Banking Security: Protocol flaws in Chip and PIN
March 14, 2012


Background:
Credit cards have been an important part of world finance since their invention in the early 1900s. Some of the earliest widespread cards were the “Charge-It” and the “Dinner's Club” cards around the middle of the century. These were first made of paper and were unverified. This changed in the 1960s with the advent of the magnetic stripe.

“Magstripe” cards, while allowing some level of verification of the card, have many problems. The easiest problem to see is wear: the magnetized strip becomes demagnetized over time, through standard handling and through magnetic field. The bigger problems also have to do with magnetized strip. Since the important information on the card is completely in the stripe, all an advisory needs to do is swipe the card once in a reader and the card can be duplicated. This has been done in restaurants and other retail establishments for years: once the waiter or staff walks away with your card, they can swipe it as many times as they want without you being the wiser. Another common attack against magstipes is the skimmer. A skimmer is a small device the sits between or within a standard card-reader, thus allowing the advisory to copy the card without ever handling the card. As technology has gotten better, skimmers have become smaller and wireless, making their detection even harder. To deal with these problems, the “Chip and PIN” was proposed (and put into production).

Chip and PIN:
Chip and PIN (based on the EMV (Europay, MasterCard, Visa) protocol) takes the basic idea of Magstripe and adds actual interaction between the card and the machine. Cards are embedded with a smart chip that stores PIN and card information. Terminals ask the card to verify the PIN number and then proceed to do transfer in real time (if the terminal is 'online' otherwise, processing might take 2-4 days). The main goals of this system are to prevent fraud and to shift liability for fraud from banks to customers. Of course, the main goal (fraud prevention) has not actually been met in practice. To see why, we look at the EMV protocol.

EMV protocol:
EMV protocol has three stages: Card Authentication, Cardholder Verification, and Transaction Authorization. The Card Authentication process is as follows: 1) the application (ATM, credit, &c) is selected from a pre-approved list on the card, 2) The transaction is begun (bank specific
methods), and 3) a record is returned with account information and the cardholder verification method (CVM) verified by a RSA signature. This signature can be generated either statically or dynamically depending on the capability of the card (Static data authentication (SDA) (pre 2009) vs. Dynamic (DDA)). These static cards led to a simple type of fraud using what was called “Yes cards”. These “Yes cards” were duplicated cards with a modified smart chip that always claimed that the PIN entered was correct. This only worked with SDA cards and offline terminals as online terminals uses some amount of live verification with the banks.

The Cardholder Verification stage is simpler than Card Authentication. Cardholder Verification takes the CVM and uses the options in it to verify the cardholder (the options being PIN, signature, or nothing). If the terminal is online, PIN verification works by encrypting the inputted PIN and verifying the answer with the issuer of the card. Yet you are dealing with an offline terminal, the card itself is asks, with the card returning the code 0x9000 if the PIN is good and 0x63Cy if it is bad.

The last stage, Transaction Authorization works by asking the card if a transaction is allowed and then asking the issuer to complete said transaction. This is all done with cryptographic verification. The Terminal generates an AC which includes the type of transaction, the amount, the currency, some transaction verification information (TVM), and a nonce. The card then response with either a ARQC for approval or a AAC for a denied request. The ARQC includes information based on the AC and standard data on the account and type of transaction and some information about the current session (IAD). This ARQC is sent to the Issuer by Terminal and the issuer responses with an ARC (what happened) and an ARPC (MAC of ARQC and ARC: i.e. verification of actual issuer). If the transaction worked out, a Transaction Certification (TC) is generated by the card and based to the Terminal (and the Issuer).

The Attack:
The attack is a man in the middle attack. The key part of the attack is in the PIN verification. The attack uses the fact that the terminal always gets the same response for an accepted PIN (0x9000). By intercepting the Cardholder Verification stage and responding for the card, this attack, through exploiting the protocol verification steps, is able to complete transactions without permission from the card or (likely) any record. The reason why this is unlikely to be detected is the fact the TVM only notes failure (defaulted to success) and the IAD only could contain information of PIN attempts (in addition, this information would not be readable by the terminal, thus, the transaction would still be pushed by the terminal (completed if offline)). Also, the final “Verified by PIN” on the printed receipt is only the view of the Terminal (the card or Issuer might think that it was verified by signature). The attack was doable with off-the-shelf hardware and some simple Python, all coming out to about $199. The tools needed were terminal, a fake card, a FPGA board, a PC, a card reader, and, of course, a stolen card. All together, this could be concealed in a backpack. However, a limitation of this attack is the fact that the fake card has to remain physically attached to the rest of the hardware. This is an acceptable limitation as many merchants allow their customers to swipe their own cards and it is even customary to turn away from the customer while they do so (to avoid seeing the PIN).
More Why? and Solutions:
As previously stated, the key reasons why this attack works are that Cardholder Authentication is broken and that the internal views of what and how things have been approved are never compared. This is possible due to the nature of the design process (closed with not external review), the documentation (large, complex, and undisciplined (mixes protocol with implementation)), and, again, the fact that the protocol is not secure. The banks have suggested two improvements: moving all cards to DDA and a combined authentication (PIN + Signature). Neither of these do anything to mitigate the attack above. Another solution is to give terminals the ability to read IADs. Here we run into the problem that banks are allowed to chose whatever methods they want in communication between the cards and the issuers, thus making all terminals able to read all IADs would be a daunting task even if the banks would allow the communication protocols to be revealed. A more reasonable solution might be to make cards request used CVM from terminals. In fact, one card already does this but fails to make sure that the later stages maintain this. However, this would require changing all the cards which would take years. As for the liability front, this attack teaches us that the printed receipts are in no way trustworthy documents and that the IAD is the only source of actual transaction receipt data.

More Attacks:
There are other attacks against Chip and PIN. A major category of these are relay attacks. Relay attack involve fake or comprised POS devices that 'relay' a legitimate cards interactions to a fake card in another location. The only real problem with this problem (besides compromising the POS device) is getting the timing correct (your fake card must be in the real POS at the same time as the real one). Defenses against relay attacks involve encrypting communications between terminals and cards (expensive), timing analysis (easier), and having merchants swipe cards (fakes, so far, need to be connected to external equipment). Another attack is EMV skimming. This is similar to Magstripe skimming but with EMV protocol cards.

Conclusion:
Chip and PIN is widespread and completely broken. The lack of authentication between stages and entities allows for a simple attack that highlights the problems of isolation of steps. In addition, the lack of external review and bad documentation allowed these flaws to fester for a very long time. However, there are ways to fix Chip and PIN. These fixes would be hard to implement and very expensive. Patching the system would likely add more, hard-to-detect flaws.

Discussion:
Q: (Why isn't there a) race to fix this (does it have to do with the liability shifting)?
A: The banks are a cartel (not fixing it is best for all of them) – this is similar to Big Tobacco not revealing that smoking is really bad for you
Q: What about the long term liability (following the Big Tobacco thread)?
A: It might be good to be the first whistle-blower company for this but why would people believe it? Saying that is not safe and that you have a new protocol sound very similar to the Magstripe to EMV transition.

Q: Why not just get rid of offline terminals?
A: Offline terminals are nice for banks (already there, cheaper)

Q: Point of Service (POS) devices seem to be the big problem, why not have them use dynamic encryption or constantly update certificates or …?
A: This might be possible but it is similar to terminals with the added angle that POS devices are so necessary to businesses making money that none want to mess with them (if it isn't broke...)

Q: This is a good paper (just an attack, no formal problems, new discoveries, &c.)?
A: Not really but it effects many people and lots of money. Also, at least it's an attack on the protocol and not an implementation (directly). Though, some more formalism in the paper would have been welcome.