Money for Nothing, Chips for Free

AUTHOR: Peter Honeyman

[**Editor's note:** This article was shortened to fit the print format. The full version will be available online at https://phrack.org]

0 - Abstract

How a team of academic hackers discovered bugs in a widely deployed smart card payment protocol, kept them secret (until now!), and turned what they learned into parties, papers, conferences, and advanced degrees.

1 - Introduction

It's 1998. I am the fledgling director of the Center for Information Technology Integration, or CITI, a small research center at the University of Michigan focused on IT infrastructure. My senior staff and I are sitting in a small conference room across the table from three gentlemen in suits.

My staff sits in silence throughout the meeting. I introduce my team and ask the visitors "Are any of you authorized to make arrests?" The head suit replies, "He is, and I am, but he is not."

I make eye contact with my team. This is what we talked about last night.

The suits are from the Electronic Crimes Branch of the United States Secret Service, charged with protecting the nation's financial infrastructure.

They heard through the grapevine that we were messing with the currency. So po-po is in the house and it's Shut The Fuck Up Friday for my staff -- let's not get arrested today -- while I try to calm the waters.

Let's back up, though.

A couple years earlier, a former doctoral student had set CITI up with an industry research grant to develop smart card-based key management for their secure video conferencing project. That project taught us a lot about smart card innards and helped us build some useful tools for smart card hacking, e.g., card readers and card extenders that let us intercept and control communications between a smart card and a smart card reader.

But let's back up some more: smart cards?

2 - Smart cards

Some folks might remember a time before the universal proliferation of credit cards. I am one of them! My mom carried some "coins" in her wallet when I was growing up in the '50s, dog tag style metal plates half the size of today's standard-size card, imprinted with account data tied to an individual merchant, such as J.L. Hudson's in Detroit or de Bijenkorf in Amsterdam. These were charge cards, which required the consumer to pay in full each month.

Around this time, many merchants began accepting charge cards from companies like American Express and Diner's Club. BankAmericaCard (now VISA) and Master Charge (now MasterCard) transformed the field with credit cards, which allowed consumers to carry a balance from month to month, an instant personal loan that was a huge hit with consumers worldwide.

With scale came great profit but also great opportunity ... for credit card fraud. Absent a way to validate a card at the time of a transaction, forgery is trivial. To combat fraud, US card issuers periodically distributed books containing page after page of lists of fraudulent and revoked card numbers. I remember these books being printed on the thinnest paper my young eyes had ever seen. This didn't work well, so US issuers induced merchants to install dial-up modems and check for card revocation on each transaction. This is a clumsy system that relies on consumers to report fraudulent use of their cards but is easy to deploy.

European vendors also sought a way to thwart wholesale card forgery, but in contrast to the deregulated dial-up market in the US, the European digital network infrastructure was governed by a collection of stodgy national ITU entities that stymied online card verification. Smart cards offered an offline alternative: a trustworthy mobile computing platform. Pin-protected smart cards were issued to consumers and card readers were issued to merchants. The card issuers controlled the cards and the readers and they were able to prevent a lot of fraud.

Chip cards quickly became a killer app for smart card proliferation in European pockets and purses. Across the Atlantic, though, there was no swell of smart card adoption. Issuers like VISA and American Express initiated pilot programs that provided cards and readers to consumers, but they didn't catch on, in part because consumers already had adequate fraud protection guarantees from issuers and had no incentive to make transactions more secure. Eventually, US issuers got on board and issued smart cards to consumers -- although without PIN protection -- and readers to merchants. It was oddly late in the game, though.

3 - Smart card R&D

Meanwhile, researchers in industry and academia were viewing with interest the potential of the smart card as a mobile trusted computing platform.

Boiling it all down, every digital identity needs a secure place to store some keys and do a little cryptography. Today we use smartphones for all that, but 25 years ago, the smart card's peculiar hardware and barren API offered interesting possibilities.

CITI was fortunate that one of its alumni was well situated at an industrial research lab, and in 1996 brought CITI in on some collaborative research. Bellcore cryptographers had cooked up some fast, scalable ciphers for videoconferencing that wanted a trusted computing base for key distribution, and it seemed like a smart card might fit the bill. At that moment, CITI needed more R&D $$$ pretty badly and proposed to take on the work. Remarkably, CITI succeeded in building the smart card-based key store. We paid some bills, learned a lot about smart cards, and got some nice papers out of the collaboration.

By coincidence, just as CITI began investigating smart card capabilities and developing smart card competencies, the University of Michigan was deploying a smart card-based ID card -- the MCard -- equipped with an electronic purse application. The University had a business agreement with a bank and a vending machine supplier: the bank installed charging stations all over campus so that students, faculty, and staff could fill their electronic purses with up to $50 and the vending machine supplier installed and serviced vending machines retrofitted with smart card modules all over campus. Somehow, they periodically settled accounts.

As with most projects like this, it probably never made or even saved anybody any money and the chip card based MCard is now a faded memory. But at the time, smart card reader-equipped vending machines were all over campus, even in off-campus offices like CITI, where we had a snack machine and a pop machine in our lunchroom.

This was an obvious target and a temptation. It didn't take us long to begin tracing the communication between the MCard and the vending machine.

Guess what we found. >:)

4 - The MCard electronic purse protocol

A smart card spends most of its life in a lonely powered-down state, crushed in a wallet or buried in a purse, but when inserted into a reader, the card springs to life, resets itself into a predetermined state, and awaits commands.

Smart card communication obeys a suite of standards, known as ISO 7816, that govern physical aspects of the card, such as shape and size, location of the contacts, the serial data interface, and describe an RPC-like protocol that allows the exchange of structured command/response pairs and challenge/response authenticators. The card stores long-term keys and other data in a hierarchical file system with idiosyncratic access controls.

We can imagine a bare-bones electronic purse protocol in which:

1 the vending machine checks that the purse has sufficient funds, then

2 the cardholder makes and receives a selection, then

3 the vending machine updates the purse.

This protocol is vulnerable to a simple attack, called tearing, in which the cardholder forcibly removes the card after step 2 (deliver the selection) but before step 3 (update the purse), and indeed, this protocol is not what we found when we traced the messages exchanged between a vending machine and an MCard. Instead, we found a two-phase protocol that resists tearing by reserving funds prior to product selection and debiting the funds and committing the transaction after product selection.

Message to MCard	MCard response
Wake up!	I'm awake
How much $$$ in the purse?	$18.23
Show me the last entry in the log	$18.23
Give me a nonce	Here is a nonce
Send nonce encrypted with shared key	Vending machine is authentic
Reserve $1.25 in the log	OK

Table: Pre-selection phase of the MCard electronic purse transaction.

The vending machine has reserved $1.25 on the MCard and awaits consumer product selection. Already, a flaw in the electronic purse protocol is evident. The card does not trust the vending machine and requires it to authenticate before writing to the log, but the vending machine never authenticates the card. This suggests an obvious replay attack.

To be clear, if we can program a computer to utter the MCard responses shown above, we can use an inverse card reader or a JavaCard to inject those messages into a vending machine. The unsuspecting (literally!) vending machine believes it is talking to an MCard with $18.23 in its purse. A bottomless purse, forever. (This feature is what got the attention of the Secret Service.)

Back to the transaction in progress. So far, the first phase of the protocol has reserved funds in a log. Following product selection, the second phase debits the purse and updates the log, authenticating itself in both updates. It's up to the next vending machine that reads the card to confirm that the log and the purse agree and to reconcile them if they don't.

Message to MCard	MCard response
Give me a nonce	Here is a nonce
Debit $1.10 from the purse, \ authenticated with encrypted nonce	OK
Another nonce-based authentication	OK
Log the $1.10 transaction	OK

Table: Post-selection phase of the MCard electronic purse transaction

To belabor the point about the replay attack, during a purchase, the Mcard authenticates the vending machine three times: twice prior to writing to the log and once during the debit operation. But at no time does the Mcard authenticate to the vending machine.

It would certainly be possible to engineer a challenge/response that runs in the opposite direction, indeed, the smart card has a specific API for this Arguably, authenticating the card is the very first thing the vending machine should do when a card is presented. So why didn't the smart card developer include this in their protocol? Or even simpler, reverse steps 2 and 3 in the bare-bones protocol described earlier.

We can only speculate. The system we observe protects against attacks on the card and leaves the vending machine wide open. This makes a certain amount of sense: to attack a card, you need only a PC and a card reader, but attacking a vending machine requires tools like CITI's bespoke card extender and inverse card reader.

So, if you don't believe hackers will invest a lot of time and attention to fraudulently purchase a bag of potato chips, you may deem it unnecessary to cryptographically authenticate the smart card. We have seen other instances where industry has a blind spot about the capabilities and interests of hackers (especially university-affiliated hackers).

"User experience" considerations may also play a role. Like most smart cards, the MCard featured an 8-bit microprocessor running at 3.57 MHz, communicating half-duplex through one contact at about 1 msec per byte.

Slow processing and communication produces authentication round-trip times longer than a second. If too many seconds elapse between card insertion and the vending machine signaling readiness, or if even a second goes by between selection of a product and the delivery of that product, the customer is going to be frustrated.

If round trips were fast, the protocol might look something like

1. Mutually authenticate
2. Check purse value
3. Customer makes selection
4. Update the purse
5. Deliver the selection
6. Eject card

The critical section lies between steps 3 and 5, updating the purse between the time the customer presses a button to make a selection and the start of product delivery. This is when the customer is most anxious about their purchase: if that little spiral holding the KitKat Bar doesn't begin to spin immediately, the user experience breaks down and havoc can ensue.

But even this protocol fails in the face of a determined hacker. Taking the inverse card reader to the next level, CITI developed a monkey-in-the-middle apparatus that filters messages from the vending machine, directing them either to a legitimate card or to a rogue application. This can defeat any protocol that doesn't cryptographically sign messages (at the very least).

5 - Responsible disclosure

We had good contacts on the card manufacturer's research team, and a productive phone call took place right away. Naturally, we shared our discovery with university officials as well; actually, that was the first phone call. The response was lukewarm thanks. But one official with whom we met surprised us by pulling a Visitor MCard out of a desk drawer and informed us that every time they inserted this card into a vending machine, five quarters dropped into the coin return!

We borrowed the card and examined it in our lab. The card has a subtle flaw: the key file for authenticating the vending machine is unreadable, so when the vending machine wants to reserve funds in the log, authentication fails and the card reports an error.

What happens next requires us to speculate. The smart card module is retrofitted to an OG vending machine stocked with items that cost less than $1.25. The module verifies that there is $1.25 in the purse, tells the vending machine to credit the customer with $1.25, and reserves $1.25 in the log. We don't know in which order the module takes these steps, but the order is important!

When the authentication step fails, the smart card module appears to throw up its hands, eject the card, and go offline. The vending machine is now holding the bag for $1.25, and the smart card module is offline. The vending machine can't credit the refund to the card, so it does what an OG vending machine knows how to do: cash refund. Five quarters drop into the coin return.

We were excited by this discovery! Together with the earlier vulnerability, we could strip a vending machine of all of its product and all of its cash (if we dared). We were especially amused by the sound of a vending machine eventually dry-heaving quarters it didn't have: <cough cough cough cough cough>. I wish we had recorded that.

Nonetheless, our sense at CITI was that these vulnerabilities were too shallow to merit a refereed publication, and to an extent was a distraction from our sponsored research, so we talked about it at a few conference rump sessions and that was the end of it.

But we also saw another possibility: funding ongoing smart card research. No, not like that! Having proved our skills to the smart card vendor, we were encouraged to submit a modest proposal (to tunnel IP and ICMP through ISO 7816), but it turned out that vendor higher-ups had bigger ideas and approached CITI with a much broader scope and an order of magnitude larger price tag in mind. CITI initiated the Program in Smartcard Technology in the fall of 1998 and celebrated: for the first time under my direction, CITI was on solid financial ground.

I know what you're thinking, but I don't think Schlumberger bought CITI's silence. We sincerely believed that our discoveries were below the LPU ("least publishable unit") threshold. And there was genuine excitement on both sides about the potential for collaboration. But I'm sure they were grateful that they weren't forced to reflash their worldwide installed base of vending machine smart card modules, and perhaps the grant was one way for them to show their gratitude.

Whatever, we spent it all ... on student stipends, tuition, staff salaries, conference organizing, publishing, traveling, etc.

One of our earliest objectives was a smart card-based Kerberos sign-on for our UNIX and Windows computers and we got pretty far with it: we found a card that supports DES and extended the Kerberos library to use that card to perform client-side cryptography and key management.

To support smart card authentication on Windows, we extended the Windows Graphical Identification and Authentication (GINA) to support the Pluggable Authentication Method (PAM) and developed a smart card PAM module.

We built a VFS layer for the smart card's ISO 7816 hierarchical file system, allowing smart card files and directories to be navigated and updated through a POSIX interface.

We implemented ISO 7816 IP tunnel endpoints on UNIX and JavaCard, implemented TCP and IP on JavaCard, and implemented a web server on top of that.

We designed and implemented a middleware layer that supports remote access to physically secure smart cards and extended Kerberos V5 and SSH to use that infrastructure for client-side key management and cryptography.

We built handsome hardware that allows a Palm Pilot to connect to a smart card and developed Palm Pilot software to use the smart card as a key store. We designed and implemented personal secure boot, an extension to GRUB that stores software component hashes on a smart card and uses the hashes to ensure component integrity at boot time. We open-sourced all the software that we developed under a very permissive license.

We also used the funds to support travel and to evangelize the smart card way. I joined IFIP WG 8.8 as Co-Vice Chair to help organize smart card conferences in Louvain-la-Neuve, Chicago, Bristol, San Jose, Toulouse, and Tarragona; ran smart card demos at CCC in a field outside Berlin; and taught a graduate seminar in smart cards at Michigan, flying in European experts as guest speakers. CITI also made a major showing at HAL 2001 (Me, Jim Rees, Charles Antonelli, Andy Adamson, Dros Adamson, Dug Song, Niels Provos, Olga Kornievskaia; some CITI sons and lovers tagged along, too) and organized a well-attended smart card hacking workshop there.

6 - Aftermath (and airing of grievances)

CITI's burst of smart card research lasted only a few years, for reasons, some of them the usual ones, like money drying up and students graduating.

We had good relationships with our industry partners in the Dallas suburbs, but the center of gravity for smart card R&D was in France -- which we loved because we were forced to travel to beautiful French cities for meetings -- but it was hard to make an impact in that crowd, which was excited about explosive growth in SIMs and not that interested in a smart card that communicated with TCP/IP and web protocols. (We honestly could not understand that.)

Also, none of us is fluent in French, alas. When we tried to get some attention by nominating ourselves for the CARTES '2000 Innovation Award, well, the nomination had to be in French. We submitted it in English. You can imagine how the French jury responded. (They didn't.) Adding insult to injury, when CARTES got around to recognizing an Internet smart card, the award went to a group that did its TCP and IP processing off-card, merely tunneling application layer payloads through ISO 7816. >:(But it wasn't just our big egos. Hardware limitations were a big reason.

CITI had a vision of the smart card as the trust anchor of a personal computing environment and built stepping stones toward that vision: a Kerberos single sign-on for UNIX and Windows you could carry in your wallet, a UNIX file system for secure storage of personal records, TCP/IP for reliable end-to-end communication, even a web server, for what it's worth! When we started, conventional wisdom held that all this -- on an eight-bit microprocessor running at 3.57 MHz with 200 bytes of RAM -- was impossible.

At that point, it wasn't clear what to do next to achieve our vision: we needed more cycles, more storage, more bandwidth. So we waited for smart cards to get better but unlike every other computer technology smart cards got no faster, no more capable. They never did, really.

Nonetheless, smart cards did succeed -- wildly, and essentially contemporaneously: mobile phone technologies exploded in the new millennium, with smart card SIMs the mechanism for low-level mobile phone identification and authentication. And it turns out that is all that a smart card really needs to do, with high-performance trust management now built directly into smartphone silicon. Soon, eSIMs and NFC payments will obsolete smart cards altogether.

Dodging the Feds So back to that tense meeting with the Secret Service. I told them all we had learned about the MCard and its vulnerabilities. Our guests were businesslike and intelligent, and the briefing was over in an hour.

Then it was time for a lab tour, the high point of which was the lunchroom. As we approached the cowering, trembling vending machines, I asked the head fed if he would care for a demo. His enthusiasm was unvarnished: he would love a demo! I pulled a white card out of my pocket and offered it with outstretched arm.

He paused. Smiled. No demonstration would be necessary. He left us with some Secret Service baseball caps and t-shirts and we all waved goodbye.

[...]