keys 폴더를 zip 파일로 압축한 뒤 [Encrypt]를 실행시켜라. 어떤 일이 발생하는가? 5번 문제와 비교하여 설명하라.



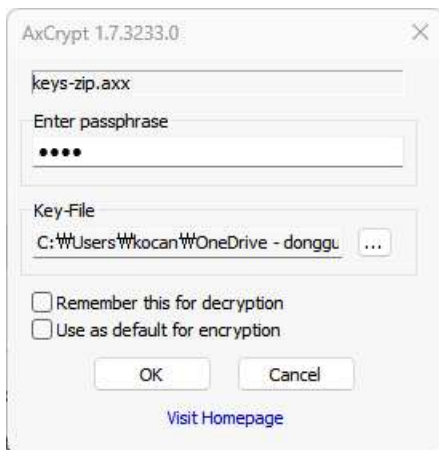
- keys 폴더를 zip 파일로 압축한 모습이다.



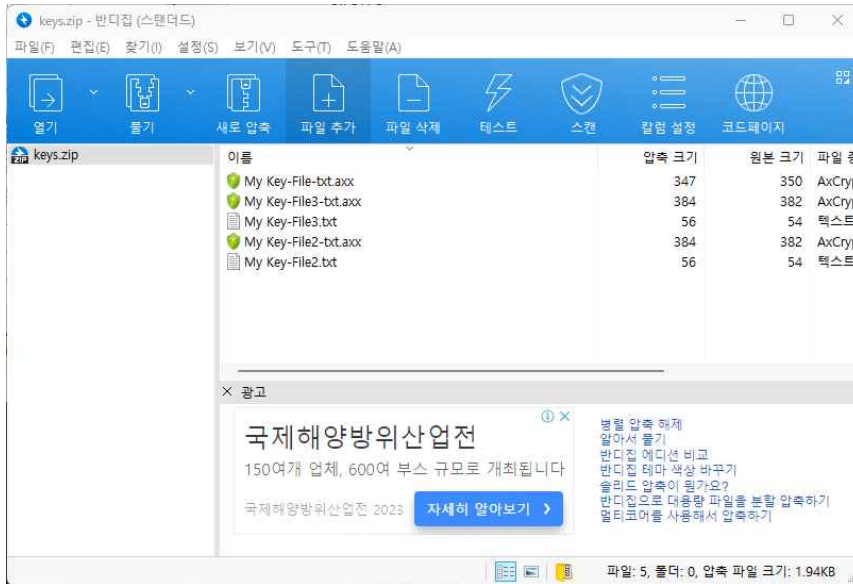
- 압축한 zip 파일에서 오른쪽 클릭 -> AxCrypt -> Encrypt 메뉴를 클릭하여 위 이미지와 같은 창을 띄운다.
- passphrase와 key-file을 입력한다. key-file은 My Key-File2.txt가 사용된다.



- 압축한 파일이 암호화된 모습을 확인할 수 있다.
- 원본 zip 파일은 사라지고, 암호화된 파일만 남게 된다.



- 암호화된 파일을 열게 되면 위 이미지와 같은 창이 나타난다.
- 복호화를 위해서 암호화할 때 입력한 passphrase와 key-file을 입력해야 한다.
- passphrase와 key-file을 입력하면 복호화가 진행되어 원본 파일이 실행된다.



- 복호화가 진행되면 위 이미지와 같이 zip 파일이 실행되는 모습을 확인할 수 있다.
- 압축된 파일에는 key-file 들과 암호화된 key-file 들이 존재한다.
- 3번과 같이 key-file 전체를 암호화하고 rename 하는 경우 key-file의 개수만큼 key-file이 필요하고, rename을 진행하면 파일을 구분할 수 없게 된다. 그러나 이 문제와 같이 key-file 전체를 압축한 다음 암호화를 진행하면 key-file 하나만 필요할 뿐만 아니라 파일 구분에도 문제가 없다.

6) 우리는 Key-File을 생성해서 파일을 암호화할 경우 AES - 128의 강도로 암호화할 수 있다는 것을 알았다. 하지만 파일들을 암호화할 때 항상 같은 Key-File을 사용하는 것이 아니라면 위의 문제들과 같이 여러 개의 Key-File이 생성하게 되고, 이것들을 관리하는 것 또한 보안 유지를 위해 중요한 일이 된다. 그러면 4번과 5번 문제를 종합하여 Key-File의 효율적인 관리방안을 모색해 보아라.

- 여러 key-file을 각각 암호화해서 사용하는 경우 key-file의 개수만큼 key-file을 만들어야 한다. key-file이 몇 개밖에 없으면 상관없지만, 만약 몇백 개, 몇천 개 이상의 key-file을 암호화하려고 한다면 관리가 힘들어질 것이다. 따라서 key-file을 안전하게 지키고 싶다면 key-file을 각각 암호화하는 것보다 key-file 전체를 하나의 파일로 압축한 다음에 압축한 파일을 암호화한다면 필요한 key-file의 개수는 줄어들고, 복호화 한 번만으로 모든 key-file을 사용할 수 있다.

ii. 공개키 암호 복호 GPG4Win 설치 및 사용

1) GPA(Kleopatra)를 이용하여 Alice와 Bob의 비밀키와 공개키를 생성하고 과정과 결과를 설명하라.



- Kleopatra를 실행한 모습이다.
- 아무런 인증서가 존재하지 않은 것을 확인할 수 있다.



- 메뉴에서 파일 -> 새 OpenPGP 키 쌍...을 클릭하여 새로운 인증서를 생성할 수 있다.



- 이름과 이메일 주소를 입력할 수 있다.
- 이름에는 Alice, 이메일 주소에는 Alice@naver.com을 입력하였다.
- 이름과 이메일 주소를 입력한 후 확인 버튼을 클릭하면 인증서 생성이 완료된다.



- 마찬가지로 메뉴에서 파일 -> 새 OpenPGP 키 쌍...을 클릭하여 새로운 인증서를 추가로 생성한다.
- 이름에는 Bob, 이메일 주소에는 Bob@naver.com을 입력하였다.
- 이름과 이메일 주소를 입력한 후 확인 버튼을 클릭하면 인증서 생성이 완료된다.

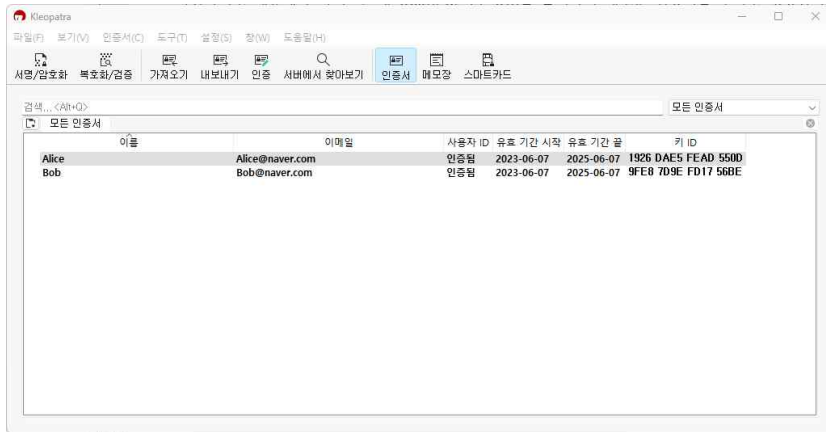
검색... <Alt+Q>

모든 인증서

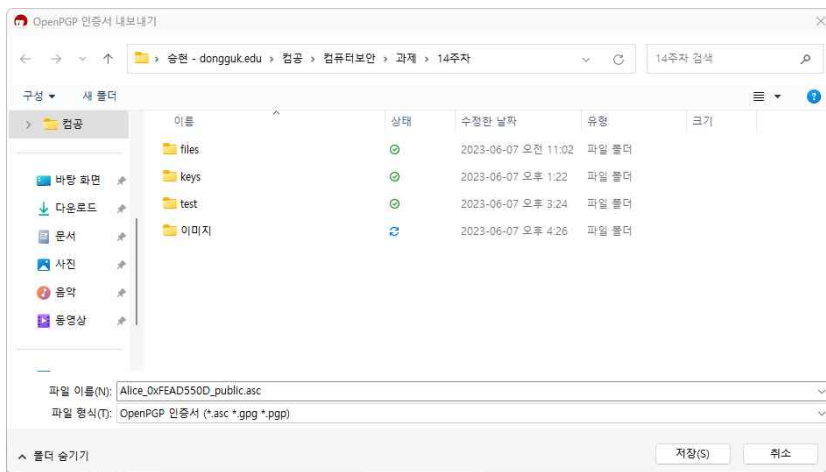
이름	이메일	사용자 ID	유효 기간 시작	유효 기간 끝	키 ID
Alice	Alice@naver.com	인증됨	2023-06-07	2025-06-07	1926 DAE5 FEAD 550D
Bob	Bob@naver.com	인증됨	2023-06-07	2025-06-07	9FE8 7D9E FD17 56BE

- 메인 화면에서 Alice와 Bob의 인증서가 생성된 모습을 확인할 수 있다.

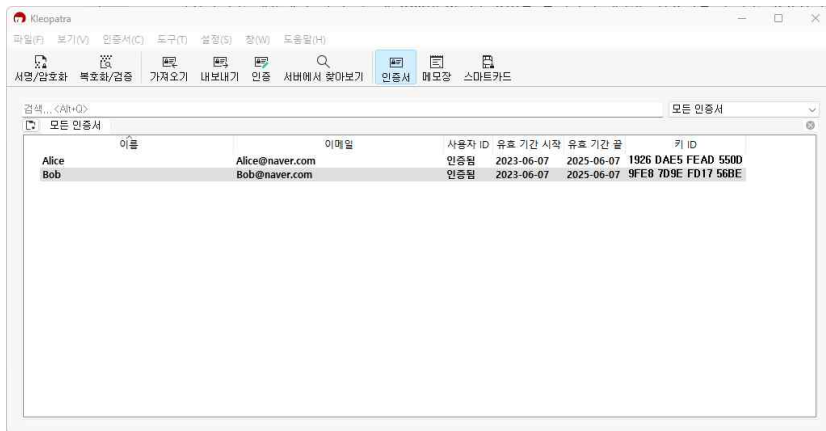
2) 2명 모두 공개키 추출을 진행하고 공개키를 서로 공유하는 시나리오를 생각해보고 작성하라.



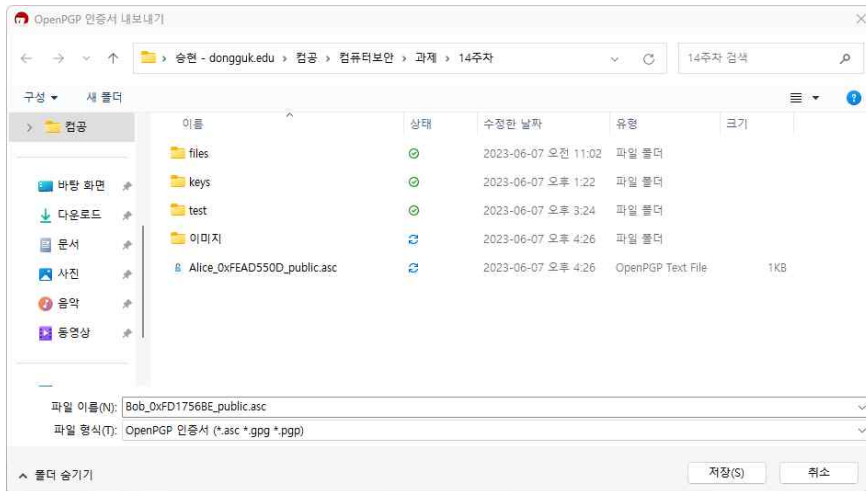
- 메인 화면에서 Alice를 선택하고, 내보내기를 클릭하여 공개키 추출을 진행한다.



- Alice의 공개키를 저장할 경로를 지정한 후 저장 버튼을 클릭한다.



- 마찬가지로 메인 화면에서 Bob을 선택하고, 내보내기를 클릭하여 공개키 추출을 진행한다.



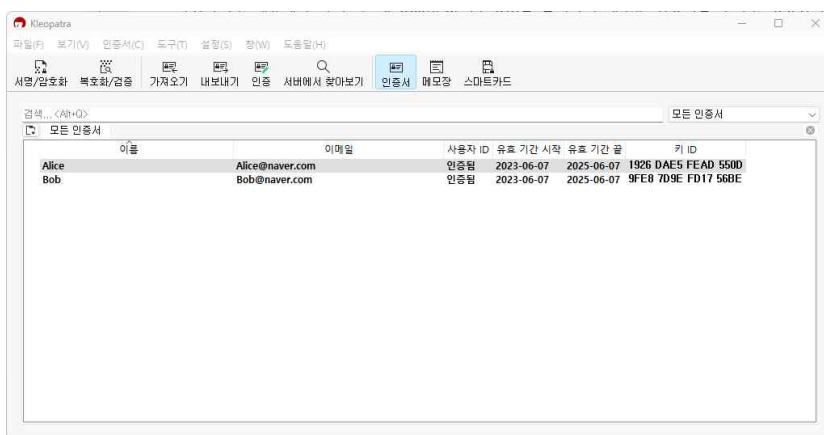
- Bob의 공개키를 저장할 경로를 지정한 후 저장 버튼을 클릭한다.

Alice_OxFEAD550D_public.asc

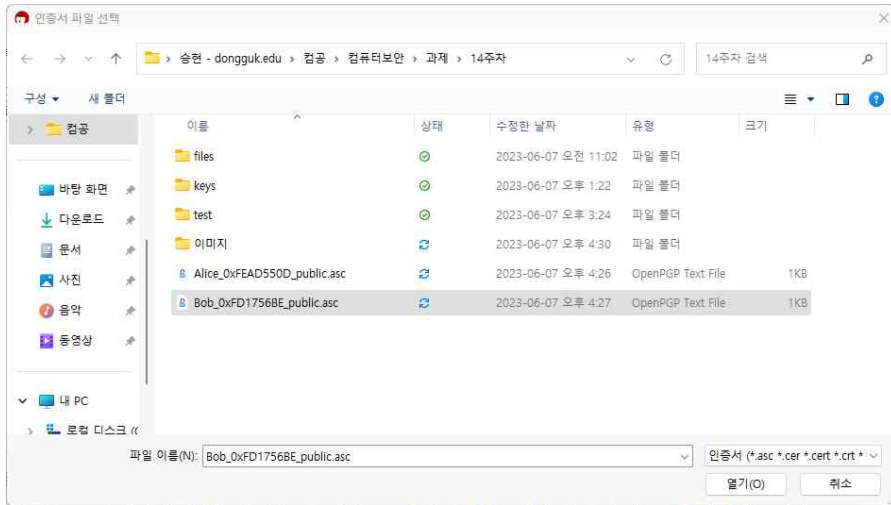
Bob_OxFD1756BE_public.asc

- Alice와 Bob의 공개키가 저장된 모습을 확인할 수 있었다.
- 공개키 공유 시나리오의 경우 클라우드 서비스를 이용한 시나리오로 구상하였다.
- Alice와 Bob은 보안 문제로 인하여 파일을 암호화하여 주고받기로 했다. 이때 암호화된 파일을 복호화하기 위해서 공유키를 공유해야 하는 문제가 발생하는데, 이때 Alice는 자신들만 사용하는 클라우드 서비스에 공개키를 업로드하여 공유하는 것이 어떨까 하는 의견을 냈고, Bob은 이에 찬성하여 클라우드 서비스에 자신들의 공개키를 올리기로 했다.

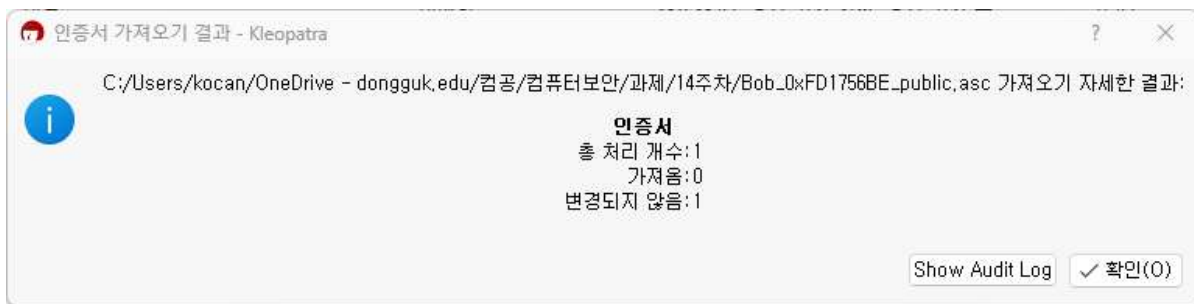
3) 본인이 상상한 시나리오를 기반으로 각각 상대의 공개키 등록 및 암호문을 작성해 보아라.



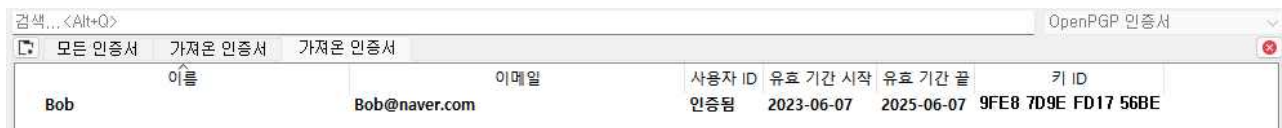
- 메인 화면에서 가져오기 버튼을 클릭하여 상대의 공개키를 등록한다.
- Alice에서는 Bob의 공개키를 가져와야 한다.



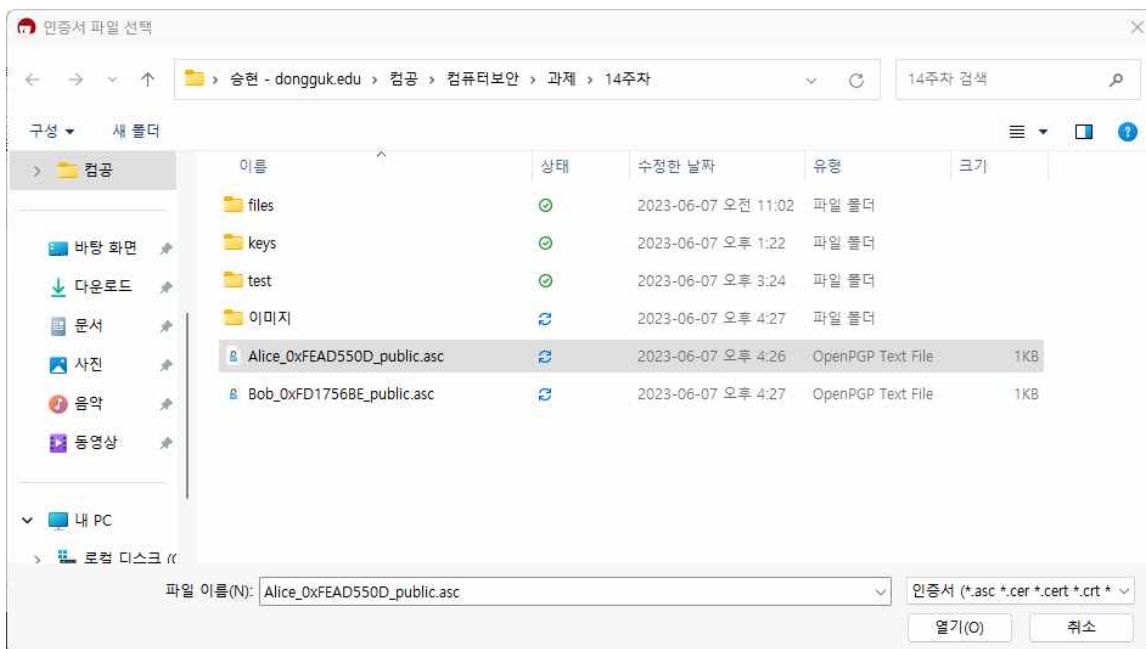
- Alice는 클라우드 서비스에 저장된 Bob의 인증서를 선택한 후 열기를 클릭한다.



- Bob의 인증서를 가져왔다는 메시지가 출력된다.

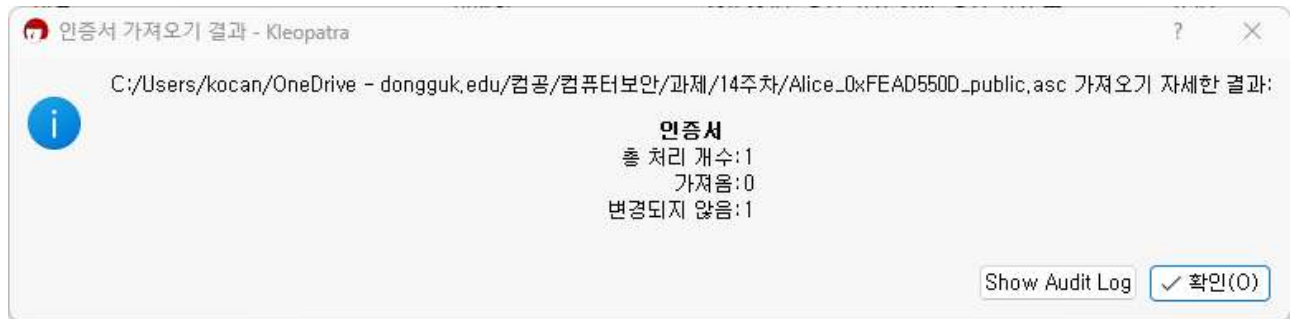


- 가져온 Bob의 공개키의 정보가 출력된다.

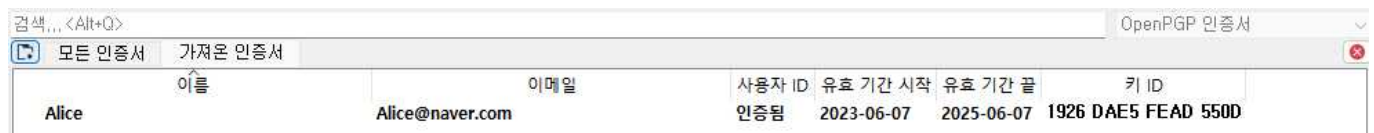


- Bob은 Alice의 공개키를 가져와야 하기에 Alice의 인증서를 선택한다.

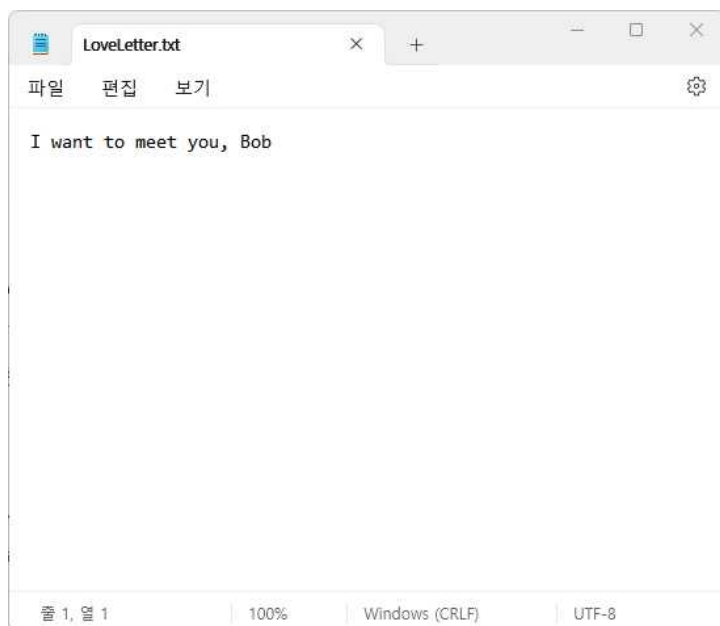
- Bob은 클라우드 서비스에 저장된 Alice의 공개키를 가져온다.



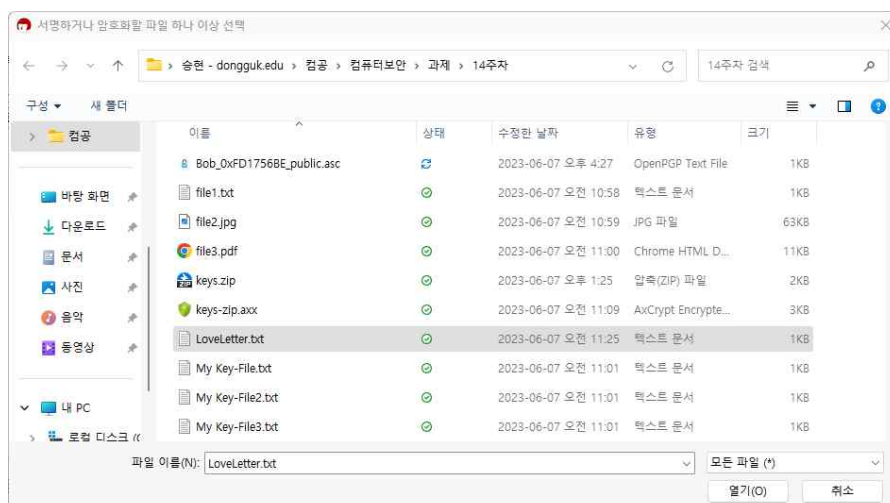
- Alice의 인증서를 가져왔다는 메시지가 출력된다.



- 가져온 Alice의 공개키의 정보가 출력된다.



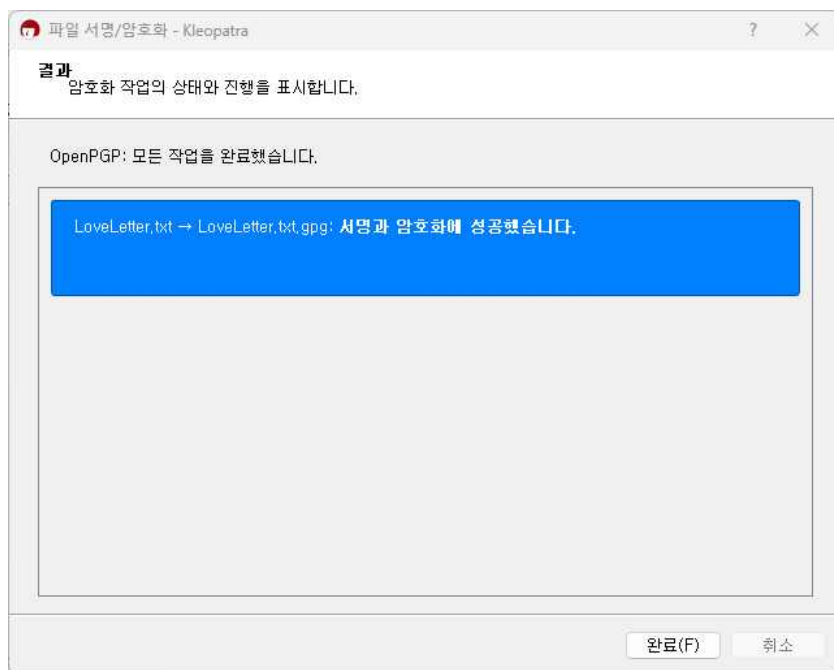
- 암호화할 평문의 내용이다.
- Alice가 Bob에게 암호화해서 전송할 것이다.




- 메인 화면에서 서명/암호화를 클릭하여 평문의 암호화를 진행한다.
- 암호화를 진행할 평문을 선택한다.



- Alice의 인증서를 이용하여 파일 서명을 진행한다.
- Bob에게 암호화를 진행한다.



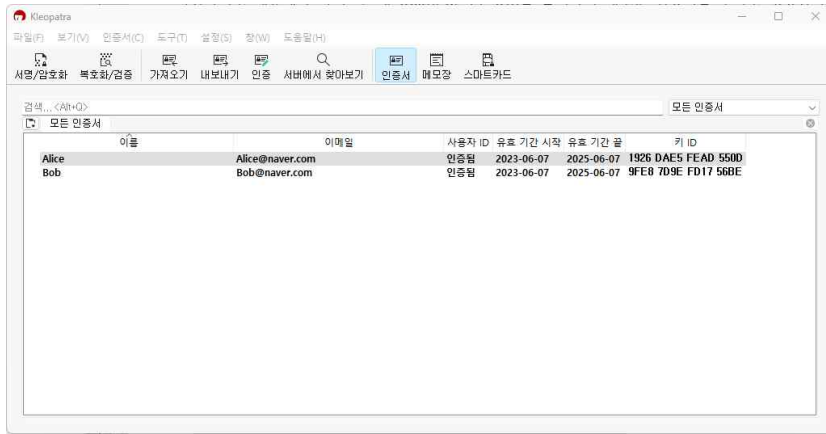
- 서명과 암호화에 성공하였다는 메시지가 출력된다.

 LoveLetter.txt.gpg

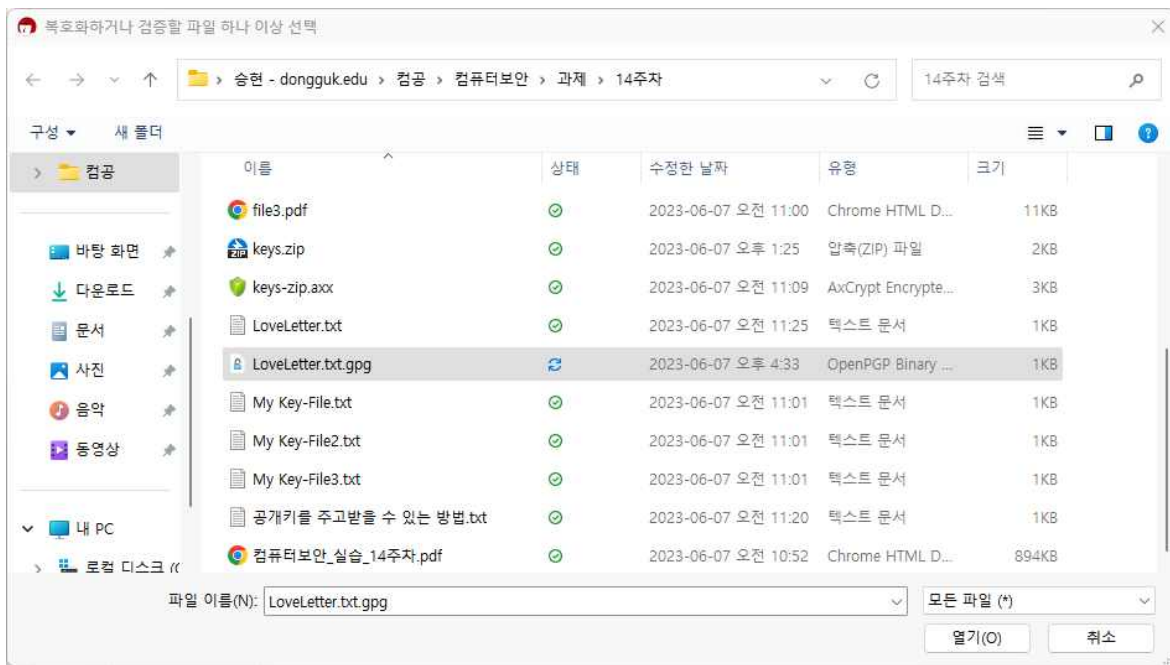
- 암호문이 생성된 모습이다.

4) 암호문 전달 및 복호화 과정을 본인이 생각한 시나리오를 바탕으로 설명하라.

- 앞서 공개키는 클라우드 서비스를 통해 공유된다고 말한 적이 있다. 따라서 Alice와 Bob의 공개키는 클라우드 서비스에서 가져올 수 있다. 이 공개키는 앞서 3번 과정을 통해 클라우드 서비스에서 가져와 각 사용자에게 등록되었다.
- 암호문은 암호화되어 있기에 네트워크를 통해 전달해도 상관없다. 물론 공개키와 같이 클라우드 서비스에서 공유해도 상관없다. 나는 암호문 또한 클라우드 서비스에 저장되어 공유한다고 가정했다.

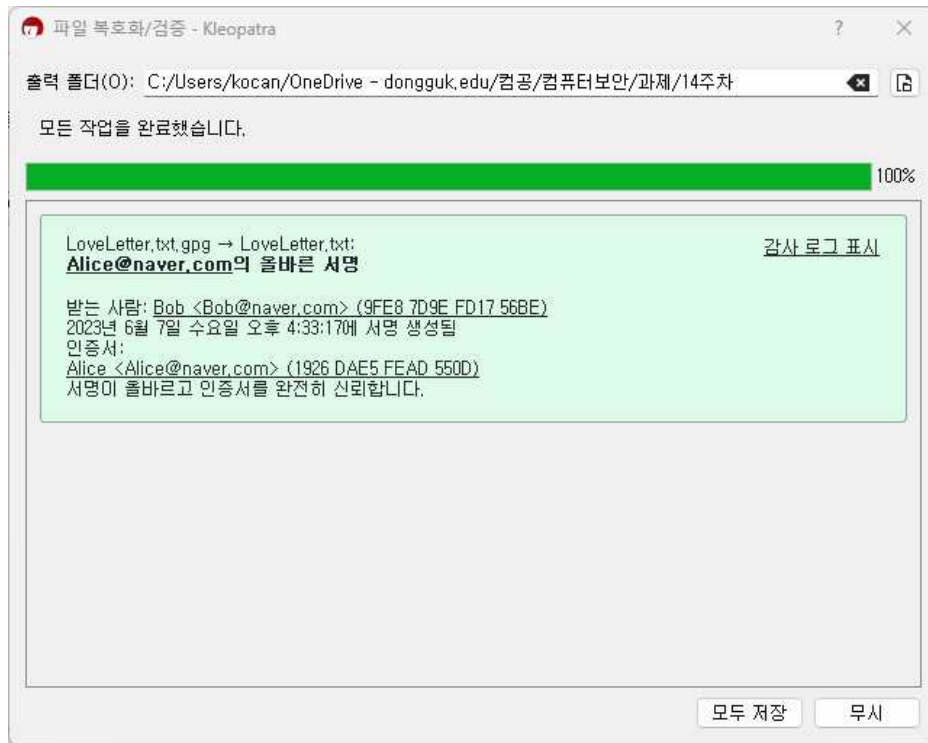


- 메인 화면에서 복호화/검증을 클릭하여 암호문의 복호화를 진행한다.

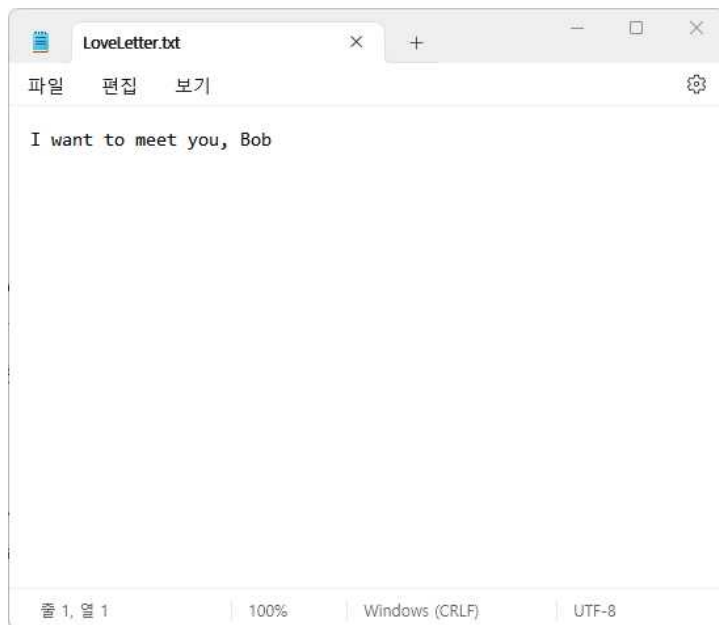


- 복호화할 암호문을 선택한다.

- 앞서 암호문은 클라우드 서비스에 저장되어 공유한다고 말했다. 따라서 클라우드 서비스에 저장된 암호문을 선택한다.



- 암호문의 복호화가 완료되었다.
- 서명이 올바르며 인증서를 신뢰한다는 메시지가 출력된다.
- 모두 저장을 클릭하면 복호화된 암호문이 저장된다.



- 복호화된 암호문을 출력한 모습이다.
- 암호화되기 전 평문과 내용이 일치하는 모습을 확인할 수 있다.

5) 여기서 전자서명은 무엇인가? 어떻게 이용될 수 있는가? 적합할 수 있는가?

전자서명은 서명자를 확인하고 서명자가 당해 전자문서에 서명했다는 사실을 나타내는 데 이용하려고, 특정 전자문서에 첨부되거나 논리적으로 결합된 전자적 형태의 정보를 말하며, 디지털 문서 또는 데이터의 무결성과 신원을 보장하기 위해 사용된다.

전자서명은 다음과 같은 방식으로 이용될 수 있다.

- 문서의 인증: 전자서명은 문서의 인증과 무결성을 보장한다. 서명자의 신원을 확인하고, 문서가 변경되지 않았음을 보장하여 문서의 무결성을 유지한다.
- 거래의 효력과 증명: 전자서명은 법적으로 효력이 있으며, 전자서명이 부여된 문서는 법적인 증거로 사용될 수 있다. 이를 통해 전자적인 방식으로 계약, 합의, 동의 등을 체결할 수 있으며, 필요한 경우에 법적인 증명이 가능하다.

전자서명은 비즈니스에서 계약 체결, 온라인 거래, 공급업체와의 합의 등 다양한 비즈니스 프로세스에 적용될 수 있고, 금융 기관과 보험 회사에서는 전자서명을 사용하여 계약, 보험 가입, 금융 거래 등을 간편하게 처리할 수 있으며, 정부 기관과 법률 분야에서 전자서명은 신분 확인, 행정 문서 처리, 법적인 증거 등에 활용될 수 있다. 이렇게 다양한 분야에서 전자서명은 적합하게 사용될 수 있다.

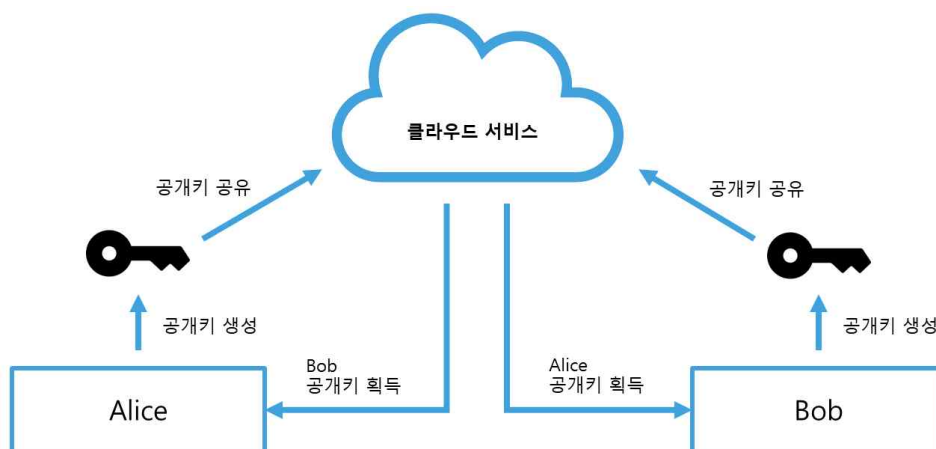
6) 암호화 및 전자서명을 동시에 사용하는 방법을 설명하라.

암호화와 전자서명을 동시에 사용하려면 다음과 같은 과정을 거쳐야 한다.

- i. 문서 암호화: 먼저 암호화할 문서를 선택한다. 선택한 문서는 암호화 알고리즘을 사용하여 암호화됩니다. 이 단계에서는 암호화된 문서를 생성하고 기존의 원본 문서는 보호된다.
- ii. 전자서명 생성: 암호화된 문서에 대한 전자서명을 생성한다. 전자서명은 서명자의 신원과 문서의 무결성을 보장하기 위해 사용됩니다. 일반적으로 전자서명은 디지털 서명 알고리즘을 사용하여 생성된다.
- iii. 서명 검증 및 문서 해독: 전자서명이 검증되면, 문서를 복호화하여 원본 문서에 접근할 수 있다. 전자서명의 유효성을 확인한 후, 암호화된 문서를 복호화하여 원본 문서에 액세스할 수 있다.

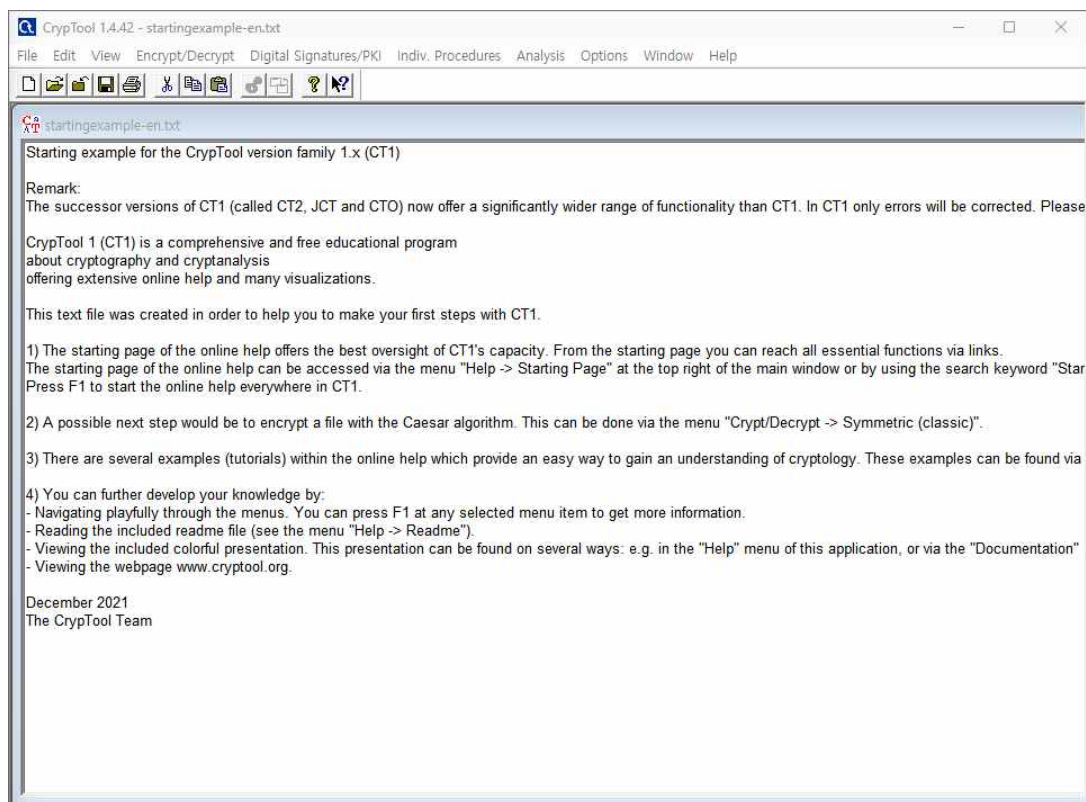
이 과정을 거치면 암호화와 전자서명을 동시에 적용할 수 있다.

7) 공개키를 교환하는 방식에 대해 본인이 작성한 시나리오를 그림으로 작성하라.

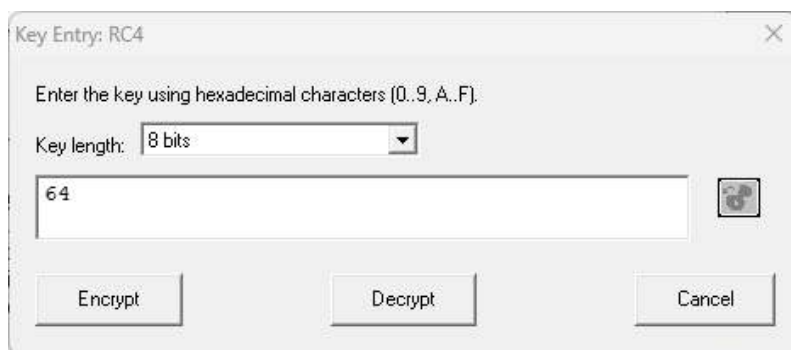


- Alice와 Bob은 공개키를 생성한 다음 클라우드 서비스에 공유한다.
- Alice와 Bob은 클라우드 서비스에서 서로의 공개키를 가져온다.

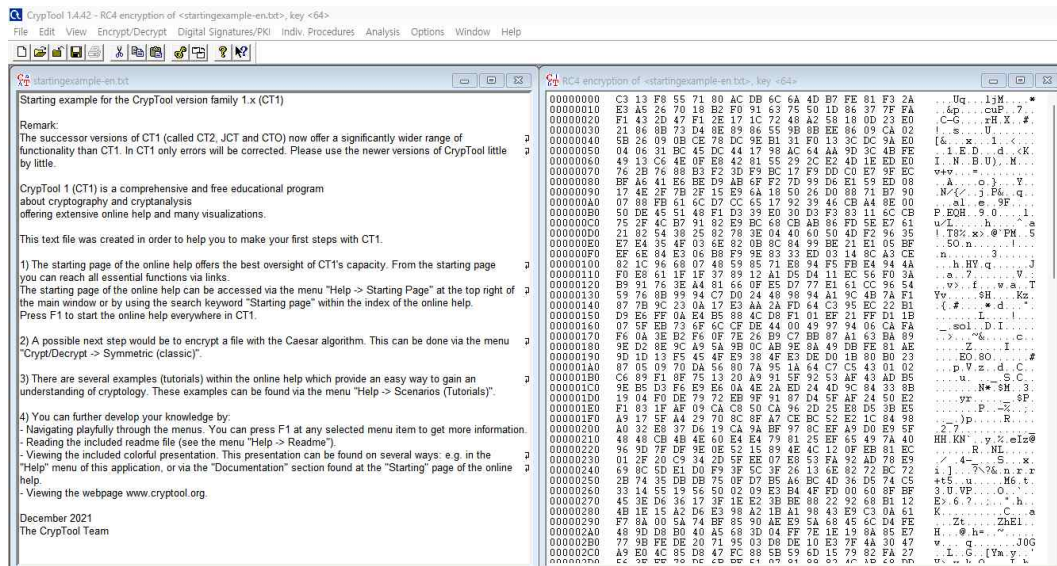
iii. CrypTool



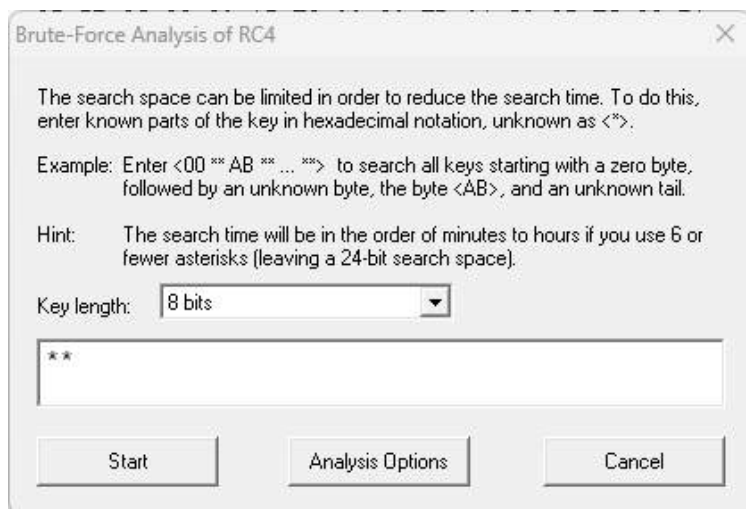
- CrypTool의 메인 화면이다.
- 메뉴에서 Encrypt/Decrypt -> Symmetric (modern) -> RC4를 선택한다.



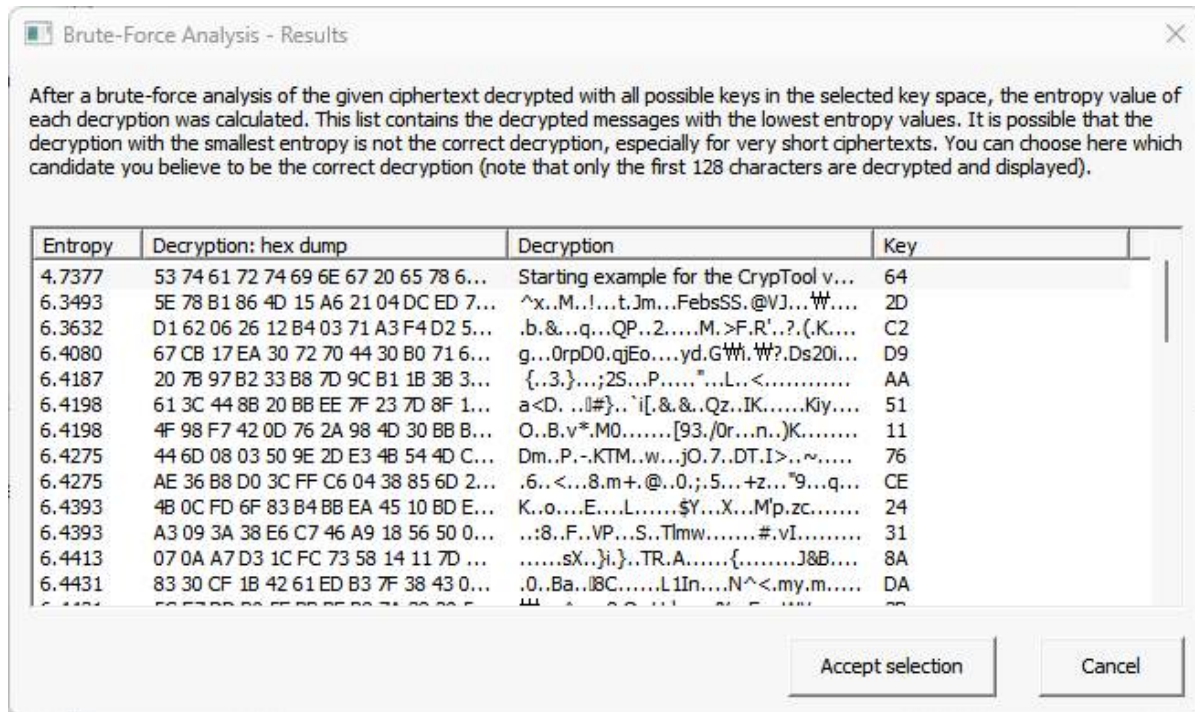
- Key length로 8 bits를 선택한다.
- Key 값은 64로 지정하였다.



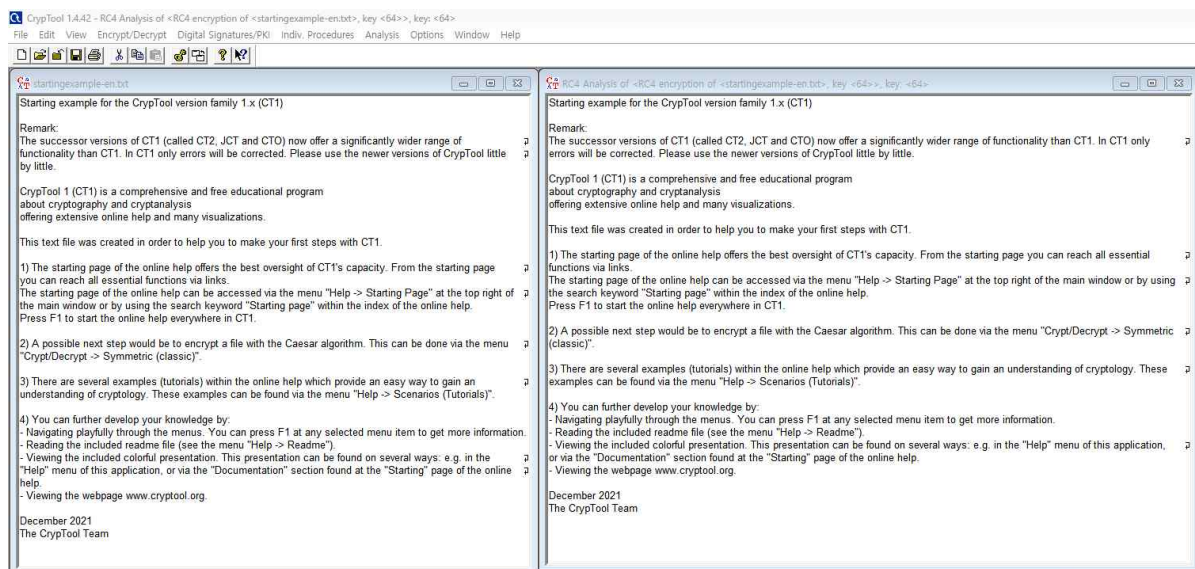
- RC4, 8 bits로 암호화가 된 암호문이 출력되었다.
- 암호화된 대상은 메인 화면에 출력되는 startingexample-en.txt.이다,



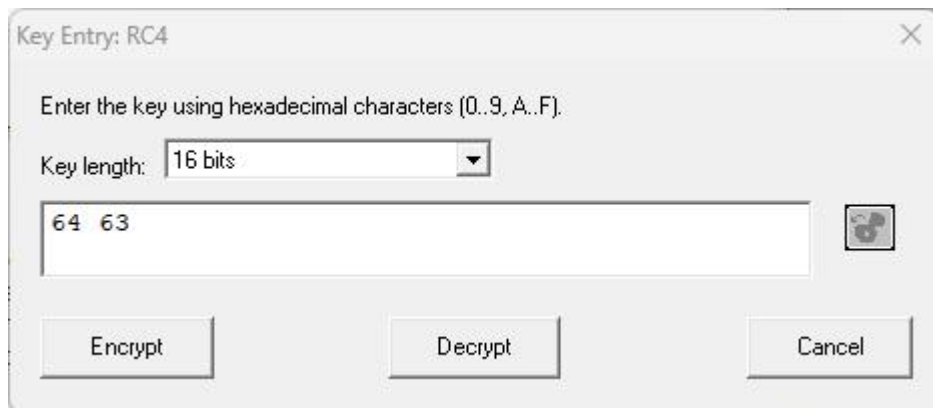
- 메뉴에서 Analysis -> Symmetric Encryption (modern) -> RC4를 클릭하여 복호화를 진행한다.
- Key length로 8 bits를 선택한다.
- 복호화에는 brute force가 사용된다.



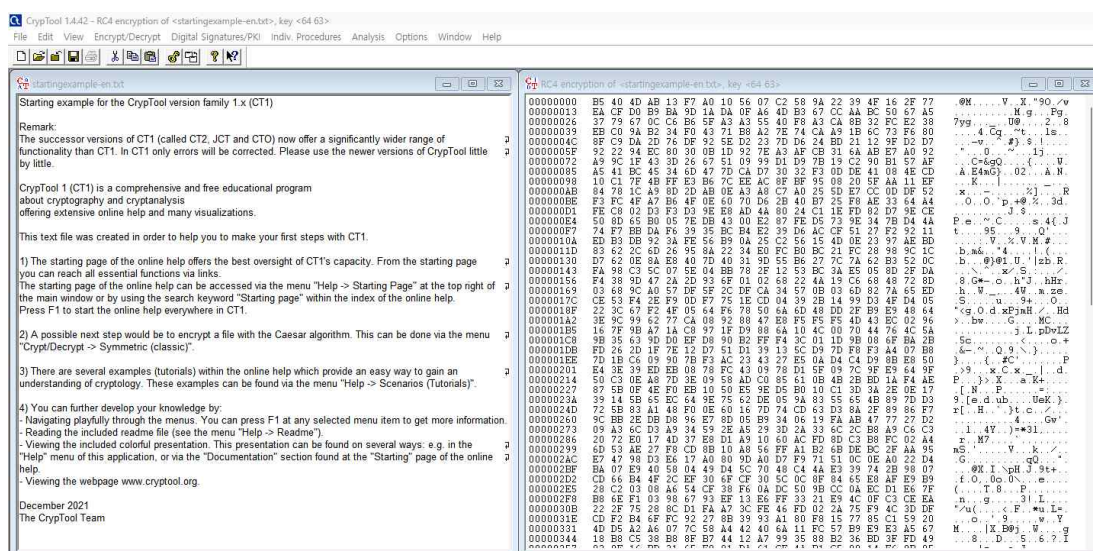
- 한순간에 Decryption이 끝났다.
- Key 후보군이 출력된다.
- Entropy가 가장 낮은 64를 선택한다.



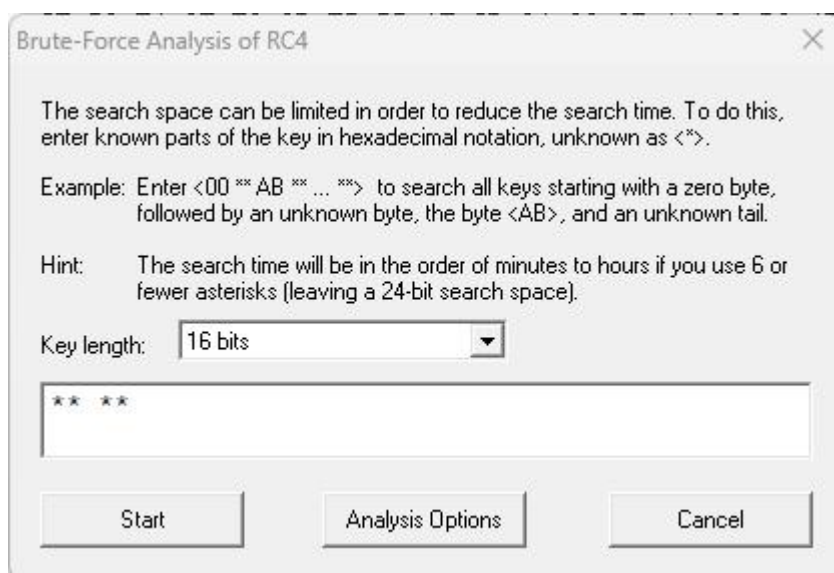
- 암호화되기 전 평문과 복호화된 암호문의 내용이 일치하는 모습을 확인할 수 있다.



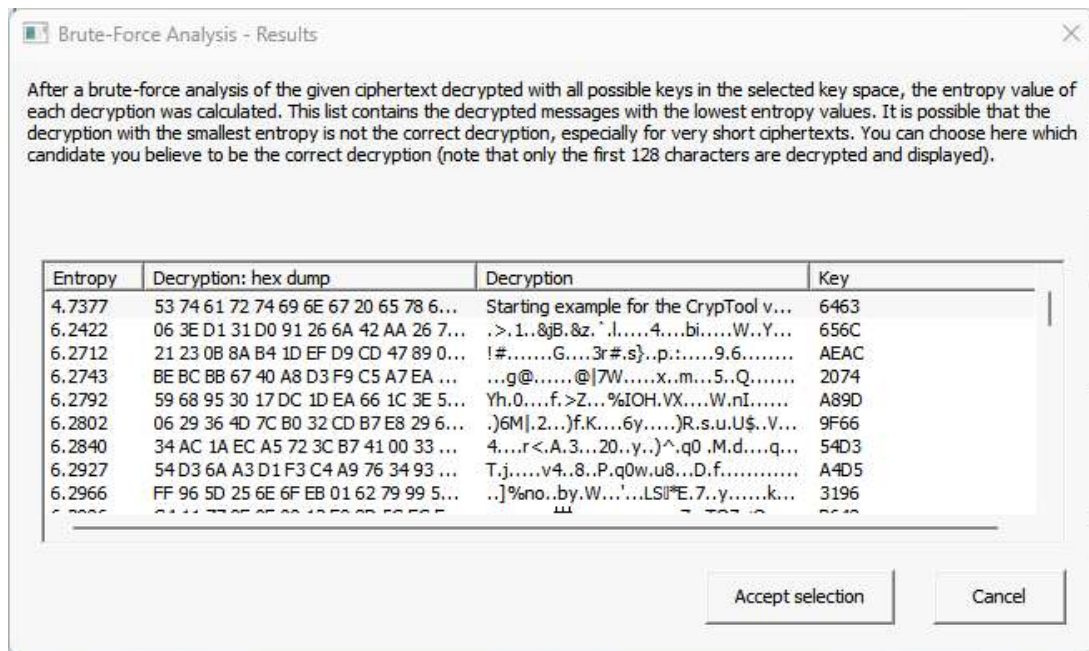
- 다시 한번 암호화를 진행한다.
- Key length로 16 bits를 선택한다.
- Key 값은 6463으로 지정하였다.



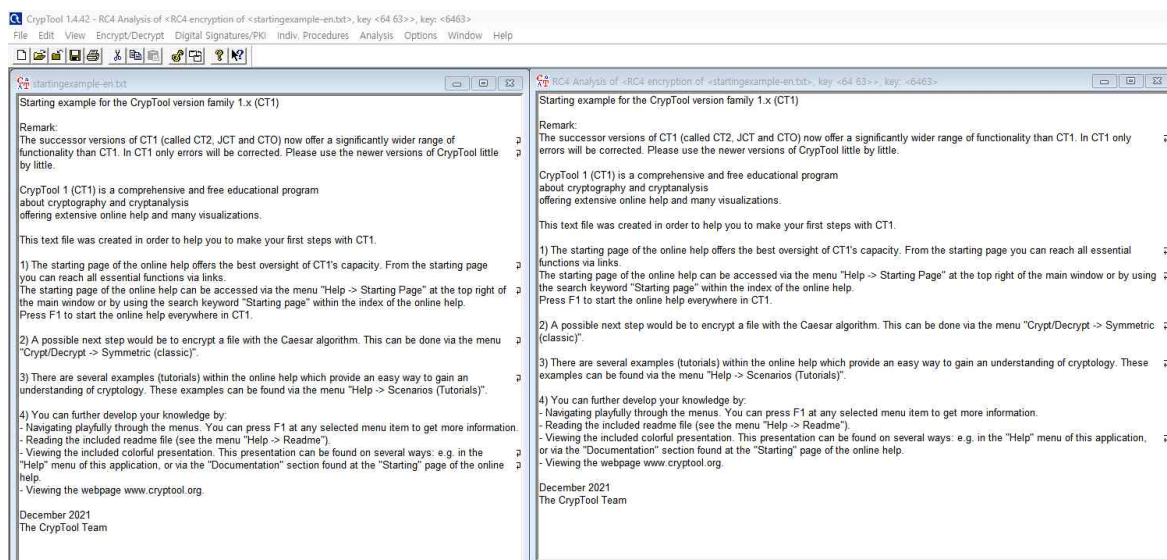
- RC4, 16 bits로 암호화가 된 암호문이 출력되었다.
- 암호화된 대상은 메인 화면에 출력되는 startingexample-en.txt.이다,



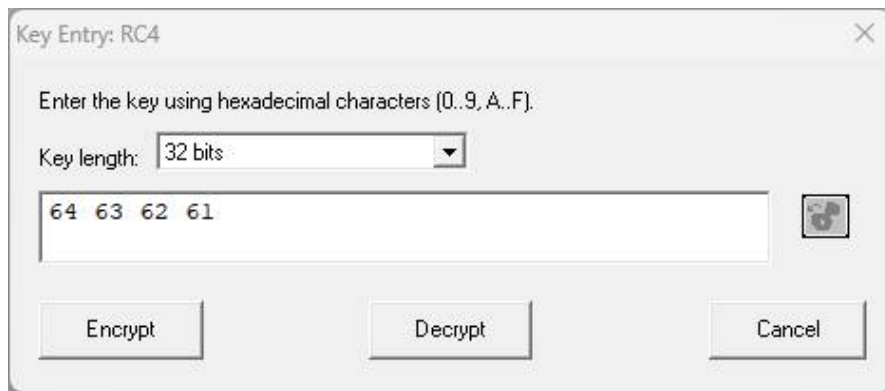
- 메뉴에서 Analysis -> Symmetric Encryption (modern) -> RC4를 클릭하여 복호화를 진행한다.
- Key length로 16 bits를 선택한다.
- 복호화에는 brute force가 사용된다.



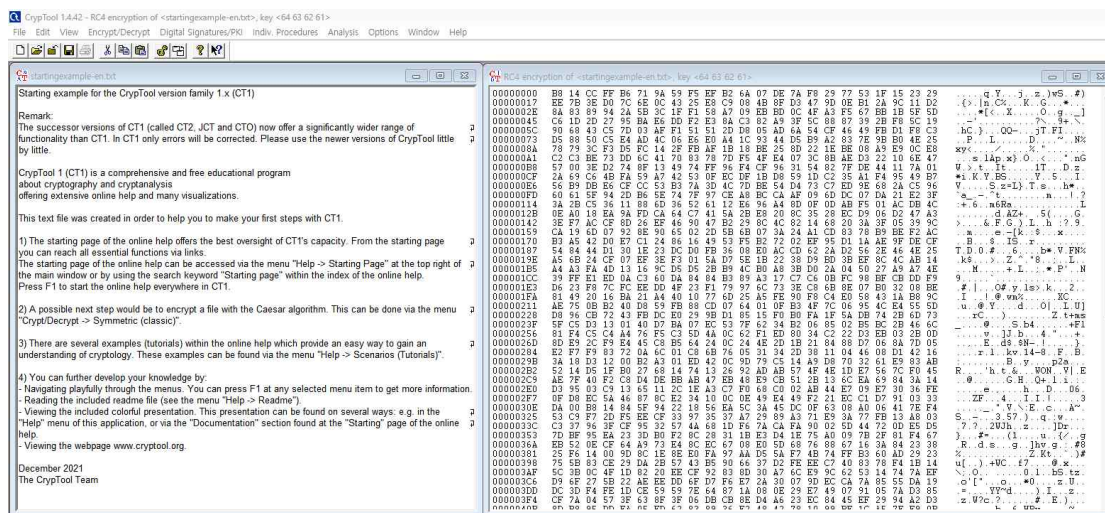
- 16 bits 또한 한순간에 Decryption이 끝났다.
- Key 후보군이 출력된다.
- Entropy가 가장 낮은 6463을 선택한다.



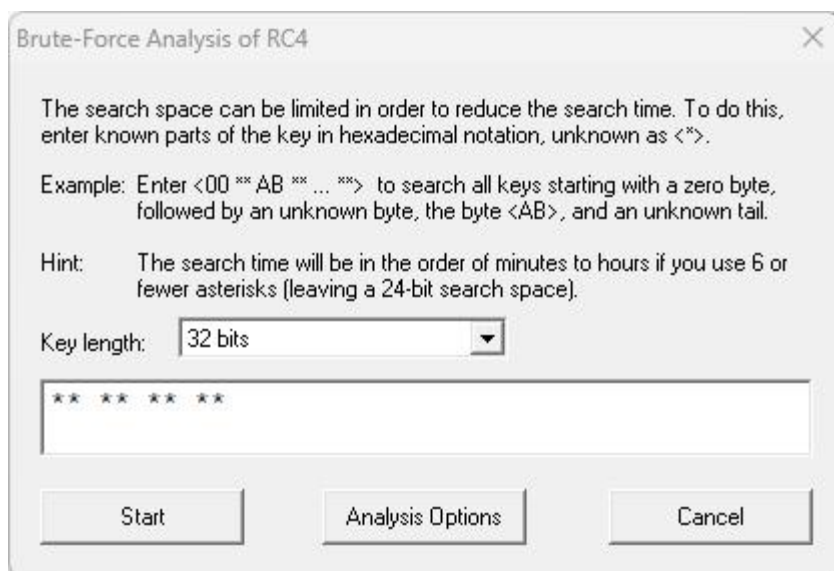
- 암호화되기 전 평문과 복호화된 암호문의 내용이 일치하는 모습을 확인할 수 있다.



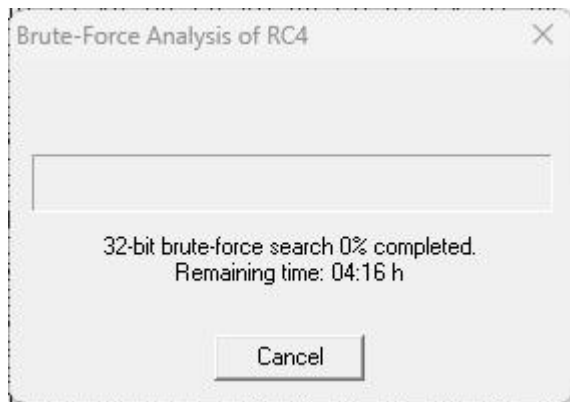
- 다시 한번 암호화를 진행한다.
- Key length로 32 bits를 선택한다.
- Key 값은 64636261로 지정하였다.



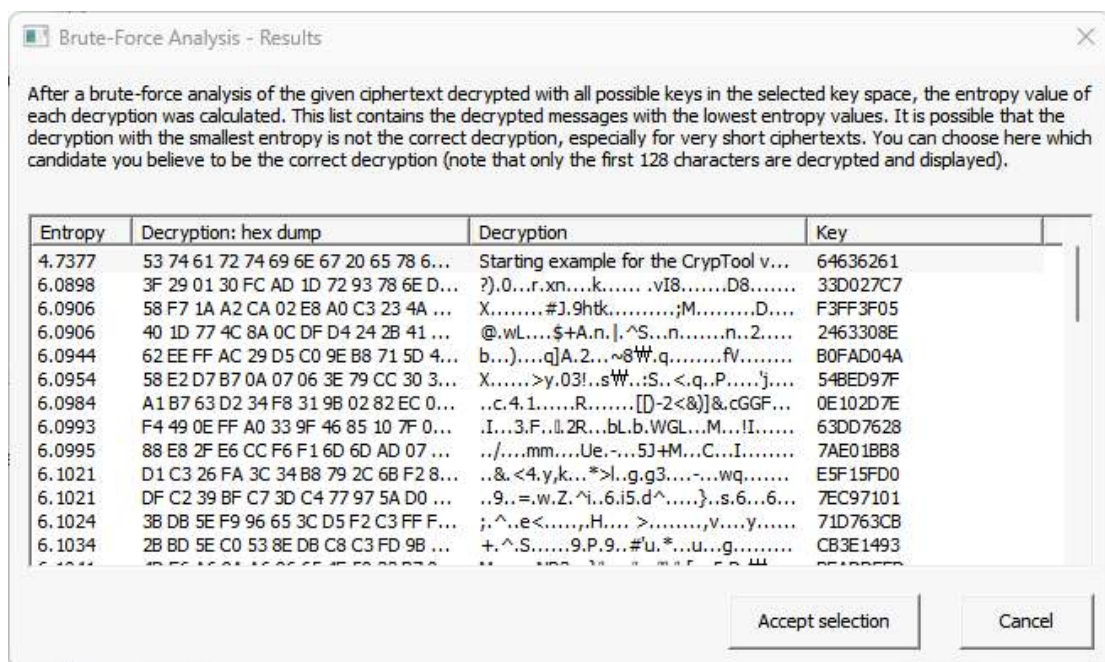
- RC4, 32 bits로 암호화가 된 암호문이 출력되었다.
- 암호화된 대상은 메인 화면에 출력되는 startingexample-en.txt.이다,



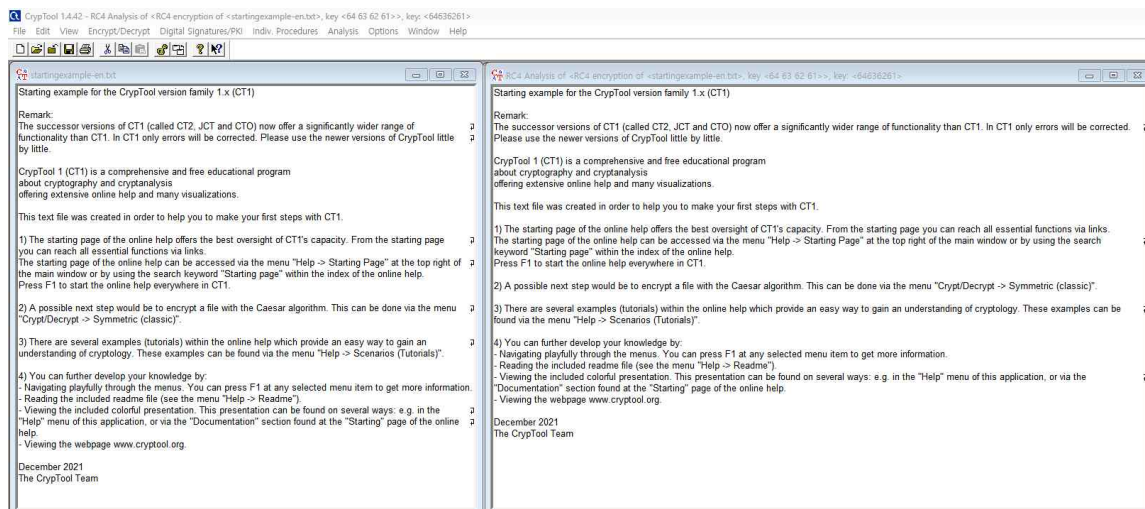
- 메뉴에서 Analysis -> Symmetric Encryption (modern) -> RC4를 클릭하여 복호화를 진행한다.
- Key length로 32 bits를 선택한다.
- 복호화에는 brute force가 사용된다.



- 32 bits key로 암호화된 암호문을 brute force로 복호화를 진행하니 8, 16 bits 보다 시간이 오래 걸린다.
- 남은 시간이 4시간 16분임을 출력하고 있다.

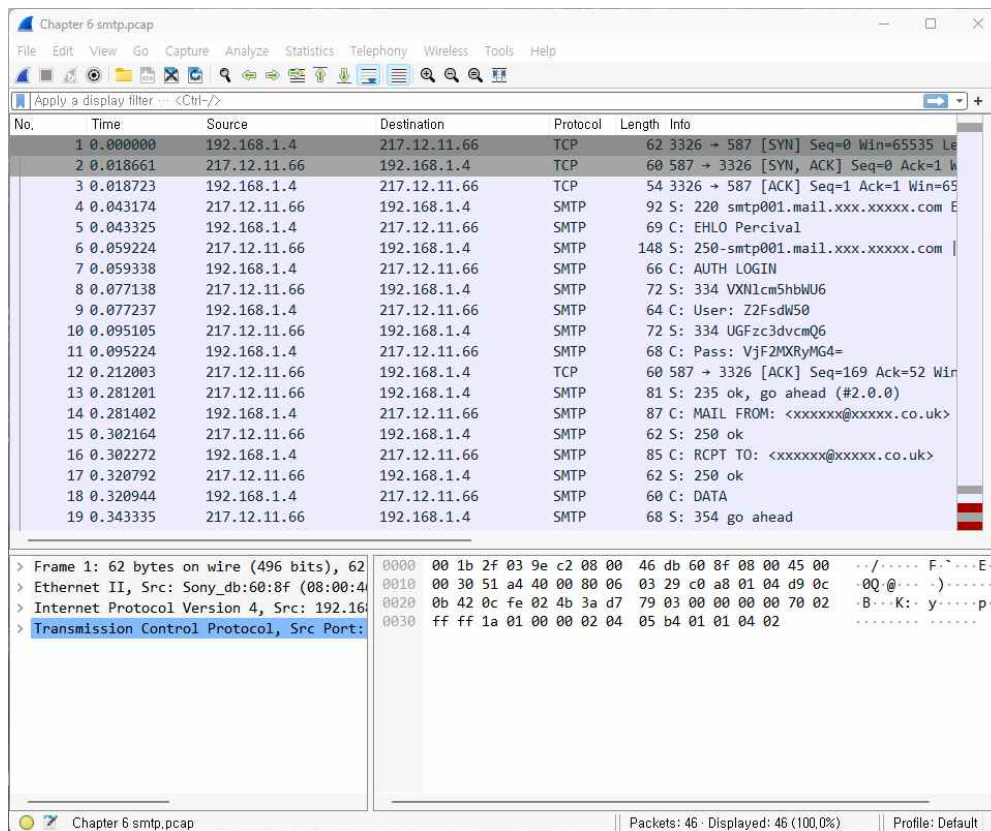


- 16 bits 또한 한순간에 Decryption이 끝났다.
- Key 후보군이 출력된다.
- Entropy가 가장 낮은 64636261을 선택한다.



- 암호화되기 전 평문과 복호화된 암호문의 내용이 일치하는 모습을 확인할 수 있다.

iv. 이메일 사용자 이름과 패스워드 추출



The image shows a Wireshark capture of an SMTP session. The main pane displays a list of 19 packets. The details pane for the selected packet (No. 1) shows the frame structure: Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol. The packet list pane shows the following details:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.4	217.12.11.66	TCP	62	3326 → 587 [SYN] Seq=0 Win=65535 Len=0
2	0.018661	217.12.11.66	192.168.1.4	TCP	60	587 → 3326 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0
3	0.018723	192.168.1.4	217.12.11.66	TCP	54	3326 → 587 [ACK] Seq=1 Ack=1 Win=65535 Len=0
4	0.043174	217.12.11.66	192.168.1.4	SMTP	92	S: 220 smtp001.mail.xxx.xxx.com E
5	0.043325	192.168.1.4	217.12.11.66	SMTP	69	C: EHLO Percival
6	0.059224	217.12.11.66	192.168.1.4	SMTP	148	S: 250-smtp001.mail.xxx.xxx.com
7	0.059338	192.168.1.4	217.12.11.66	SMTP	66	C: AUTH LOGIN
8	0.077138	217.12.11.66	192.168.1.4	SMTP	72	S: 334 VXN1cm5hbWU6
9	0.077237	192.168.1.4	217.12.11.66	SMTP	64	C: User: Z2FsdW50
10	0.095105	217.12.11.66	192.168.1.4	SMTP	72	S: 334 UGFzc3dvcmQ6
11	0.095224	192.168.1.4	217.12.11.66	SMTP	68	C: Pass: VjF2MXRyMG4=
12	0.212003	217.12.11.66	192.168.1.4	TCP	60	587 → 3326 [ACK] Seq=169 Ack=52 Win=65535 Len=0
13	0.281201	217.12.11.66	192.168.1.4	SMTP	81	S: 235 ok, go ahead (#2.0.0)
14	0.281402	192.168.1.4	217.12.11.66	SMTP	87	C: MAIL FROM: <xxxxxx@xxxxx.co.uk>
15	0.302164	217.12.11.66	192.168.1.4	SMTP	62	S: 250 ok
16	0.302272	192.168.1.4	217.12.11.66	SMTP	85	C: RCPT TO: <xxxxxx@xxxxx.co.uk>
17	0.320792	217.12.11.66	192.168.1.4	SMTP	62	S: 250 ok
18	0.320944	192.168.1.4	217.12.11.66	SMTP	60	C: DATA
19	0.343335	217.12.11.66	192.168.1.4	SMTP	68	S: 354 go ahead

The details pane for the selected packet (No. 1) shows the following structure:

- Frame 1: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
- Ethernet II, Src: Sony_db:60:8f (08:00:42:60:8f:00), Dst: 217.12.11.66 (01:00:5e:00:00:00)
- Internet Protocol Version 4, Src: 192.168.1.4, Dst: 217.12.11.66
- Transmission Control Protocol, Src Port: 3326, Dst Port: 587

- Wireshark에서 SMTP.pcap 파일을 연 모습이다.
- 수많은 패킷의 전송 시간과 송신지, 수신지, 프로토콜, 크기, 정보가 출력된다.

Wireshark · Follow TCP Stream (tcp.stream eq 0) · Chapter 6 smtp.pcap

```
220 smtp001.mail.xxx.xxxxx.com ESMTP
EHLO Percival
250-smtp001.mail.xxx.xxxxx.com
250-AUTH LOGIN PLAIN XYMCOKIE
250-PIPELINING
250 8BITMIME
AUTH LOGIN
334 VXN1cm5hbWU6
Z2FsdW50
334 UGFzc3dvcmQ6
VjF2MXRyMG4=
235 ok, go ahead (#2.0.0)
MAIL FROM: <xxxxxx@xxxxx.co.uk>
250 ok
RCPT TO: <xxxxxx@xxxxx.co.uk>
250 ok
DATA
354 go ahead
Reply-To: <xxxxxx@xxxxx.co.uk>
From: "Xxxxxx xxxx" <xxxxxx@xxxxx.co.uk>
To: <xxxxxx@xxxxx.co.uk>
Subject: Testing testing 1 2 3 (Multiple attachments)
Date: Sat, 14 Jul 2007 10:31:37 +0200
MIME-Version: 1.0
Content-Type: multipart/mixed;
        boundary="-----_NextPart_000_0000_01C7C602.28C76960"
X-Mailer: Microsoft Office Outlook, Build 11.0.5510
Thread-Index: AcfEkVgAjSI1YuAtTWmtYdhKzktf7w==
X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3138
X-MS-TNEF-Correlator: 00000005304EB0A4426C6489833E3DA4C08EF4C448B2C00

This is a multi-part message in MIME format.

-----_NextPart_000_0000_01C7C602.28C76960
Content-Type: text/plain;
        charset="us-ascii"
```

20 client pkt(s), 10 server pkt(s), 18 turn(s).

Entire conversation (15 kB) Show data as ASCII Stream 0

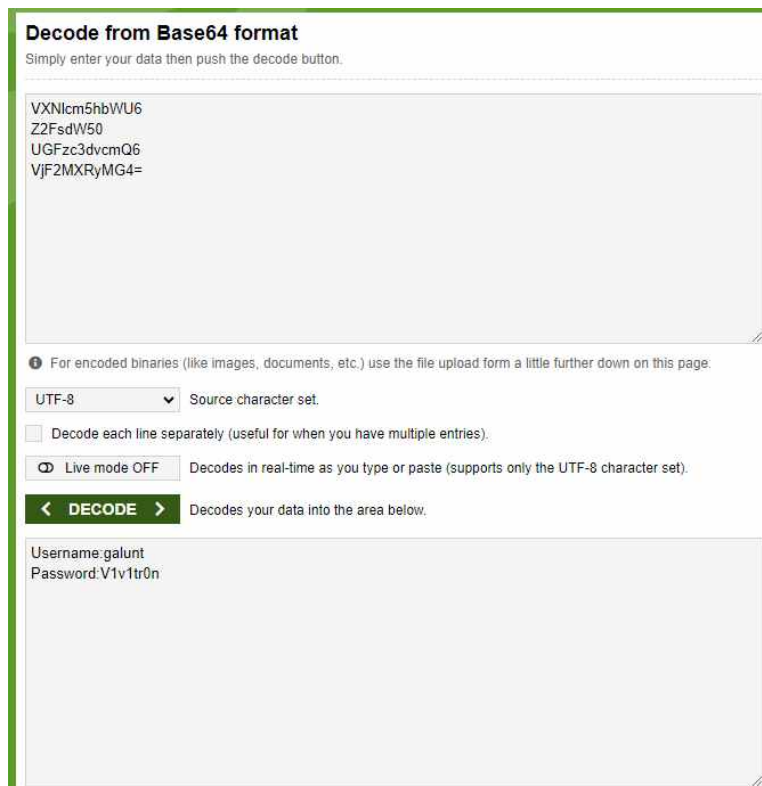
Find: Find Next

Filter Out This Stream Print Save as... Back 닫기 도움말

- Follow TCP Stream을 선택한 모습이다.
- TCP stream에 대한 정보가 출력된다.
- 여기서 334 값을 찾아 인코딩된 사용자 이름과 패스워드 문자열을 찾아야 한다.



- 334 값을 찾는 모습이다.
- Base64로 사용자 이름과 패스워드 문자열이 인코딩되어있다.



- 인코딩된 사용자 이름과 패스워드 문자열을 다시 디코딩한 모습이다.
- 사용자 이름은 galunt, 패스워드는 V1v1tr0n이다.
- 원래 디코딩하면 사용자 이름과 패스워드 문자열이 붙어서 출력되나, 내가 임의로 줄바꿈을 하여 보기 좋게 하였다.

3) 느낀 점

- 오늘부로 컴퓨터보안 실습이 전부 끝났다. 길다면 길고 짧으면 짧은 14주간의 실습이었다. 그동안의 실습을 돌아보면 어려운 과제도 종종 있었지만 재밌었다고 생각한다. 특히나 리눅스를 사용하는 실습을 진행하면서 리눅스에 대한 이해도가 많이 늘어났다. 평소에 CLI를 잘 사용하지 않아서 처음에는 리눅스를 사용하려니 많이 불편했었다. 그래도 실습 진행하면서 리눅스를 많이 사용하니 적응을 빨리할 수 있었다. 리눅스에 있는 여러 가지 기능을 사용해볼 수 있어서 좋았다. 만약 컴퓨터보안을 듣지 않았다면 아마 리눅스를 사용하지 못하고 졸업했을 것이다. 개인적으로 windows 운영 체제보다 리눅스 운영 체제를 사용하는 것이 더 재미있다. 다만 리눅스에서 사용되는 명령어를 배우면서 windows에서 사용하는 명령어랑 혼동되는 경우가 좀 생긴 것 같다. 명령어가 통일되었으면 좋겠지만 그럴 일이 없을 거 같아서 슬프다. 둘 다 까먹지 않게 상기하면서 다녀야겠다. 아무튼 중간중간에 실습이 어려웠고 시간 많이 소모해서 조금 힘들었지만 그만큼 보람찬 실습이라고 생각하고 지식을 많이 쌓을 수 있어서 좋았다. 한 학기 동안 실습 진행하신 조교님 너무 감사드리고 앞으로 좋은 일만 가득했으면 좋겠습니다.