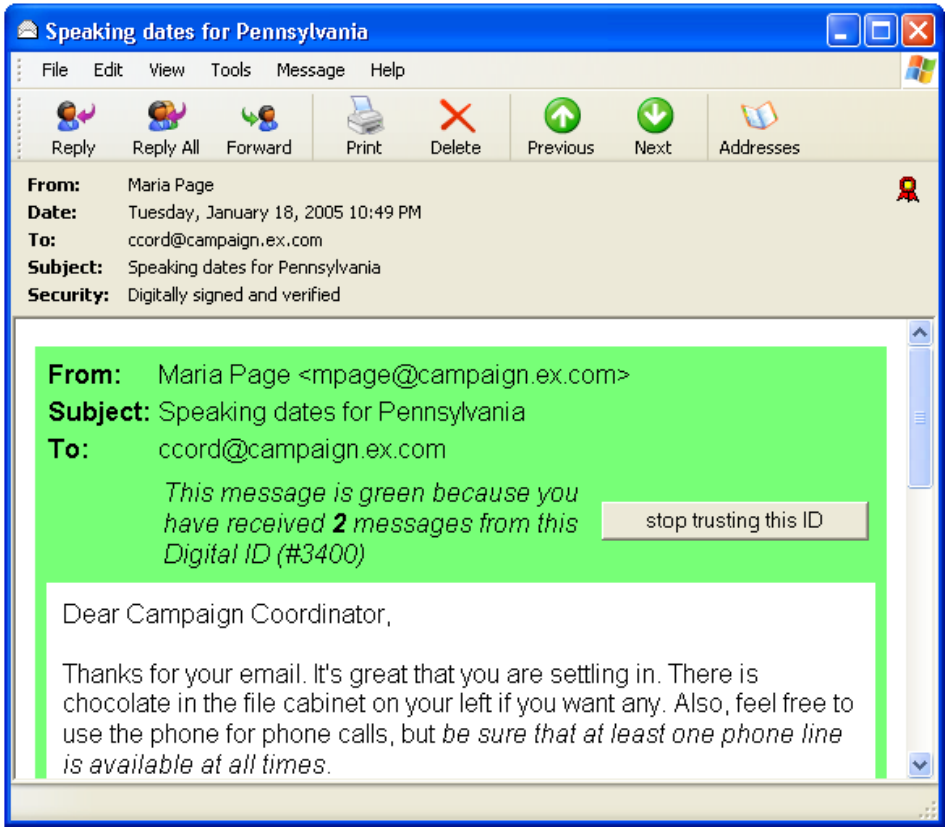
Aligning Security and Usability with Key Continuity Management



Simson L. Garfinkel

Harvard Center for Research on Computation and Society

http://www.simson.net/

I've been doing Security & Usability.

There seem to be two main ways that this work has proceeded:

- ✘ Work on authentication (hard problem).
- ✘ Work on new interfaces.
- ✘ Work on underlying rules and principles.

I'm taking a different track for aligning security and usability:

- ✔ Re-evaluating underlying models and mechanisms so that we can get more security with our existing interfaces.
  - File Sanitization
  - Secure Messaging
- ✔ Finding the best ideas and trying to put them all in one place.
- ✔ Working with vendors like Microsoft and Apple to incorporate these ideas into their products.

First project: Hard drive deletion

I purchased 250+ hard drives on the secondary market between 1998 and 2002.

- Many of the disks contained confidential information [Garfinkel & Shelat, IEEE S&P '03]
- Most of the confidential information could only be recovered using forensic tools.
- Hypothesis: people had been trying to erase the info, but their tools were not very good.

The trace-back study revealed that many cases involved the failure of a trusted organization or individual to correctly sanitize the drive before disposing of it.

Second Project: Email security survey

- Survey of 470 Amazon.com merchants in US and UK.
- 20% had been receiving S/MIME-signed messages for a year.
- Majority (72%) thought that receipts sent *from merchants* should be digitally signed, sealed with encryption or both

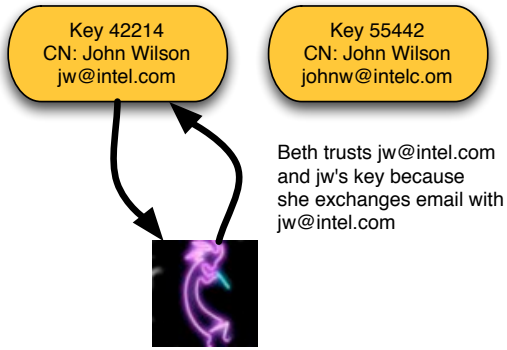
— Garfinkel *et. al*, FC2005 and CHI2005

Third Project: Enabling Email security through opportunistic encryption and Key Continuity Management

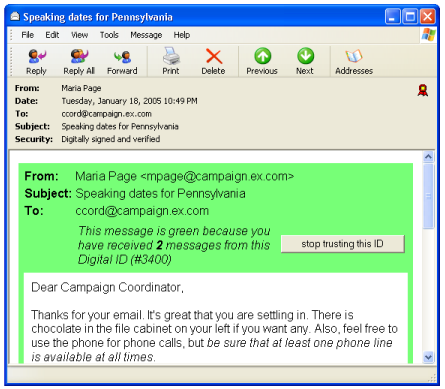
- Stream — an opportunistic PGP proxy
- CoPilot — a design of Stream for S/MIME
- Johnny 2 — a user test of CoPilot

This presentation focuses on the use and promise of Key Continuity Management

1. Why KCM can help solve the secure messaging problem



2. CoPilot: Implementing KCM with S/MIME and Outlook Express



3. Johnny 2: A user test of KCM

Secure Messaging — email that is *signed* and *sealed* — seems to be the grand challenge of usability and security.

- Public key cryptography was developed for secure messaging.
- This project is nearly thirty years old:
  - ➔ 1976 — Diffie Helman
  - ➔ 1977 — RSA
  - ➔ 1987 — RFC 989 (PEM)
  - ➔ 1991 — PGP Released
  - ➔ 1996 — S/MIME
- Today we use public key cryptography for SSH, SSL, and code signing — but there's virtually no secure email.

Either it's really hard to get this right, or nobody really cares.

People do care about email security. (Garfinkel et al, FC05)

In our study of Amazon.com merchants:

- 59% thought that email receipts from online merchants should be digitally signed.
- 47% thought receipts should be sealed

And they have the tools — sort of.

- 45% respondents would upgrade their email client to get more security.
- 54% of those using S/MIME-capable mail clients didn't know that they could receive digitally signed mail!

We should also want email security, because security could help with the largest security threats we face today:

- Phishing
- Spam

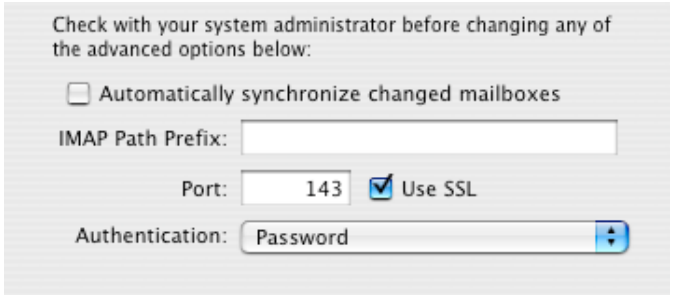
These threats can be addressed with signing alone.

So why is nobody sending signed mail? Why don't *businesses* like VeriSign send signed mail???

Gutmann argues that the emphasis on certification has been a distraction. [Gutmann 05]

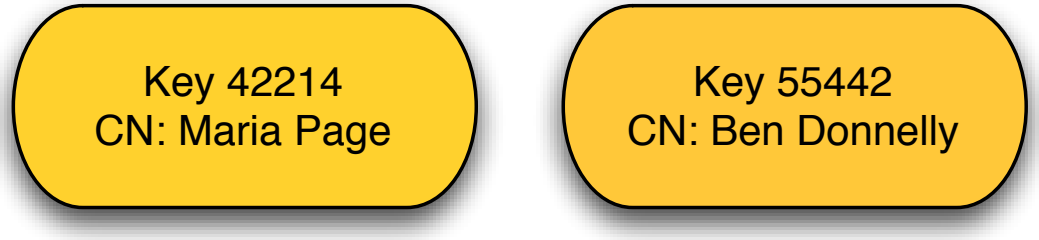
STARTTLS in SMTP, POP and IMAP has secured far more email than S/MIME or PGP.

Most of these certificates appear to be self-signed.



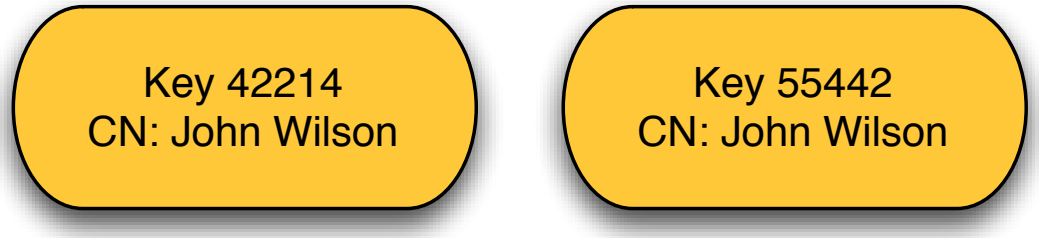
Perhaps the problem is that the CA approach is fundamentally not very usable.

Recall that a certificate is a statement signed by a CA that binds a key to a particular Common Name (CN):



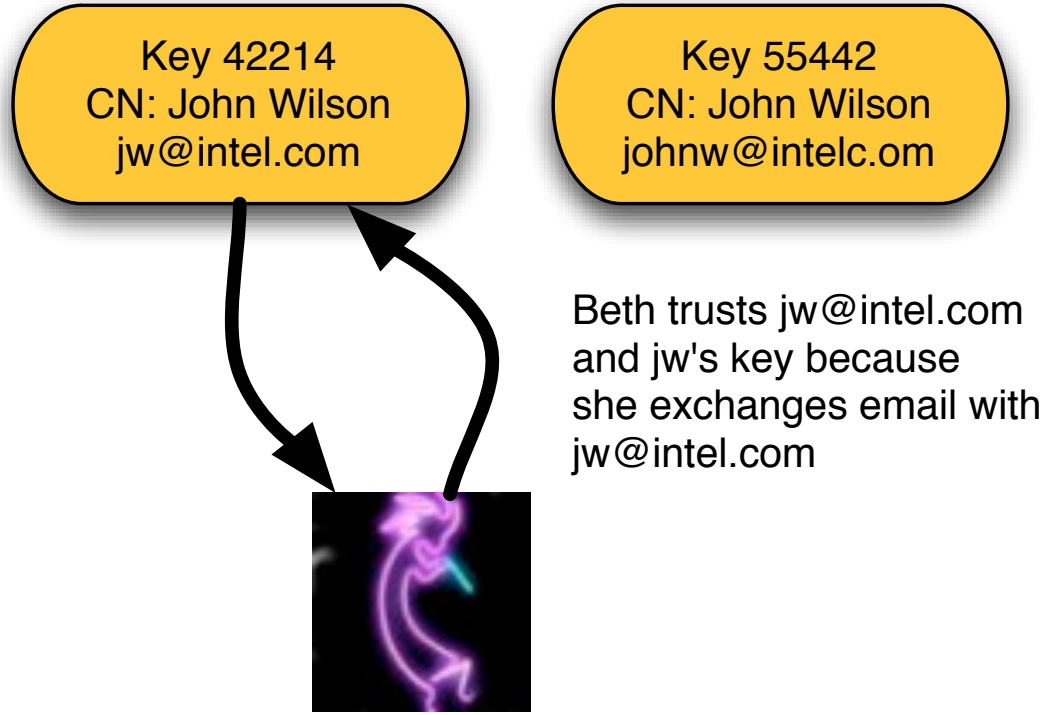
The theory is that humans understand names, not public keys.

Ellison argues that certified names are useless because names are not unique, not even within a company. [Ellison 02]



Certification has proven to be the hard problem that is gating secure email.

An alternative is to directly certify relationships:



We rarely want to send confidential information on the first try. We first verify that the person can receive it, that they are reading their mail, etc.

PGP avoided this problem

Phil Zimmerman handed me PGP 2.0 on a floppy with his key.

At that same party, somebody else gave me their key's fingerprint on a business card.

Today if you want to email somebody, you get their PGP key off their web page — or ask them to email you their PGP key.

The Stream SMTP and POP transparent proxy was a kind of automatic PGP assistant. [Garfinkel DGo'03]

Stream:

- ✔ Made PGP keys on the fly when it detected new From: address;
- ✔ Hid user's PGP key in the outgoing email header.
- ✔ Automatically incorporated keys that were discovered.

Planned but never implemented:

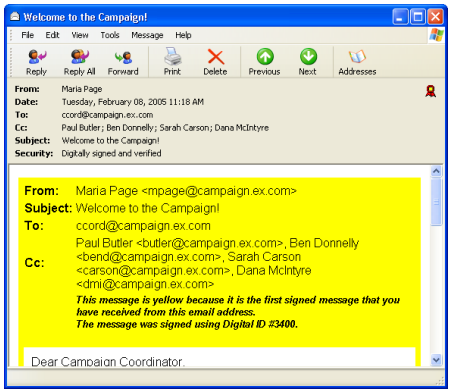
- ➔ Alert user if a correspondent's key changes.
- ➔ Automatically distribute and back up private keys.

The real problem with Stream was that PGP has poor penetration and poor usability.

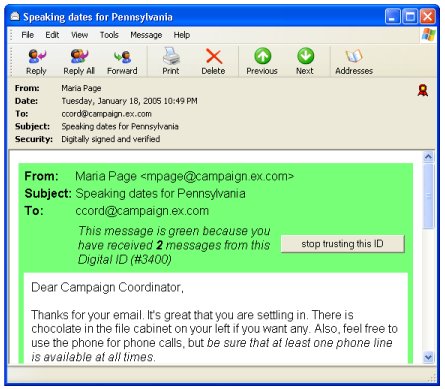
Most of Stream's goals can be achieved with S/MIME, by changing the certification model.

- S/MIME distributes certificates by sending them with signed mail.
- You need a transparent, zero-click way to make new certificates:
  - Option 1: create self-signed certificates.
  - Option 2: Some sort of automatic email answer-back system.
- You need an expert that watches the certificates used for signing and alerts on new `(cert, addr)` combinations.

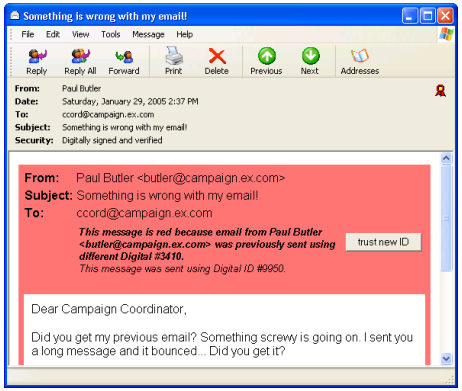
CoPilot Implements the Key Continuation Management interface on top of Outlook Express.



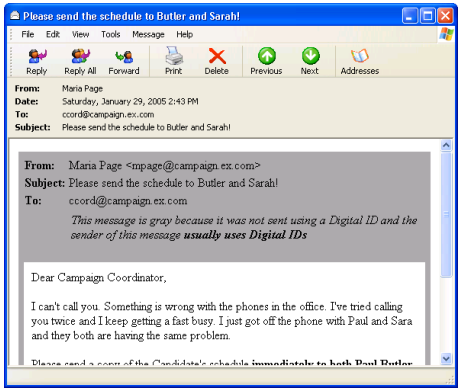
New Key



Same Key

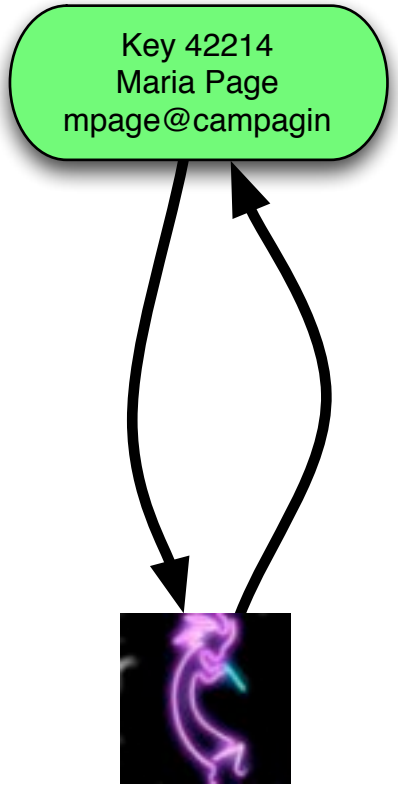


Changed key



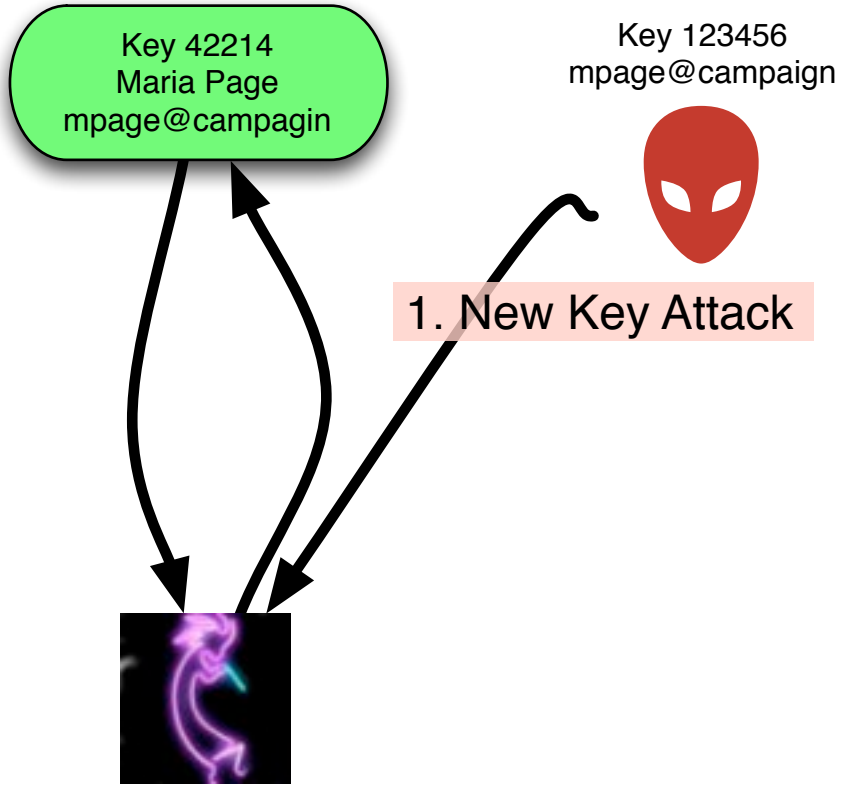
No Key

Unfortunately, this interface creates a number of attacks



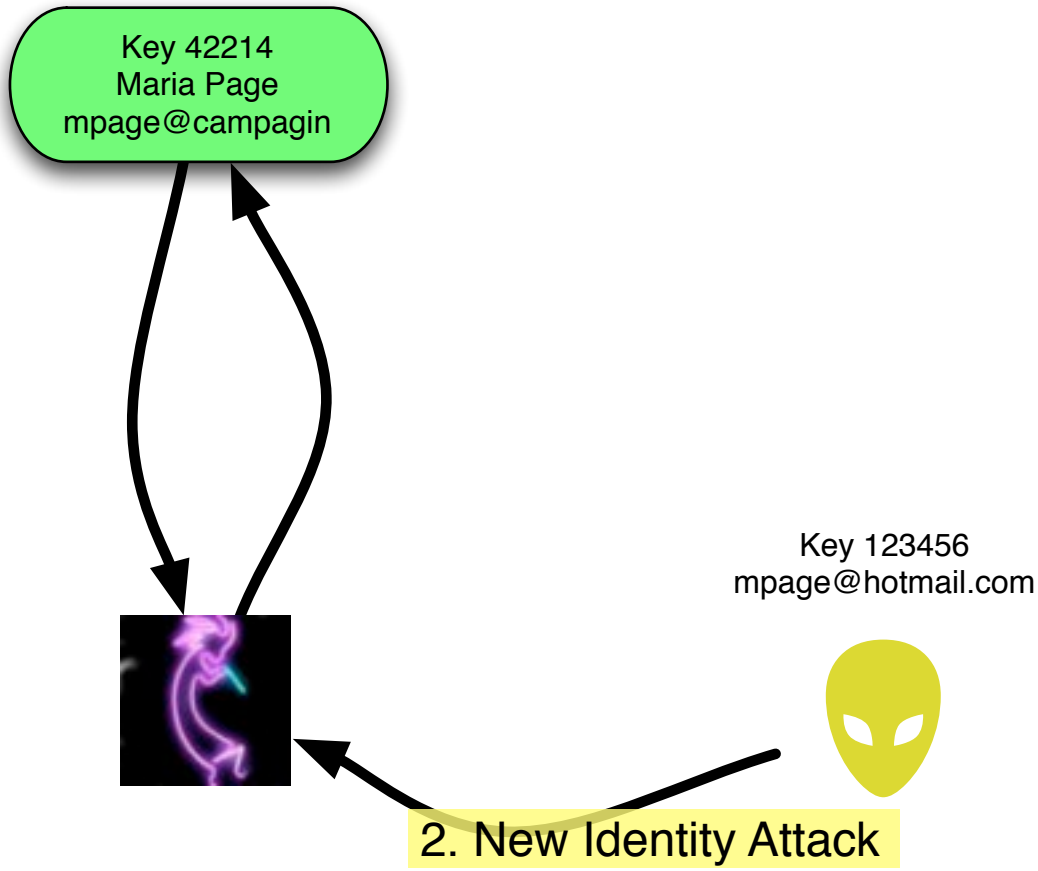
Normal Communications

Unfortunately, this interface creates a number of attacks



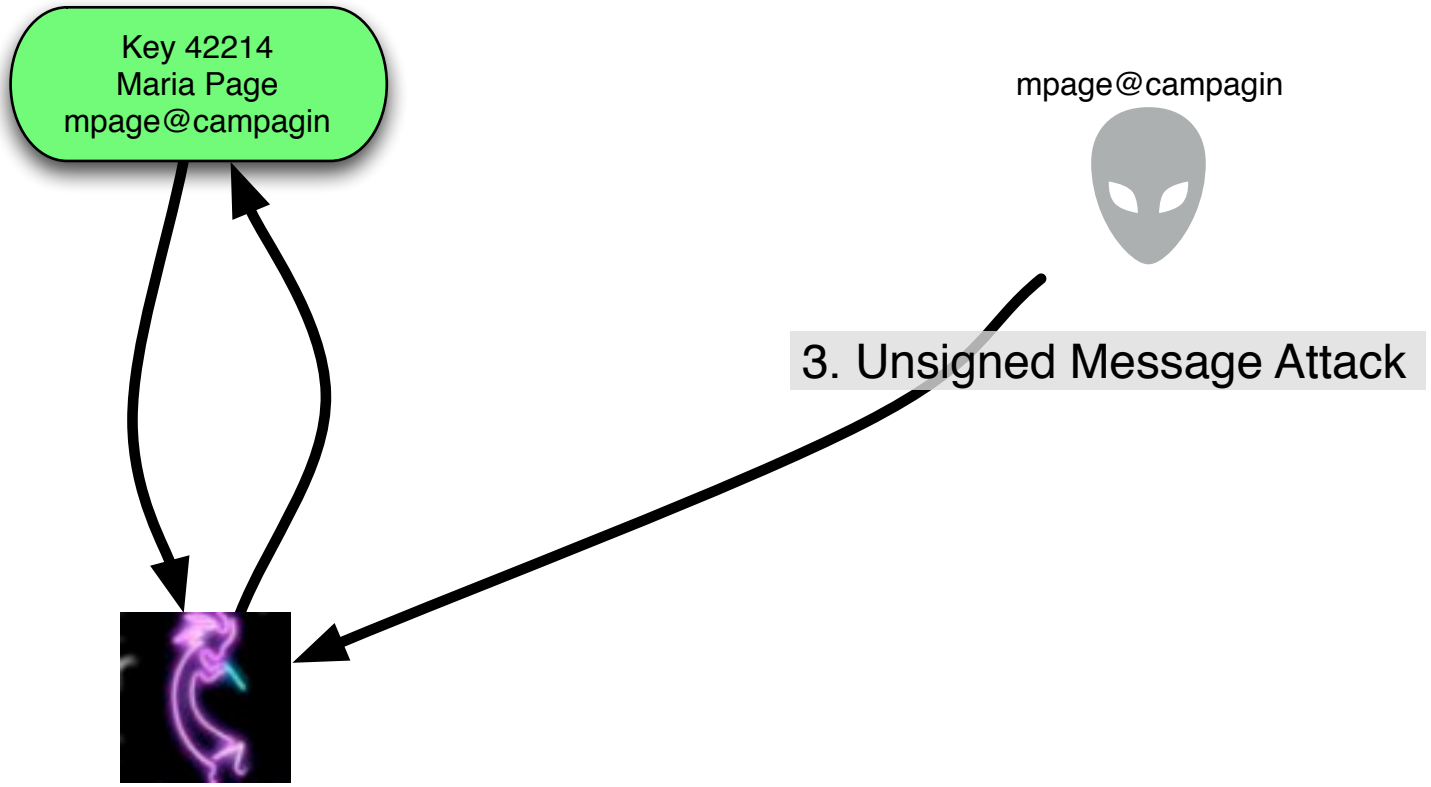
New Key Attack: (Forged From:, New Cert)

Unfortunately, this interface creates a number of attacks



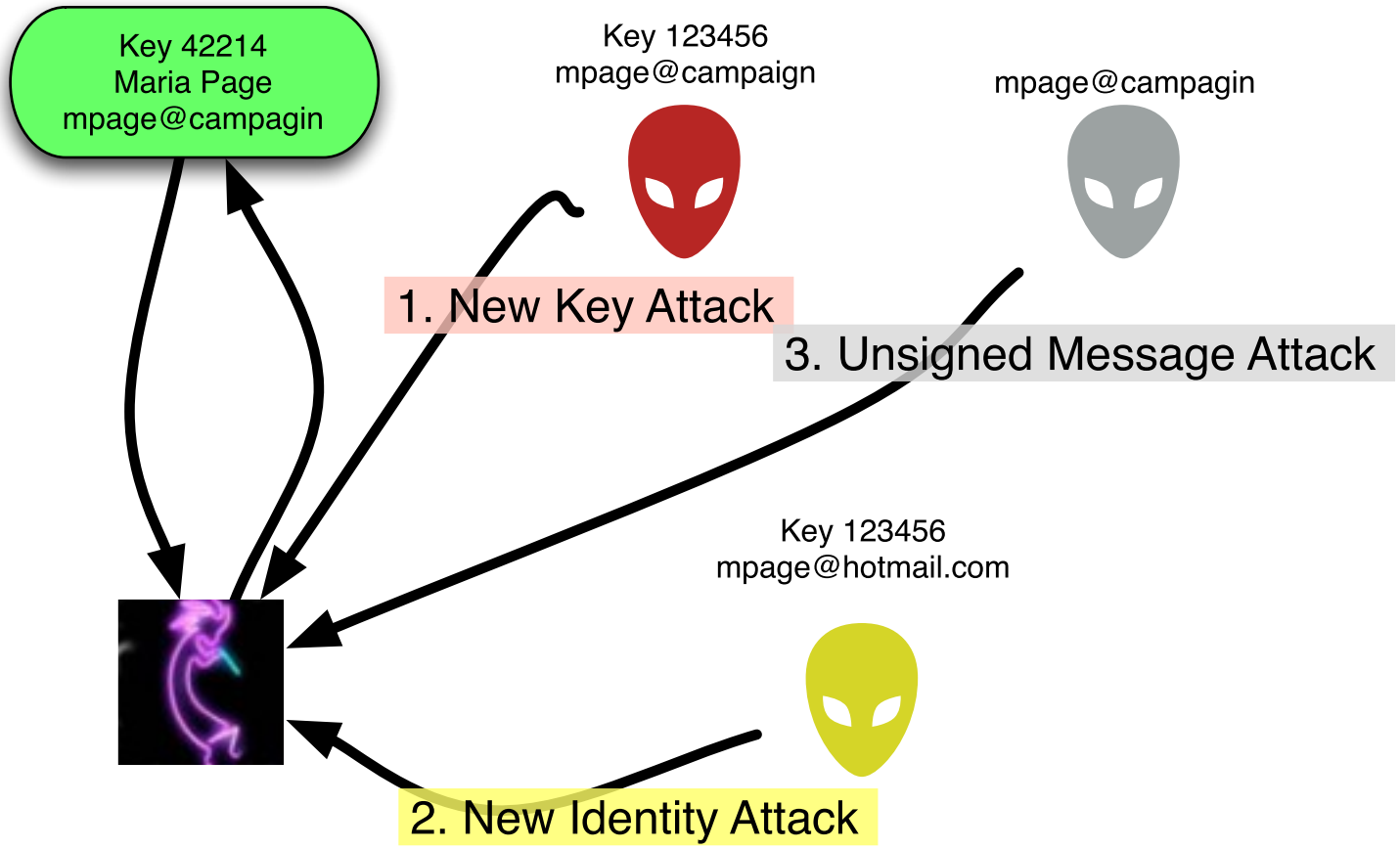
New Identity Attack (From Hotmail, New Cert)

Unfortunately, this interface creates a number of attacks



Unsigned Message Attack (Forged From:, No Cert)

Unfortunately, this interface creates a number of attacks



Can untrained end-users resist these attacks?

The original plan: Test KCM with Whitten's "Why Johnny Can't Encrypt" protocol.

- Scenario: A campaign volunteer sending out the schedule to other campaign workers.
- Whitten's test subjects rapidly suspended their disbelief.
- Use *Johnny* as our control group: see if KCM has a higher success rate and lower spoof rate than PGP.

Why Johnny Can't Encrypt:
A Usability Evaluation of PGP 5.0

Alma Whitten
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
alma@cs.cmu.edu

J. D. Tygar[1]
EECS and SIMS
University of California
Berkeley, CA 94720
tygar@cs.berkeley.edu

Abstract

User errors cause or contribute to most computer security failures, yet user interfaces for security still tend to be clumsy, confusing, or near-nonexistent. Is this simply due to a failure to apply standard user interface design techniques to security? We argue that, on the contrary, effective security requires a different usability standard, and that it will not be achieved through the user interface design techniques appropriate to other types of consumer software.

To test this hypothesis, we performed a case study of a security program which does have a good user interface by general standards: PGP 5.0. Our case study used a cognitive walkthrough analysis together with a laboratory user test to evaluate whether PGP 5.0 can be successfully used by cryptography novices to achieve effective electronic mail security. The analysis found a number of user interface design flaws that may contribute to security failures, and the user test demonstrated that when our test participants were given 90 minutes in which to sign and encrypt a message using PGP 5.0, the majority of them were unable to do so successfully.

We conclude that PGP 5.0 is not usable enough to provide effective security for most computer users, despite its attractive graphical user interface, supporting our hypothesis that user interface design for effective security remains an open problem. We close with a brief description of our continuing work on the development and application of user interface design principles and techniques for security.

1 Introduction

Security mechanisms are only effective when used correctly. Strong cryptography, provably correct protocols, and bug-free code will not provide security if the people who use the software forget to click on the encrypt button when they need privacy, give up on a communication protocol because they are too confused about which cryptographic keys they need to use, or accidentally configure their access control mechanisms to make their private data world-readable. Problems such as these are already quite serious: at least one researcher [2] has claimed that configuration errors are the probable cause of more than 90% of all computer security failures. Since average citizens are now increasingly encouraged to make use of networked computers for private transactions, the need to make security manageable for even untrained users has become critical [4, 9].

This is inescapably a user interface design problem. Legal remedies, increased automation, and user training provide only limited solutions. Individual users may not have the resources to pursue an attacker legally, and may not even realize that an attack took place. Automation may work for securing a communications channel, but not for setting access control policy when a user wants to share some files and not others. Employees can be required to attend training sessions, but home computer users cannot.

Why, then, is there such a lack of good user interface design for security? Are existing general user interface design principles adequate for security? To answer these questions, we must first understand what kind of usability security requires in order to be

[1] Also at Computer Science Department, Carnegie Mellon University (on leave).

The idea of comparing results directly with *Johnny* didn't quite work out.

- *Johnny* didn't have an attacker
- *Johnny* didn't use third-party certification
  — it used email answerback certification.

  (Therefore, Johnny only protected against a passive attacker.)
- Whitten said that the results were qualitative.
- Details of the original protocol had been lost.

The Johnny 2 Scenario:

It's Whitten's Scenario, except:

- The subject plays the role of a campaign volunteer.
- Three developed personas:

  Maria The Campaign Manager.
  Sarah Graphic artist.
  Paul Campaign treasurer
  Ben IT Coordinator.
- The enemy campaign tries to steal documents through a spoofing attack.
- The attacker pretends to be Paul, Sarah, and Maria in a series of escalating attacks.

Big Question to answer:

What will the users do when faced with the attacks?

New Key Attack	Attacker Paul says that he is having computer problems (new key, old email address).
New Identity Attack	Attacker Sarah says she is working from home and using Hotmail.
Unsigned Message Attack	Attacker Maria sends mail from her Campaign account, but it's not signed.

Other questions that the *Johnny 2* scenario can answer:

- Do users understand the difference between *signing* and *sealing*?
- If users can trivially sign/seal their email, will they?
- If users can seal confidential information before they send it, will they be less concerned about the destination?

The big question we don't need to answer:

Is it just as secure as CA model?

This isn't a fair question. . .

. . . KCM doesn't replace the CA, it replaces no crypto at all!

. . . If you have a CA, you can still use the CA model.

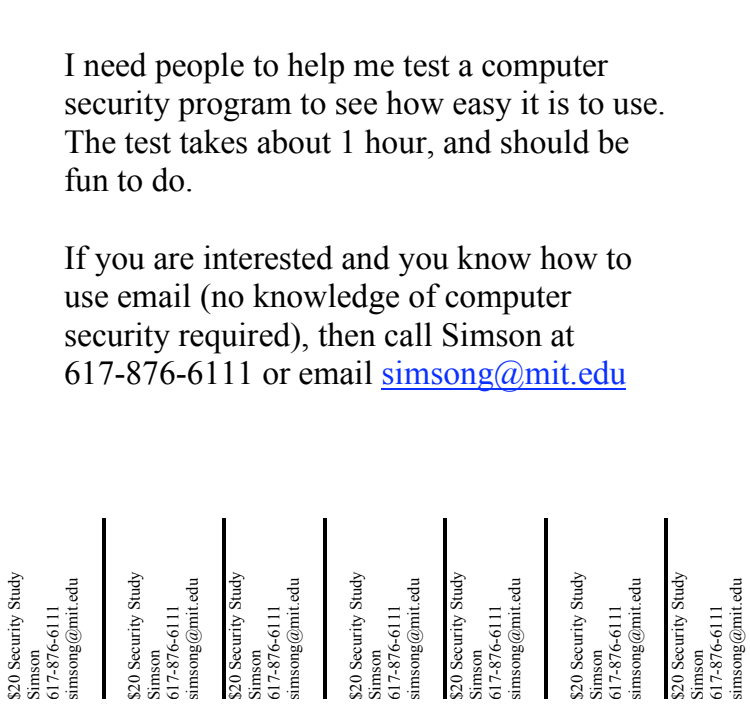
Johnny 2 User Study

Recruiting posters at MIT.

43 subjects aged 18–63
($\overline{x} = 33, \sigma = 14.2$)

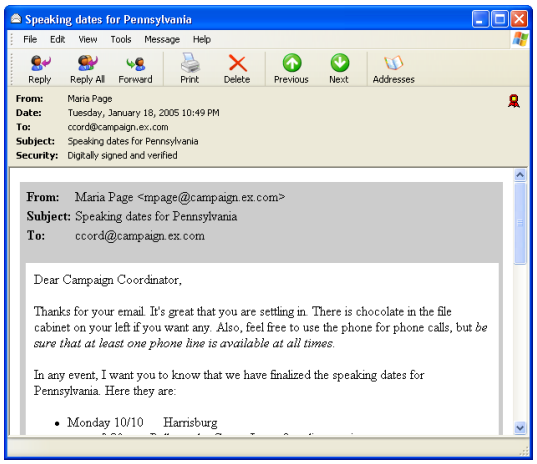
19 Men, 24 Women

17 to 57 minutes
($\overline{t} = 41, \sigma = 10.32$)

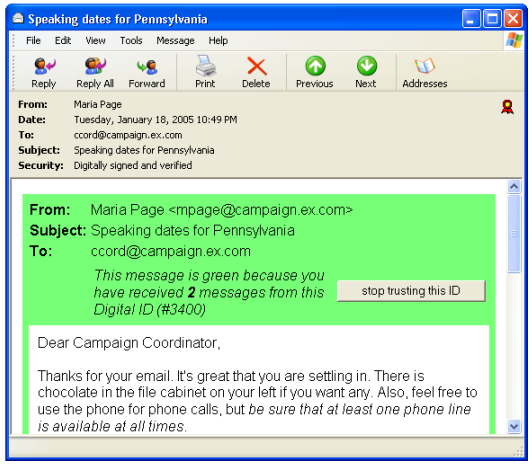


Three Cohorts compared.

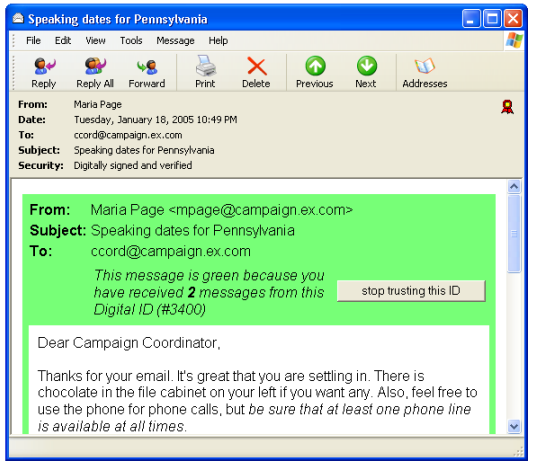
No Color



Color

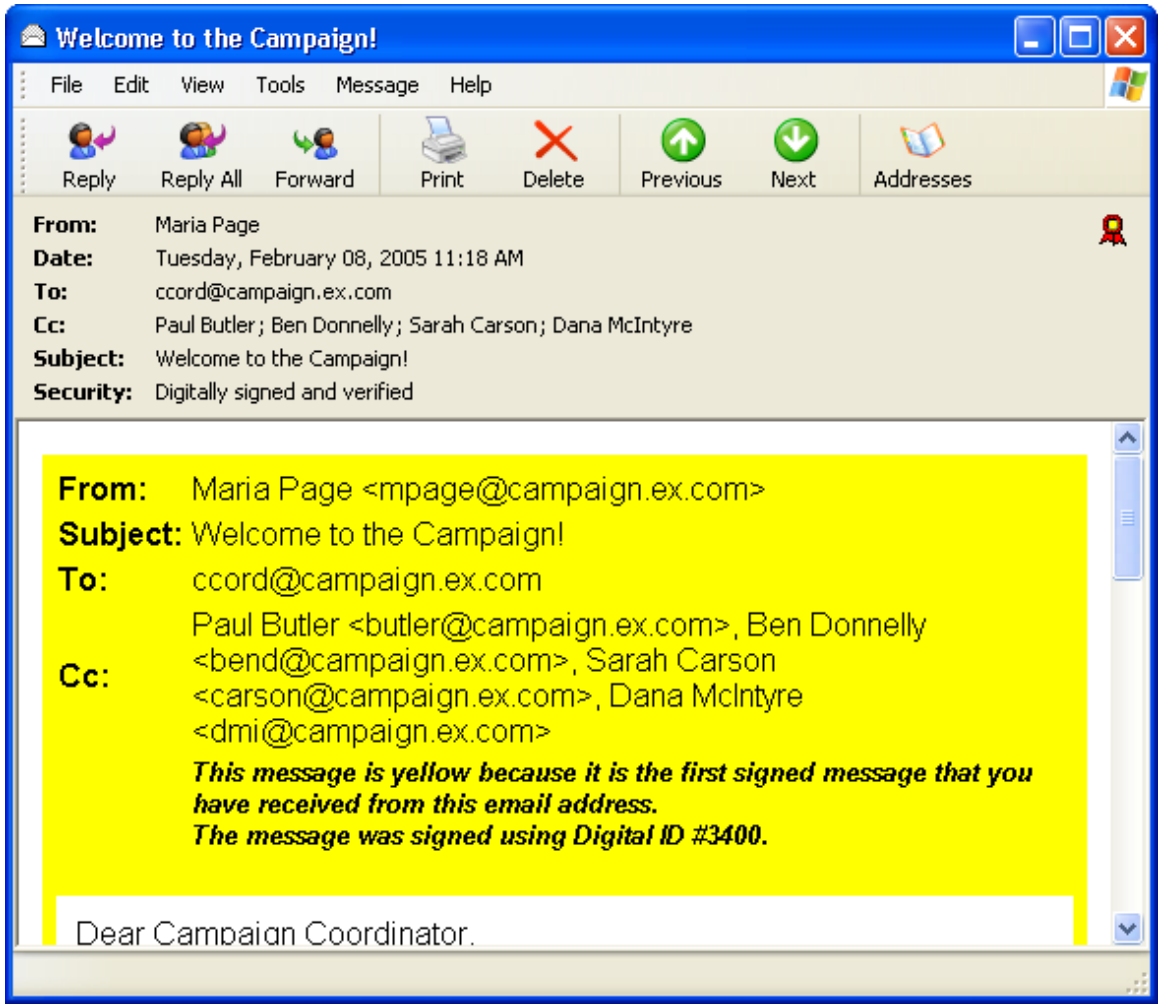


Color + Briefing



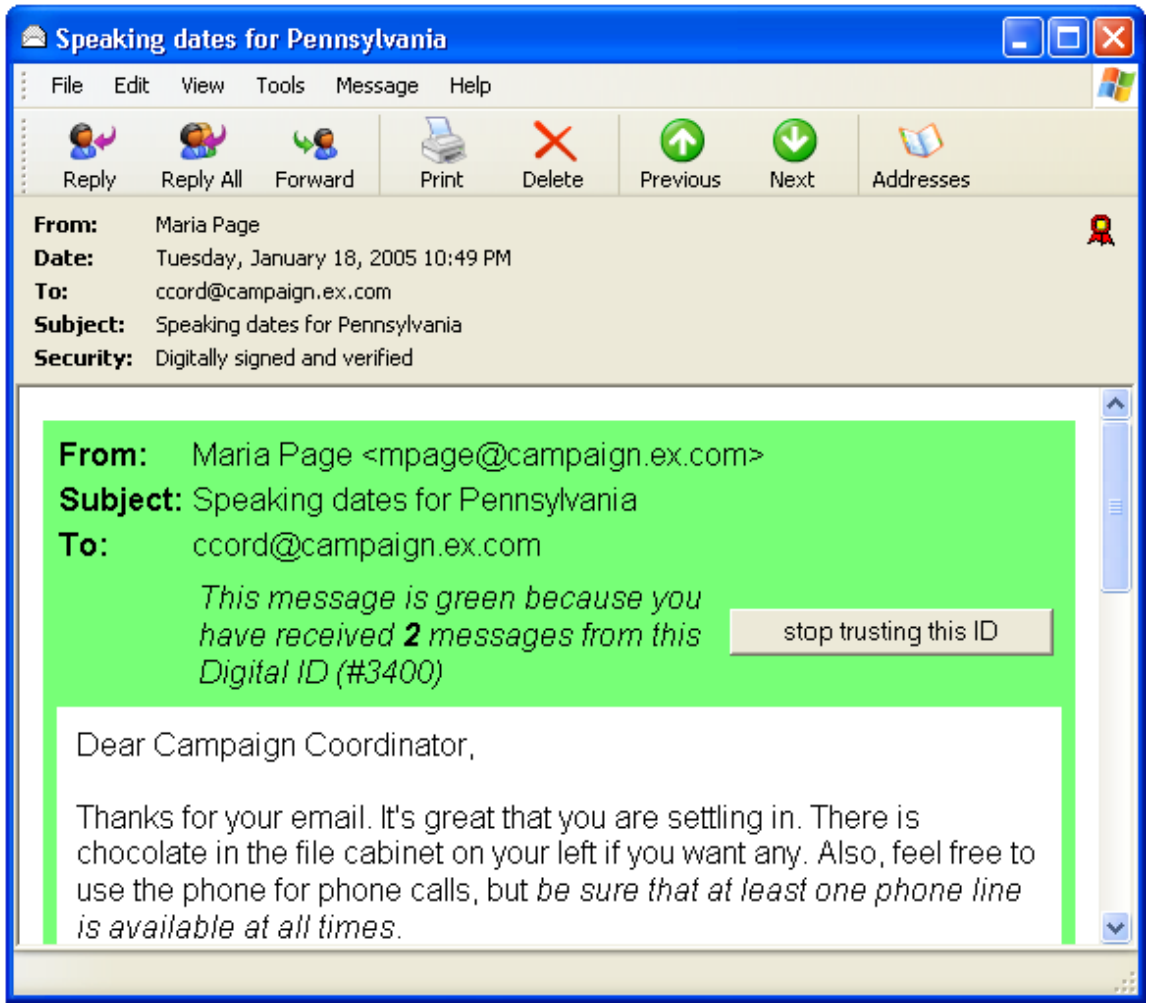
A Green Border will appear around an email message each successive time that a particular Digital ID is used with an email address.

Scenario Message 1:
Greetings from Maria Page



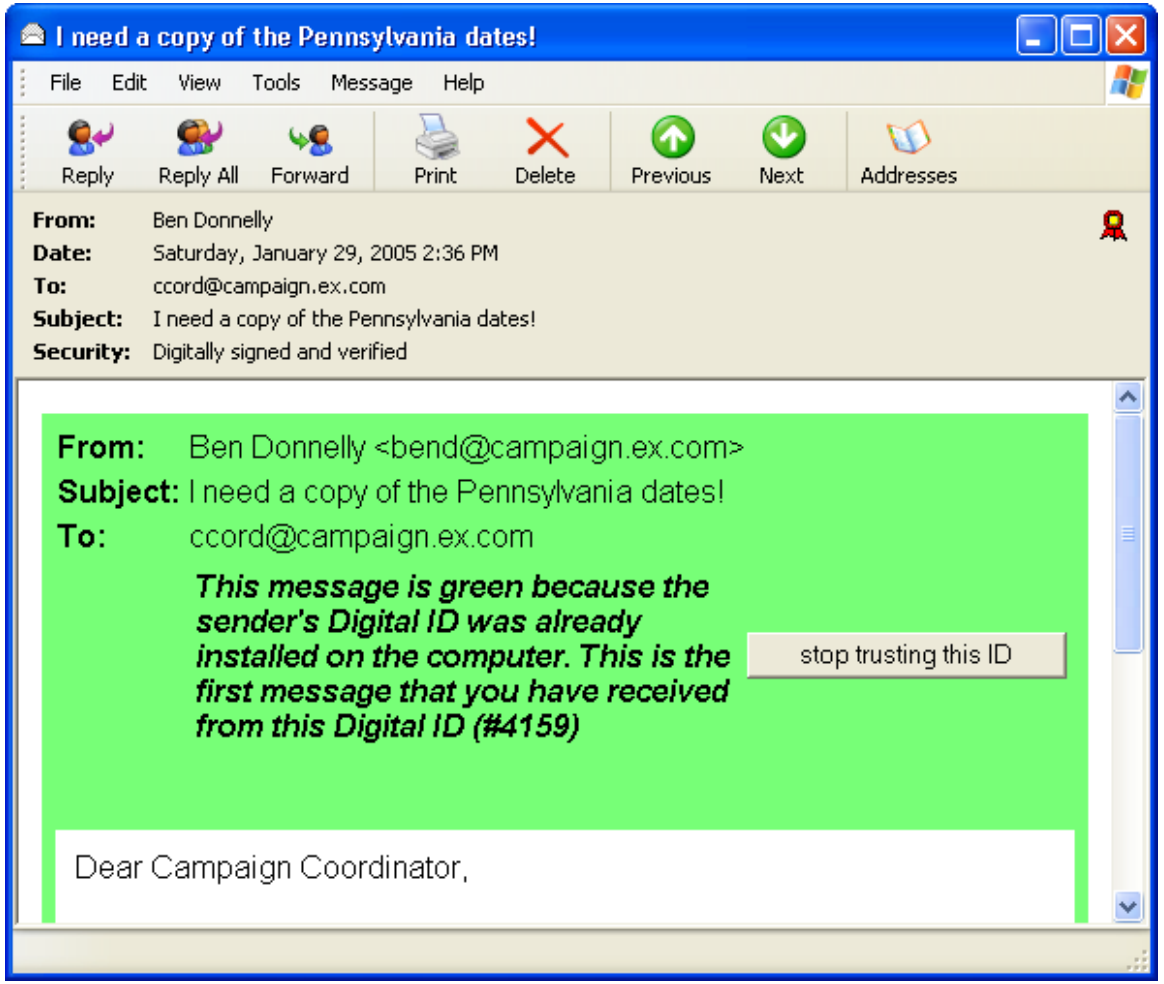
Orients user and provides list of campaign worker roles.

Scenario Message 2:
Maria sends the schedule



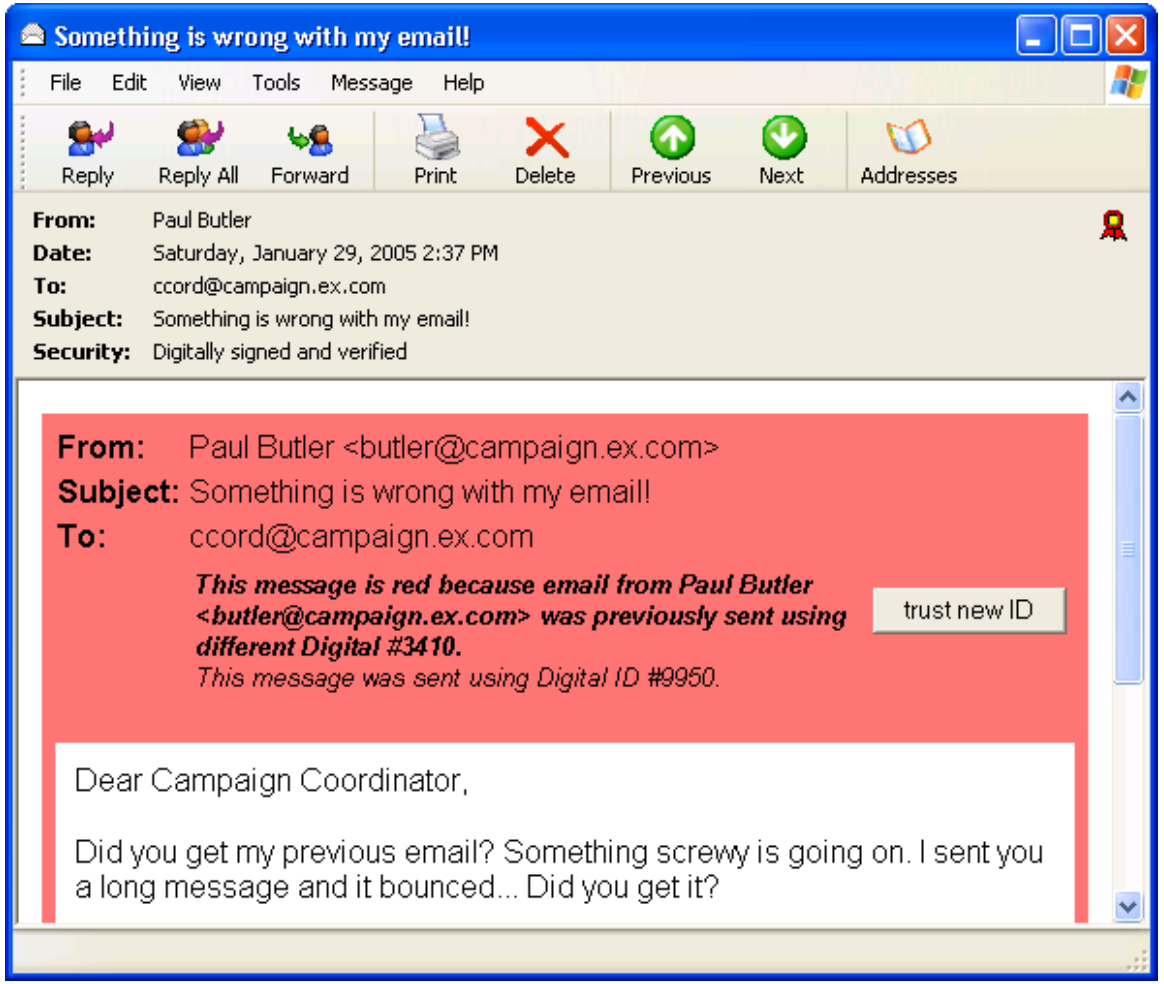
Tests to see if the subject can follow directions.

Scenario Message 3:
Ben asks for the schedule



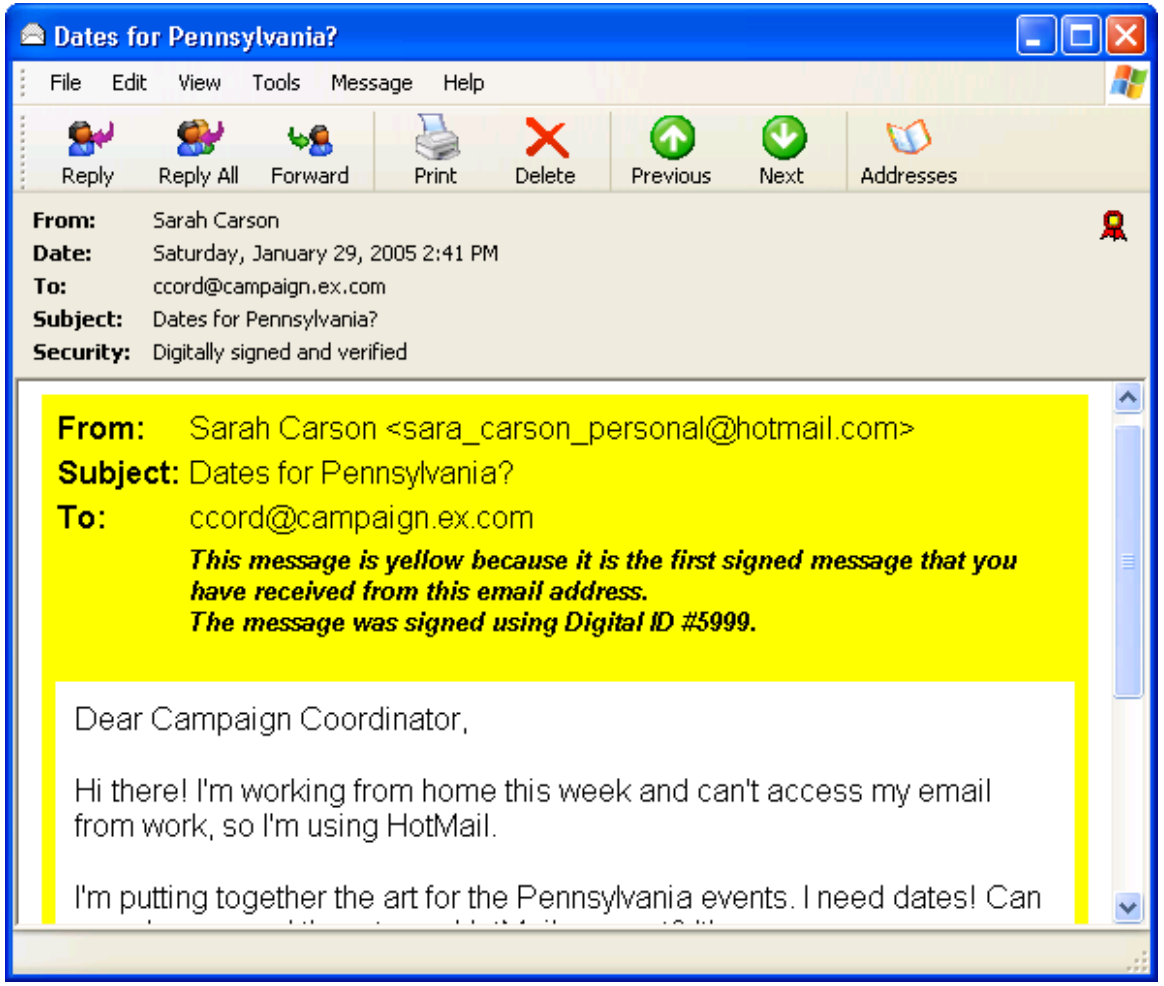
Will the subject trust a legitimately signed message?

Scenario Message 4:
Attacker Paul asks for schedule



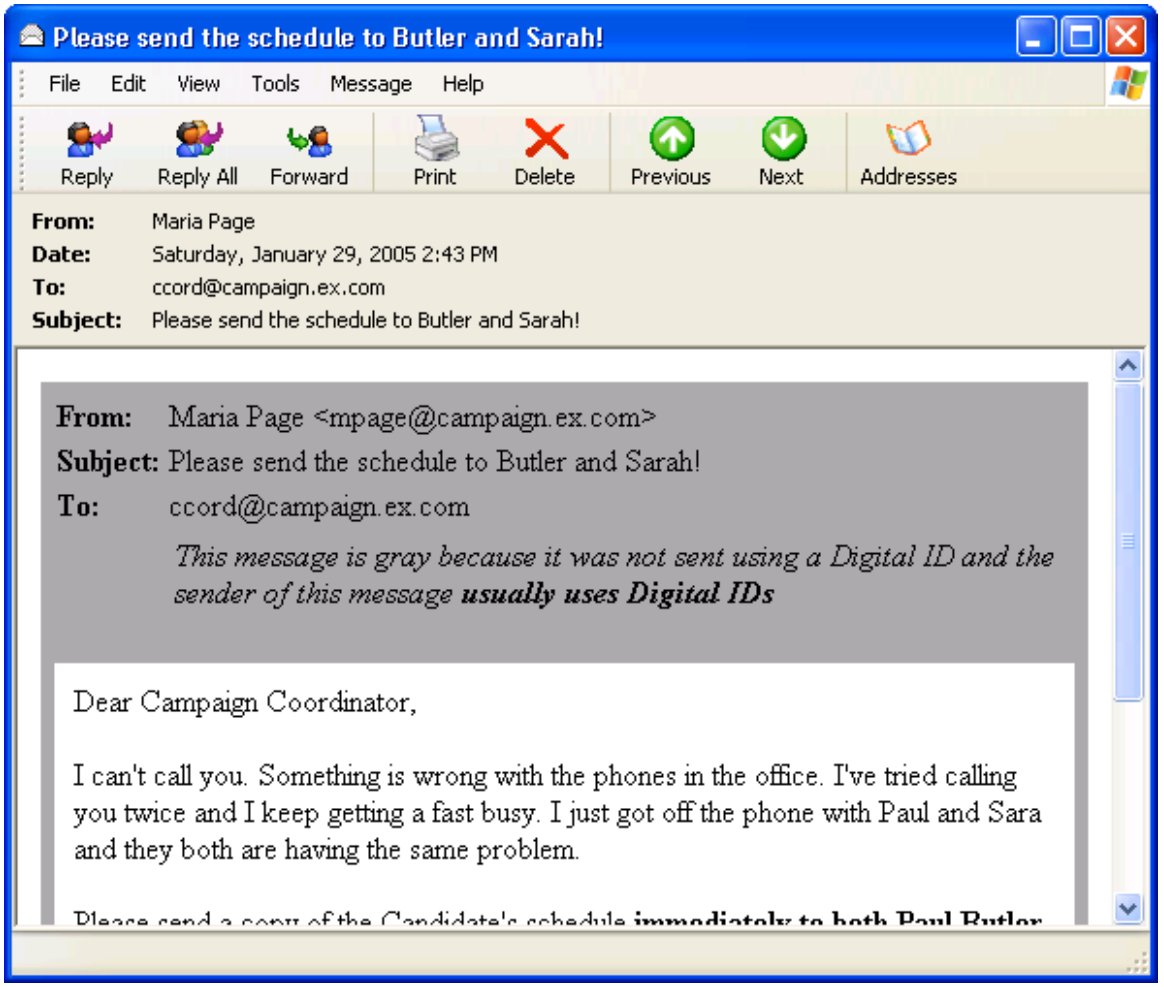
New Key Attack
(combined with a Reply-To: attack)

Scenario Message 5:
Attacker Sarah asks for schedule



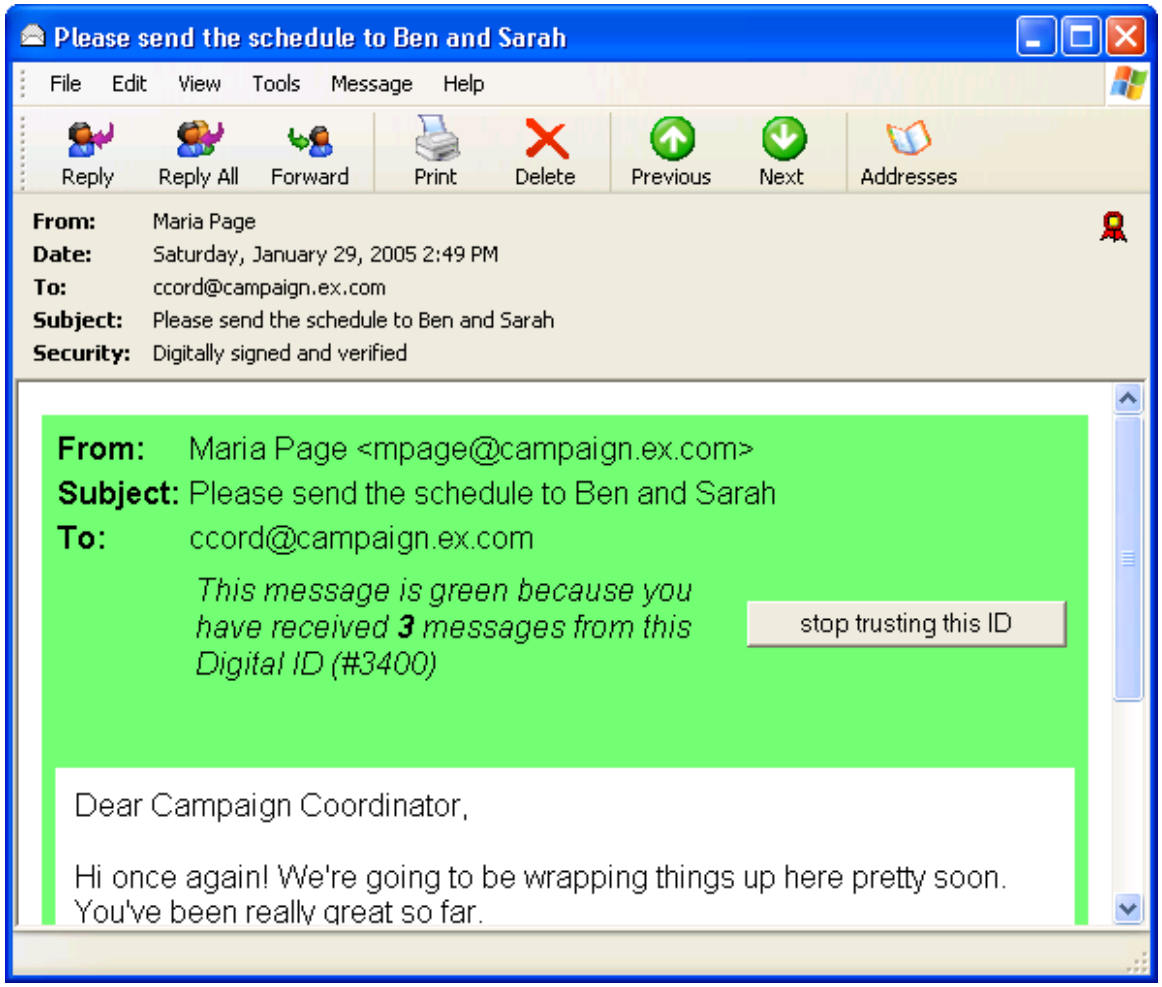
New identity attack

Scenario Message 6:
Attacker Maria demands that schedule be sent to attackers Paul and Sarah



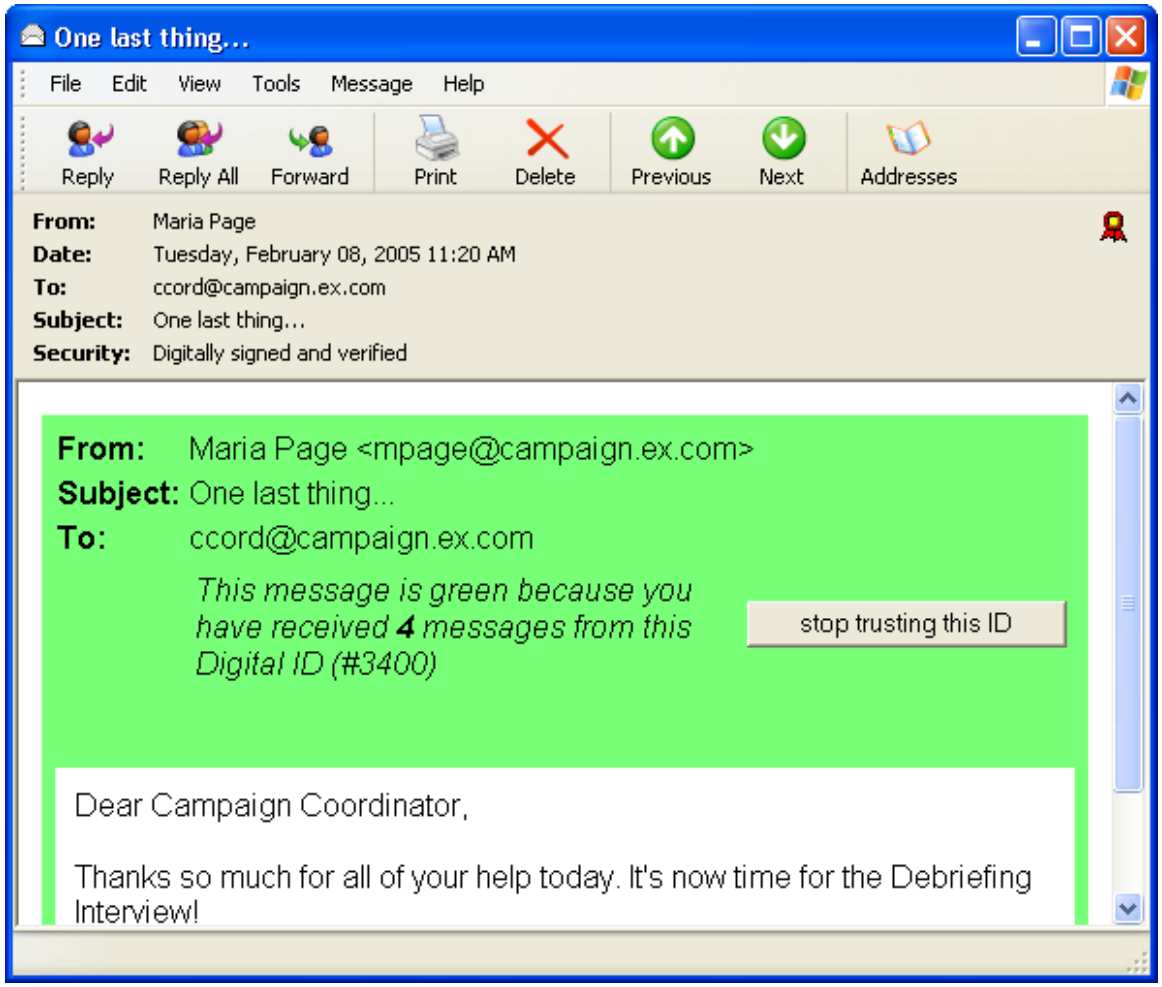
Unsigned message attack

Scenario Message 7:
Maria Page asks that schedule be sent to Sarah and Ben



Another test or “control” message

Scenario Message 8:
Maria Page thanks the subject



This proved to be a nice way to end the experiment.

Results, Task Comprehension:

Most subjects:

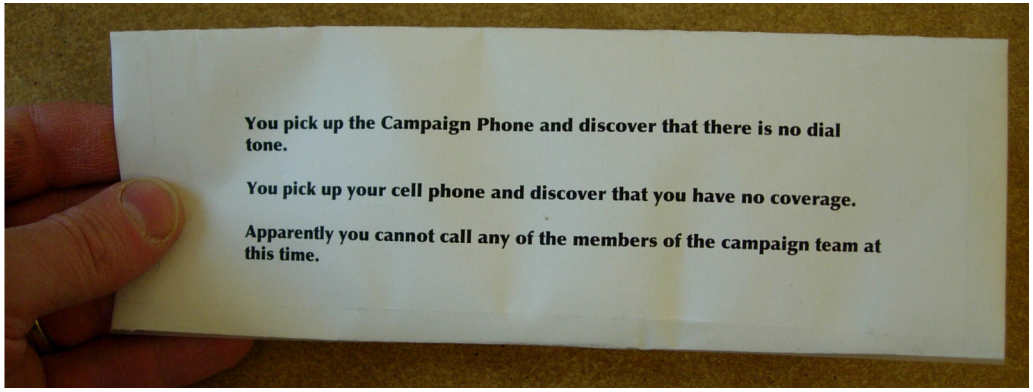
- Understood and enjoyed the scenario.
- Understood the concept of a “signed message” as authenticating the sender.
- Didn’t realize that signing prevented message modification

Many people who were attacked didn’t realize it at all; some realized it after-the-fact.

Many users struggled to authenticate the new identity and unsigned messages.

- A few people looked at the digital signature using Microsoft's certificate tools. They saw that the message was signed, but didn't know what it meant.
- Many users tried Email answer-back. A few mistook Attacker Maria's message for an answer to a message that was sent.
- Roughly half the users asked for the phone.

Well, we didn't let them use the phone

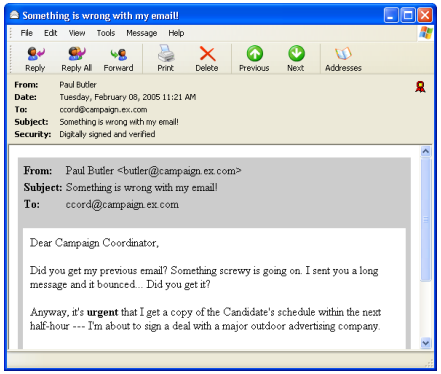


"You pick up the Campaign Phone and discover that there is no dial tone.
"You pick up your cell phone and discover that you have no coverage.

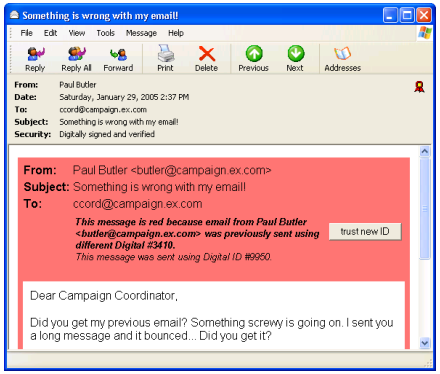
"Apparently you cannot call any of the members of the campaign team at this time."

KCM was very successful against the New Key Attack:

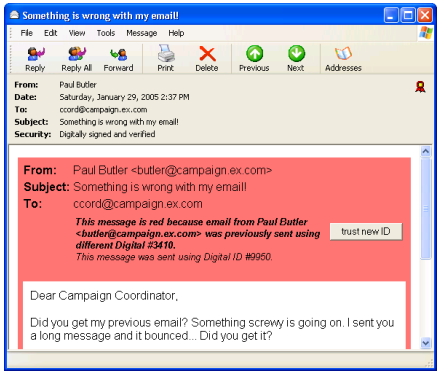
No Color



Color



Color + Briefing



A Red Border . . . someone else is trying to impersonate the sender.

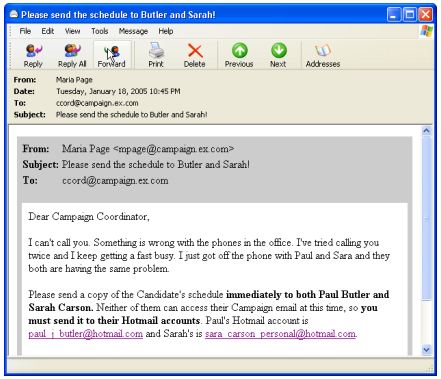
Rate of successful attack:

71% 64% 13%

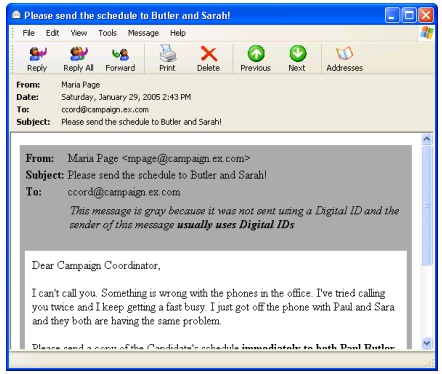
$p = 0.001$

KCM works well against the Unsigned Message Attack:

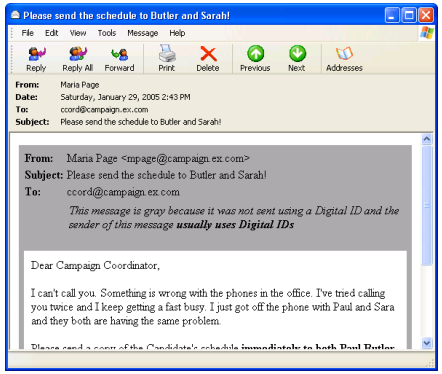
No Color



Color



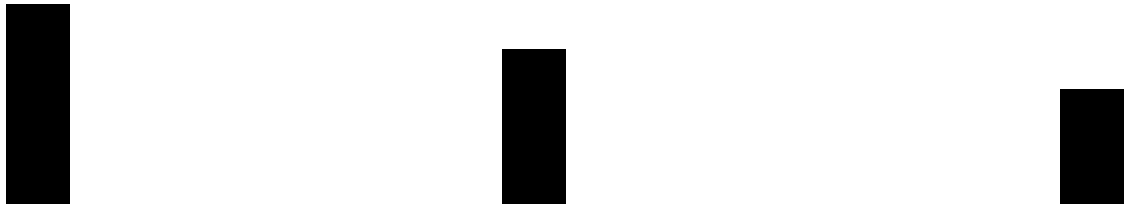
Color + Briefing



A Gray Border . . . someone else who is trying to impersonate the sender.

Rate of successful attack:

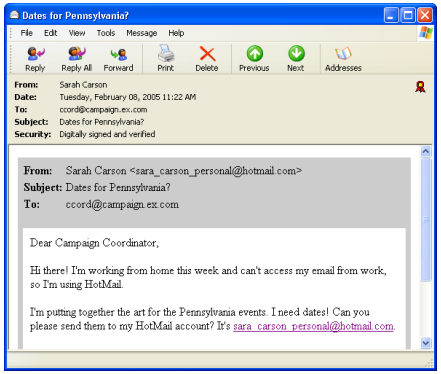
75% 58% 43%



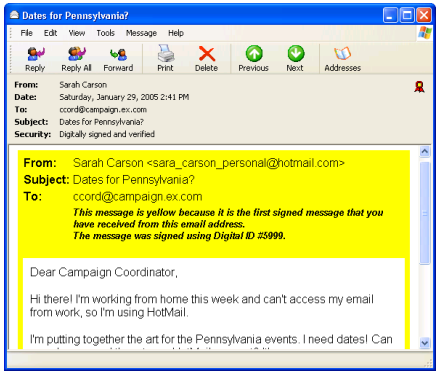
$p = 0.046$

KCM didn't help against the New Identity Attack:

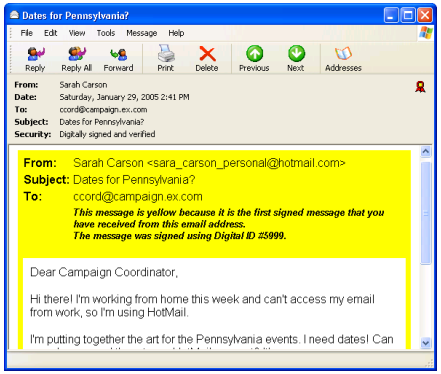
No Color



Color



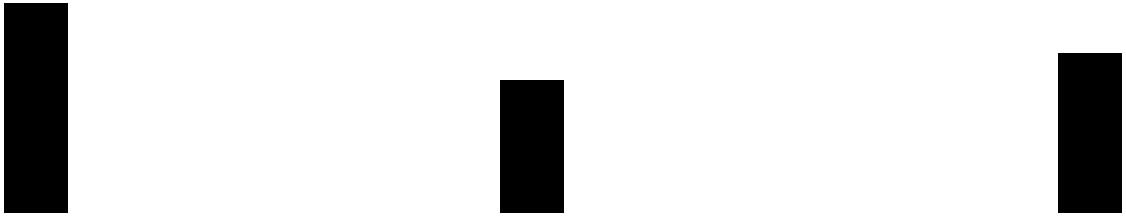
Color + Briefing



A Yellow Border will appear around an email message the first time a particular Digital ID is used with an email address.

Rate of successful attack:

79% 50% 60%



$p = 0.31$

The New Identity Attack is successful because the indicators are ambiguous!

- The attack matches a common situation in real-life.
- Subjects said that they knew there was a risk, but decided to ignore it.
- Only two noticed that Sarah's name was misspelled!

Evaluating the Usability of Encryption:

- Suprisingly, more people in NoColor encrypted than in Color or Color+Briefing
- It appears that they were (incorrectly) using encryption as a proxy for authentication
- Many people were confused by the Sign and Encrypt buttons in the OE interface

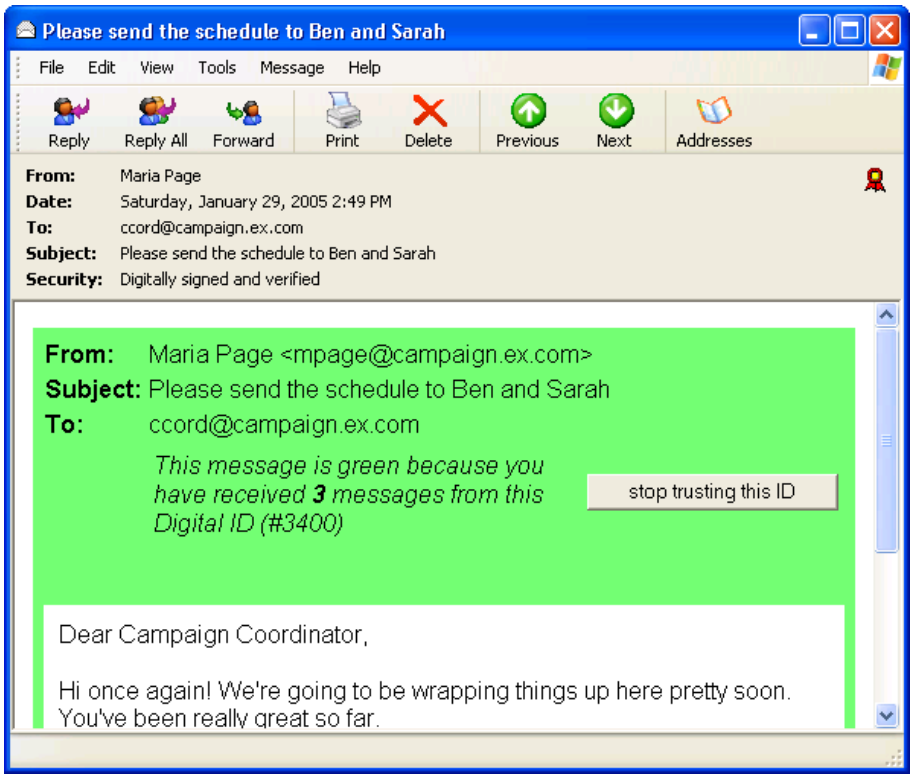
Colort	n	Clicked "encrypt" to seal email sometimes	always
NoColor	14	50%	21%
Color	14	36%	36%
Color+Briefing	15	20%	13%
$p =$		0.087	0.59

Interesting failings:

- Subjects were confused regarding single-click vs. double-click. They would double-click the “encrypt” button to no result!
- Subjects wanted to know how to make a Digital ID for Attacker Paul so they could send him the schedule!

Evaluation of CoPilot's Interface:

- People liked the colors, didn't read the text, and didn't understand the button
- People ignored the headers
- Serious confusion on commands vs. status on buttons
- Heavy users of web mail were the most confused.



Conclusion and Recommendations:

- We've previously argued that much commercial mail sent by eBay, Amazon, etc., should be signed.
- Johnny 2 shows that people can understand and use KCM with little or no training.
- S/MIME is much more usable than people give it credit.
- The hard thing is getting a certificate.
- KCM gives people certificates automatically, but leaves them susceptible to the New Identity Attack. (This is the phishing problem.)

Deployment Strategies:

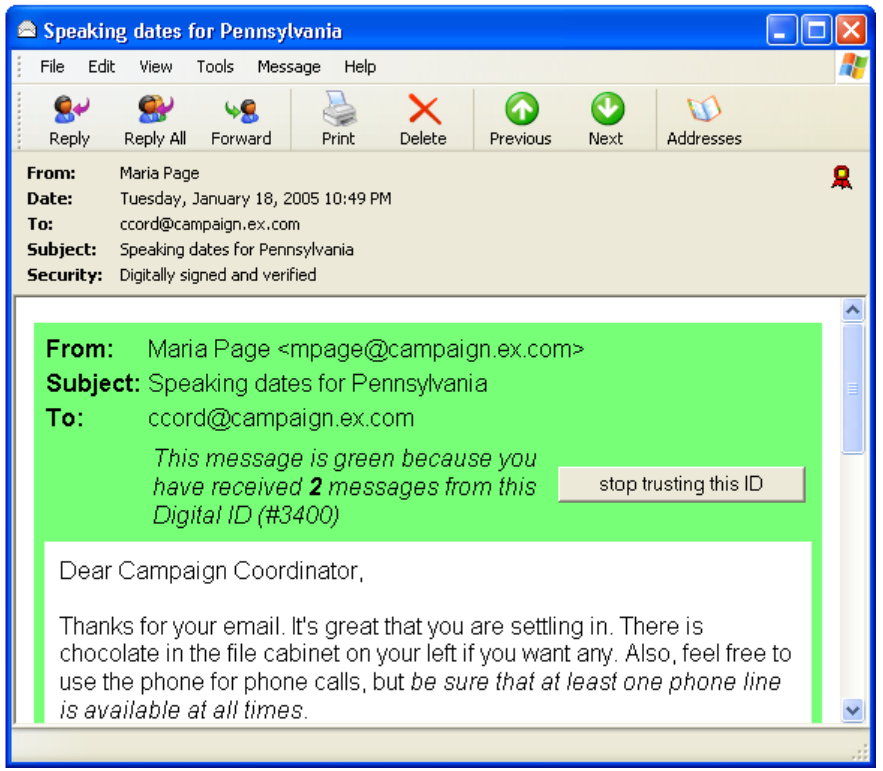
- You could build this in right now.
- Or Microsoft & Thawte could work together to make it easier for people to get email-only certificates.

We can improve usability and security by making better use of the tools we have already deployed.

Merchants like Amazon, eBay and PayPal should use S/MIME to sign their outgoing mail.

Most of what key continuity management offers can be accomplished with e-mail only S/MIME certificates.

A “CoPilot” that explains what certificates means can increase understanding, which increases usability and security.



Acknowledgements: Rob Miller, David Clark

Questions?