Katura Harvey, Sara Leroy

Project 4 ATM design and implementation

Design document

1. Describe your overall protocol in sufficient detail for a reader to understand the security mechanisms you put into place, without having to look at the source code. This must include the format of the files you create (with the init and bank programs) and the messages you send between ATM and bank.

The init program generates .bank and .atm files. The bank and ATM communicate with each other through the router. All information from bank, using a shared symmetric key generated by init, is sent encrypted for ATM to decrypt and vice versa.

To send a message:

·       A counter is added at the beginning of the message

·       The digest is computed

·       A random iv is computed until one with no null bytes is obtained. This is computed in both ATM and bank

·       The digest concatenated to the message is encrypted with the iv and secret key

·       The iv, concatenated with the ciphertext, is sent

To receive a message:

·       The first 16 chars obtained are the iv

·       Using the iv and secret key, the remainder of the message is decrypted

·       The next 128 chars are the digest, the remainder is the message

·       The digest of the message is computed and checked to see if it matches the received digest

·       The first number in the message, the counter, is verified to see if it is >= the stored counter.

When a user creates an account with the bank, the bank creates the user’s card adding an ascii character key so that when the card is accessed later on, it is the original card and not a forged one. Only when the create-user command is followed by valid set of arguments does the user receive a card.  The programs generate files:

        .atm and .bank files which store a 32 byte secret key assigned by init

        .card files which store a random 32 printable ascii character key

When users interact with the bank, the bank will have these behaviors in response to user commands:

create-user:

1.     Bank checks for valid arguments and whether or not the user exists

2.     If valid, bank creates user.card file adding the random ascii characters at the beginning and updates number of users

deposit:

1.     Bank checks if the user exists

2.     If user exists, it then checks that the amount to deposit is not greater than the INT\_MAX value

3.     If valid arguments, bank increases user balance by deposit amount

Balance:

1.     Bank checks if user exists

2.     If user exists, bank prints balance

When users interact with the ATM, the ATM and the bank will have these behaviors in response to user commands:

ATM-Bank interface:

begin-session:

1.     ATM sends request for <username> to bank

2.     If found, bank sends pin and key on card file. Else sends not-found

3.     ATM checks if <username>.card exists, and if key matches, else fails

4.     ATM prompts for PIN and checks if it matches, else fails

5.     ATM sets current\_username = username and logged\_in = true

withdraw:

1.     ATM sends request for <username> to withdraw amount

2.     Bank looks up username’s balance

3.     If amount is less than equal to balance, subtract amount from balance and send success. Else sends fail

4.     ATM prints appropriate message

balance:

1.     ATM sends request for username’s balance

2.     Bank sends balance

3.     ATM prints balance

2. List, one by one, the specific attacks that you have considered and describe how your protocol counters these threats. This is critical for how we will be grading this part of the project.

**An adversary who is *not* in possession of a user's *ATM card*, but knows the user's PIN:**

o The ATM checks both that the card file is present when a user tries to start a session, and a valid pin. If no user card is present, then an error message will be printed and the ATM will go back to the prompt. If the adversary tries to give a fake card file, he/she will still not be able to login because each card file generated by the bank contains a random key (which does not depend on the PIN). Then the ATM will go back to the prompt.

**An adversary who is *not* in possession of a *user's pin*, but has access to the user’s card:**

o    The ATM checks for a valid pin when a user attempts to start a session. Without the pin, the adversary will not be able to log into another user’s account simply using that user’s card. The card does not contain information that the adversary can use to guess what the pin can be.

**An adversary inspecting all packets sent through the router:**

o     The digest concatenated to the message is encrypted with the iv and secret key. The iv, concatenated with the ciphertext, is sent. To decipher this, the attacker would need the key which was originally generated by the init program. Therefore, merely by inspecting packets, the attacker will not be able to infer any data.

**An adversary modifying all packets sent through the router:**

o    When the bank or ATM attempt to decrypt the sent packages, the digest received are compared with the digest generated by the ATM or bank file. In case they are different, the ATM/bank ignore the commands and return to prompt.

**An adversary dropping some or all packets sent through the router:**

o    Because of the counter added at the beginning of each messages, if the adversary were to drop a packet, the bank/ATM would detect that, by checking the counter in the message against the counter the bank/ATM store. In the case that the counters are not equal to each other, whatever command was sent is ignored and the program returns to the prompt

**An adversary creating new packets to send to ATM or bank through the router:**

o    Each packet received by ATM and bank, is checked for a valid counter and a valid digest. The adversary might have guessed the count of the next packet to be sent however, even if the bank/ATM were able to compute the digest to get a message, if the digest created by the bank/ATM from the message does not match the digest received, the command and message are ignored and the program returns to prompt.

**An adversary duplicating some or all packets sent through the router (replay attacks)**

o   encryption and decryption is done with random ivs and a counter

o   messages passed between atm and bank will be less than ULONG\_MAX messages in one session

**In case of a buffer overflow attack:**

o   the integer inputs for create-user, balance, deposit, and withdraw are all checked against INT\_MAX

o   all string argument inputs are checked for maximum array size and in cases input sizes are greater, the program will output error messages

**In case of a “dictionary attack” to guess the user’s pin:**

o   a time stamp is recorded along with the user’s input for a pin, when he/she attempts to log into the ATM. When this pin, checked against the user’s valid pin, fails, the ATM checks the number of failed attempts in a 30 second interval. If the user fails 3 times within 30 seconds, the ATM will lock for 60 seconds, and will not accept any user input.

Multiple users on the same ATM

o

**A user logged in with a valid account attempts to log into another user’s account**

o    the ATM keeps a boolean value “logged\_in” which is set to 1 after a user begins a session. This boolean value is only set to 0 in end-session. Therefore, if afterwards a user inputs begin-session before end-session, when the logged\_in boolean is checked and validated to true, the atm prints a message and returns to the logged\_in user’s prompt.

3. You can also mention threats that you chose to ignore because they were unimportant, as well as threats that were important but you were unable to address.

**An adversary in possession of the user’s card may rename it to use it as his/her own:**

o   It is unimportant because the ATM does not check the card for valid login. The ATM checks that the user name matches a valid pin. Even if the adversary could inspect the user’s card, he/she would not be able to infer the pin.

o   It is also unimportant because the bank does not check the user’s card for the balance and deposit commands.

o   In this scenario it is assumed that the adversary will not have both access to the pin and the user’s card.