A client in the cross-hairs

how one software company deals with the challenge of protecting its users on the web

Rob Franco

Lead Program Manager, Windows CardSpace

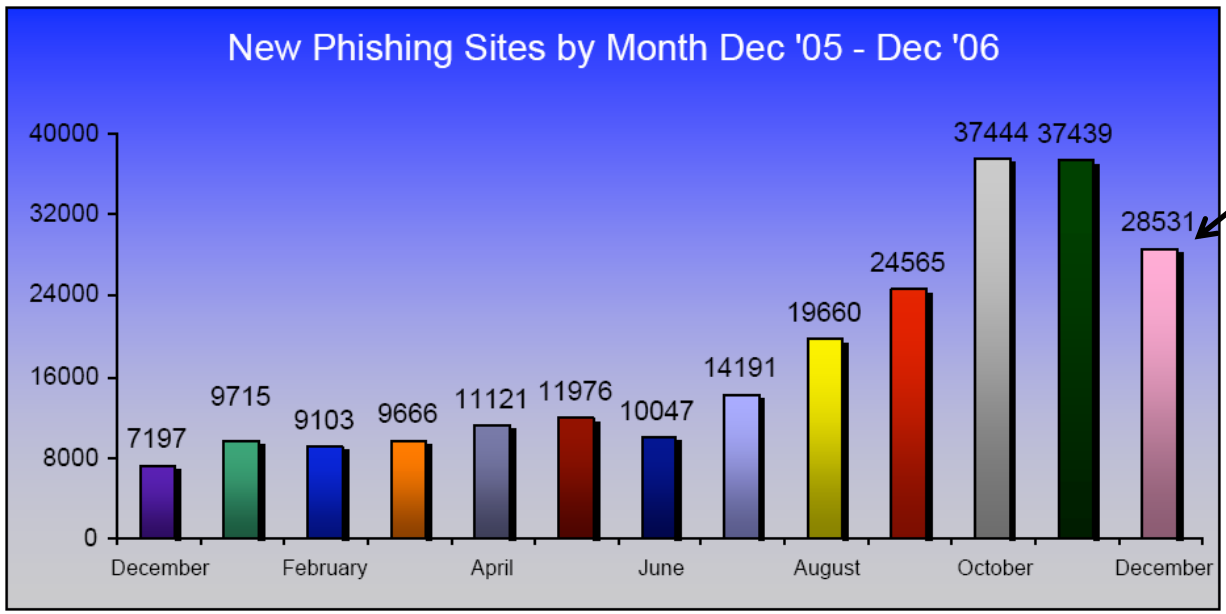
Contact: rfranco at microsoft dot com

Today's discussion

- Seeking the "Big Picture" for trust on the web
- Building trust today with IE7 and CardSpace
- A sample of Microsoft research on emerging threats

Phishing grows as counter measures take hold

- Financial Institutions growing in efficiency taking down phish sites and thwarting attacks before and after launch time
- Far more server power to drive attacks and collect consumer data



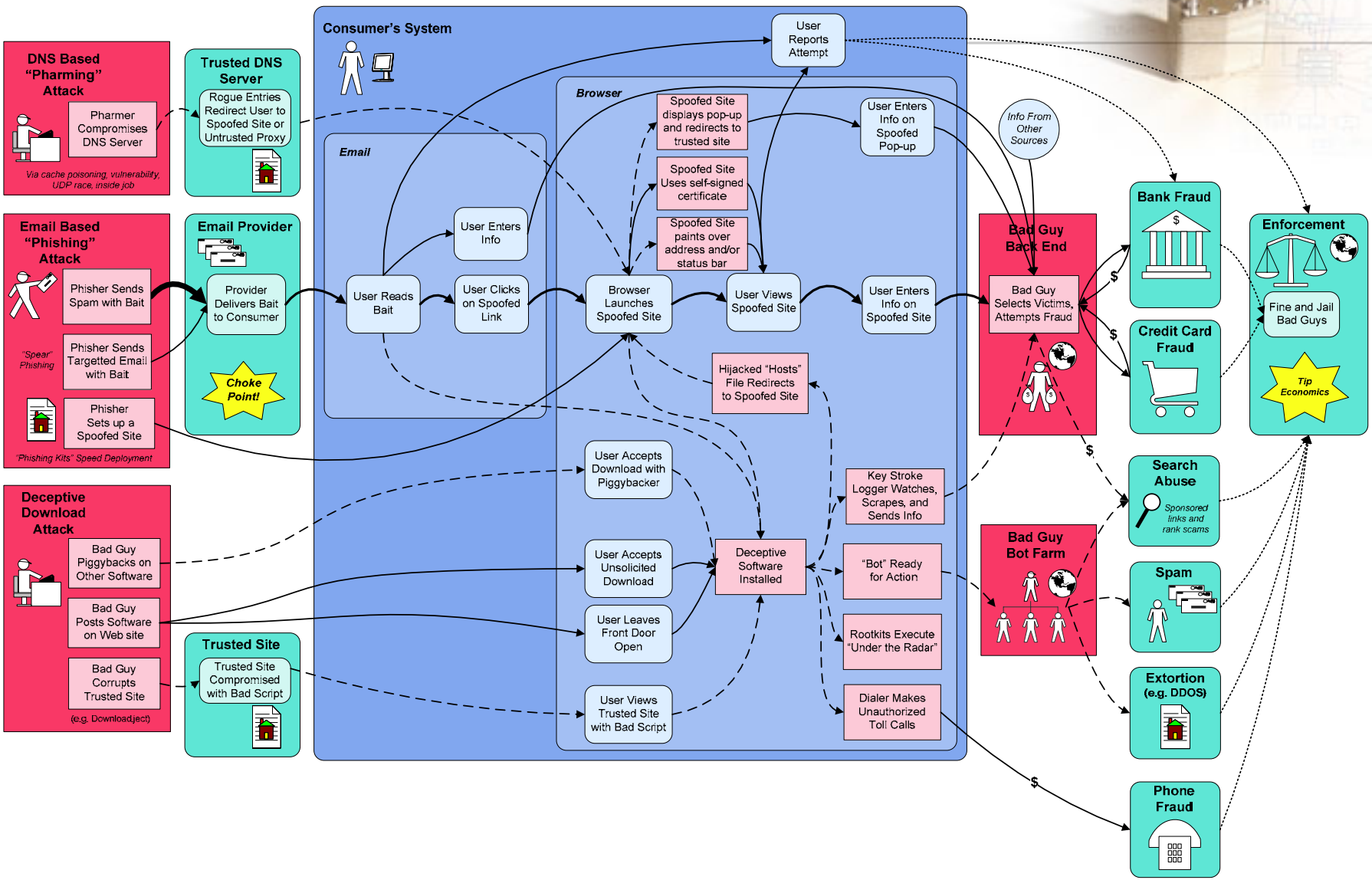
This is significantly lower than November (37,439), but it is perhaps due to phishers taking their usual holiday vacation.

Source: http://www.antiphishing.org

- 38 million pieces of “potentially unwanted” software were detected by Windows Defender between July 1, 2006, and December 31, 2006.

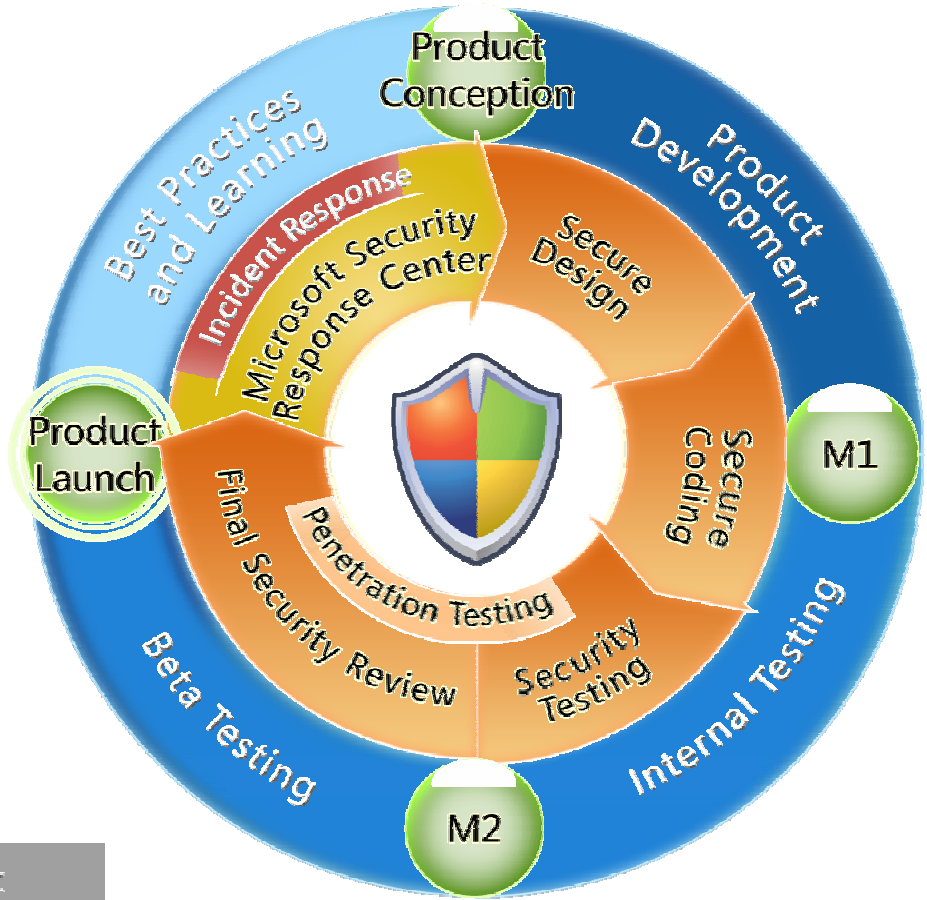
Source: Microsoft Security Intelligence Report

Internet "Battlefield"



© 2007 Microsoft Corporation

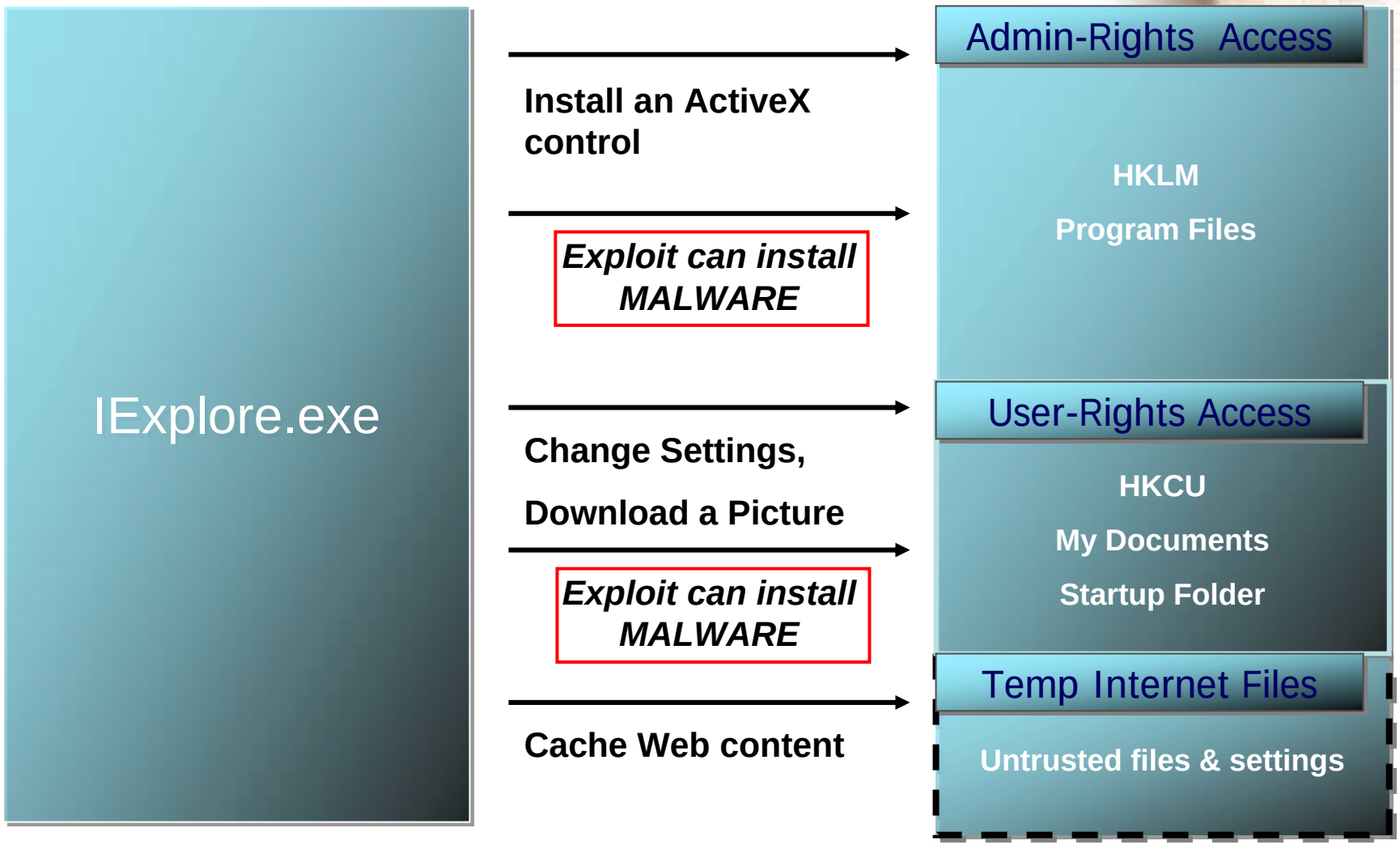
Microsoft Security Development Lifecycle



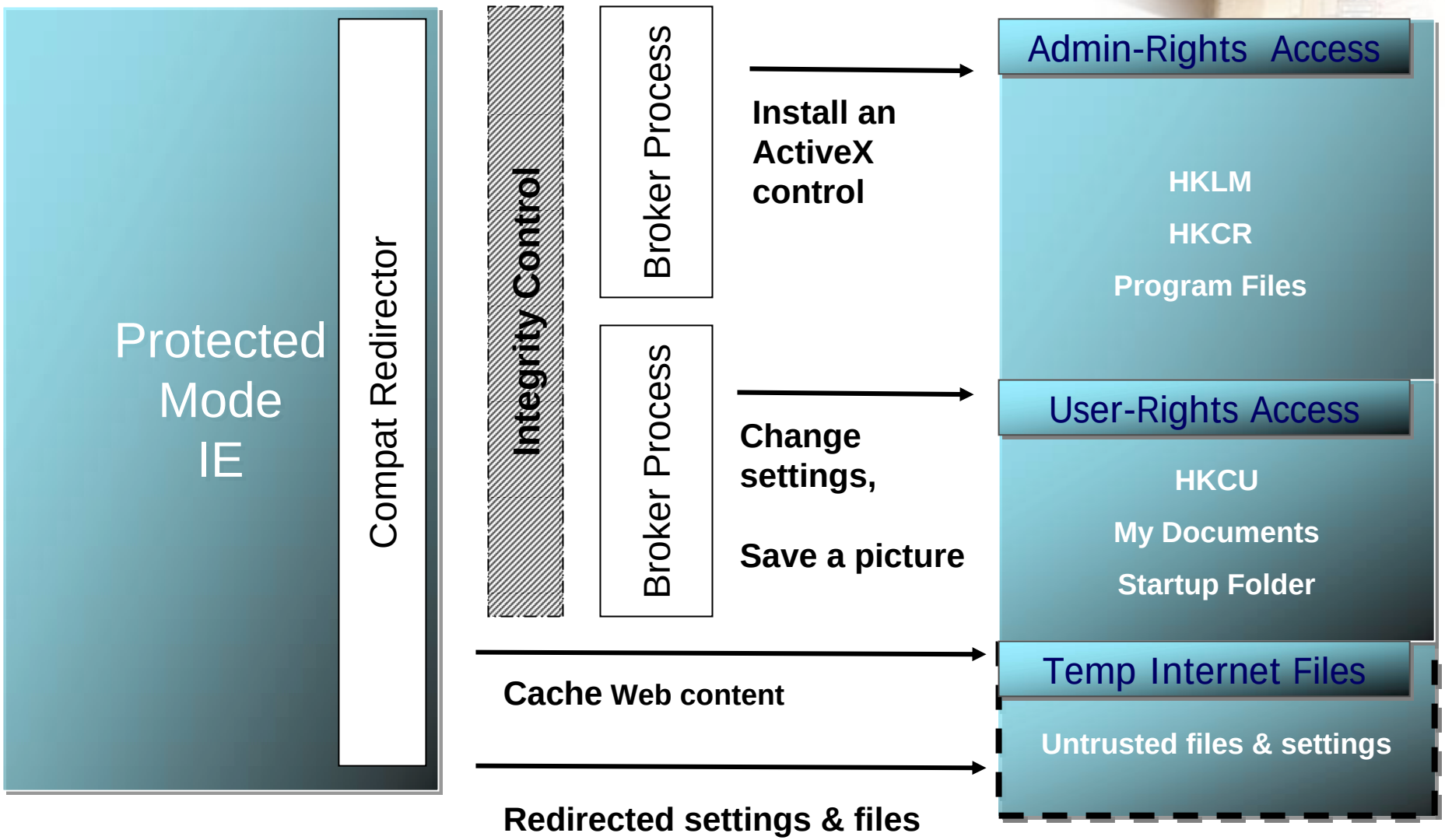
Microsoft Product Development Lifecyle

Microsoft Security Development Lifecycle

IE running as Admin on XP



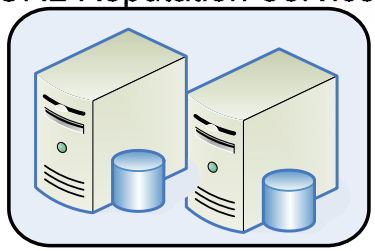
IE running in Protected Mode on Vista



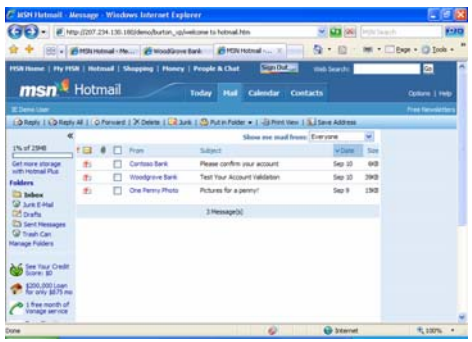
Phishing Filter

Client-side Heuristics, Allow-list and Web service

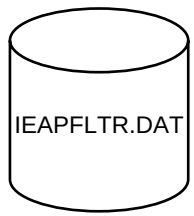
URL Reputation Service



https://urs.microsoft.com

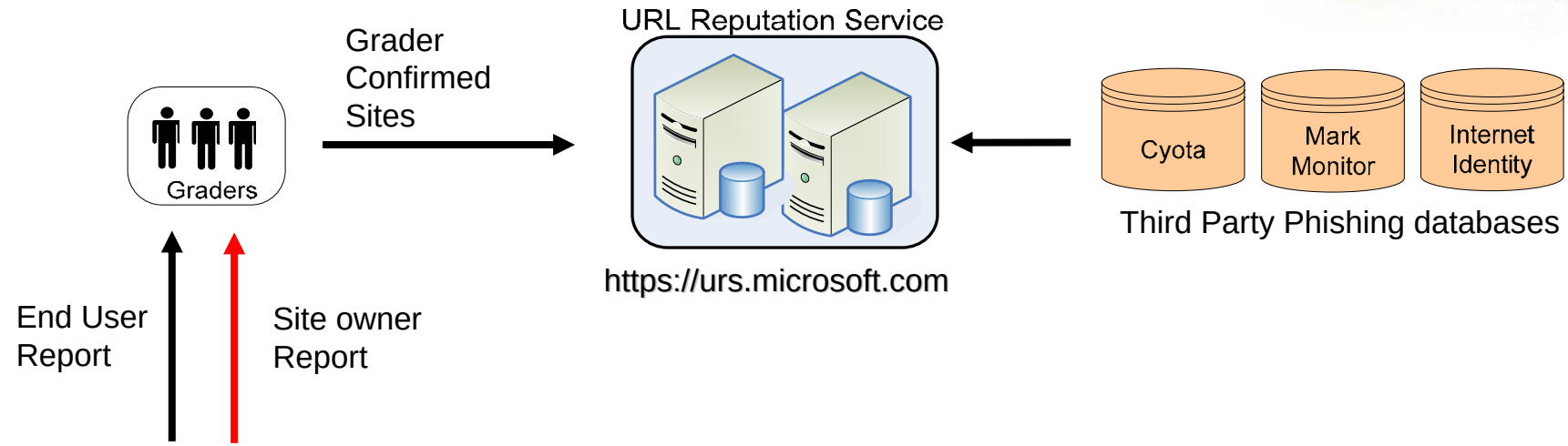


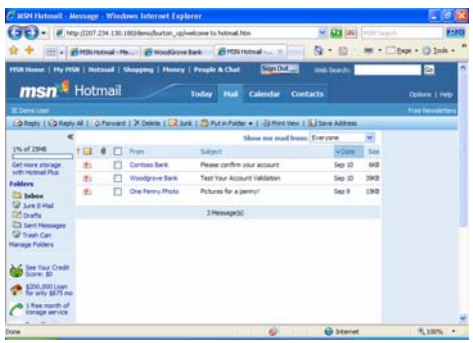
Known Good URLs



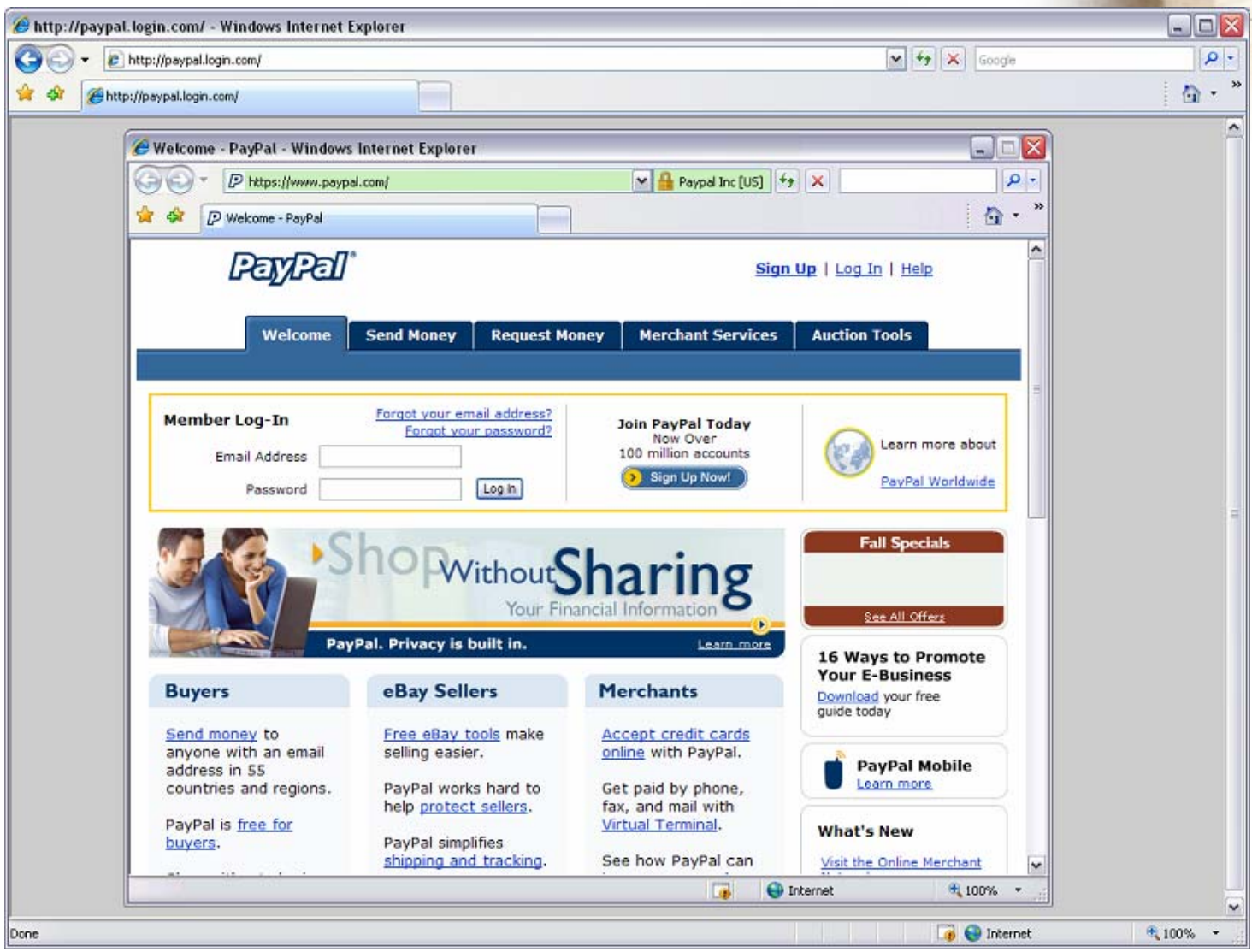
Phishing Filter

Populating the URL Reputation Service





Picture in Picture Spoof



The Laws of Identity

Established Through Industry Dialog

1. User control and consent
2. Minimal disclosure for a defined use
3. Justifiable parties
4. Directional identity
5. Pluralism of operators and technologies
6. Human integration
7. Consistent experience across contexts

Join the discussion at www.identityblog.com

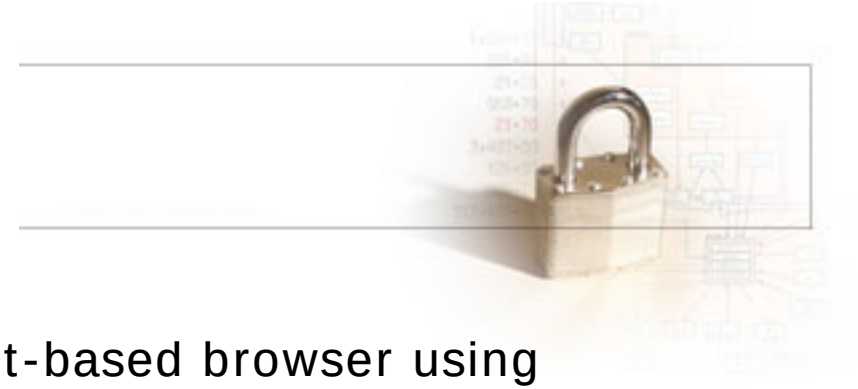
CardSpace Summary

- Reduces dependency on passwords
- Puts users in control
- Gets asymmetric keys in play without consumers realizing it
- Agnostic under the covers
  - Open standards (WS-*)
- Special ceremony, visual secrets
  - Separate Desktop
- Remembers relationships

Microsoft Research on emerging threats

MashupOS: Operating System Abstractions for Browsers

Web Browser Evolution



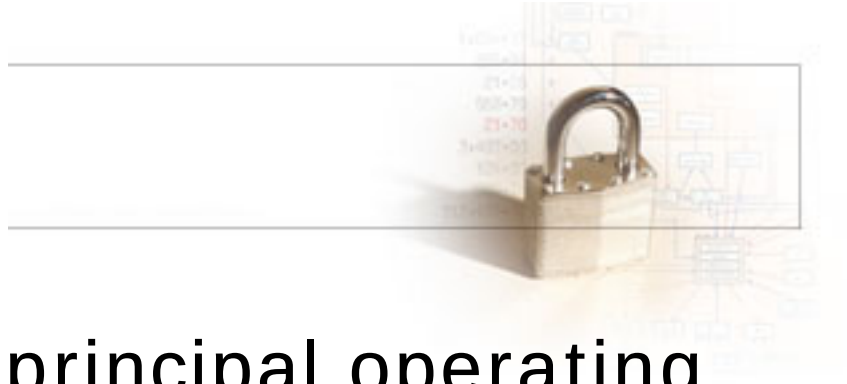
- Web browser history:
  - *1991*: World-Wide Web, first text-based browser using hypertext by Tim Berners-Lee
  - *Nov 1993*: Mosaic 1, GUI, images
  - *Dec 1994*: Netscape 1, cookies, multiple connections, tag
  - *March 1996*: Netscape 2, frames, JavaScript, beginning of DOM, SSL, Java, plugins
  - *August 1996*: Netscape 3 mouseovers; IE 3: CSS
  - *1997-1999*: browser bars, DHTML in both IE 4 and Netscape 4
  - *March 1999*: XMLHTTPRequest
  - *2000*: Outlook Web Access using XMLHTTPRequest
  - *Recent years*: AJAX, client-side mashups, Web 2.0
- Static documents, one site at a time → data content from different sites (images, frames) → programmability with DOM → dynamic HTML → AJAX & client-side mashups

Single-principal → Multi-principal platform (principal = site/domain)

Browser Abstractions Lag Behind

- Same origin policy (SOP)
  - A document or script loaded from one "origin" cannot get or set properties of a document from a different "origin".
- Binary trust model
  - Full trust: <script> third party code inclusion as library
  - No trust: <iframe> third party code isolation
- For other trust levels:
  - Sacrifice security for functionality:
    - JavaScript for data transport → "JavaScript Hijacking" [Fortify]
    - Live.com, iGoogle: fully isolated gadget or inline gadget:
      - "*Inline modules can... give its author access to information including your Google cookies... Click OK if you trust this module's author*"
  - Sacrifice functionality for security:
    - Social network web sites (myspace.com) deny scripts in user profiles to mitigate cross site scripting

MashupOS



- A browser-based multi-principal operating system
- Initial focus: abstractions for protection and communications
  - Prevent one principal from compromising the confidentiality and integrity of other principals
  - Communications allow principals to communicate in a controlled manner
- Initial paper at HotOS 2007
- Full paper to come

BrowserShield for Sanitizing Dynamic Content

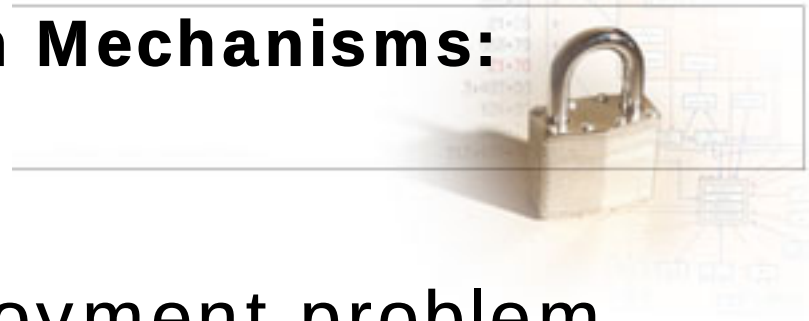
Motivation

- Web browser is a key interface between users and their electronic services and has rich functionalities and extensions.
- Web browser has also become a popular vector of attack: in 2005
  - IE: 8 patches,19 flaws
  - Firefox: 6 bulletins, 59 flaws
- Examples: tag attribute buffer overrun, flawed activeX controls, etc..

State-of-the-Art Protection Mechanisms: Patching

- Delayed deployment
  - User-driven
  - Hard to roll back
  - Administrator testing before patch application
- Dangerous time window between patch release and patch application
  - Attackers reverse-engineer patches to launch attacks
  - 90+% attacks exploiting known vulnerability [Arbaugh et al. 00]

State-of-the-Art Protection Mechanisms: Shielding

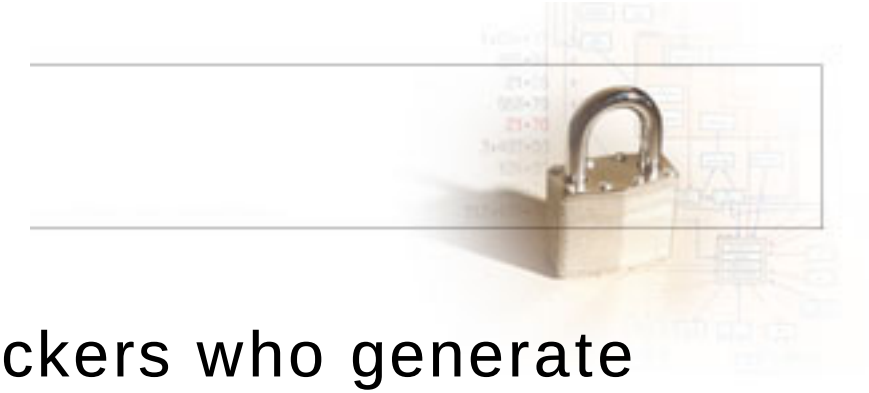


- Address the patch deployment problem
  - Patch the network input to the application rather than the application code itself
- Shield is a firewall that filters application level protocol traffic according to vulnerability signatures
- Vulnerability signature: vulnerability protocol state machine + reaction to malicious message format
- Easy deployment: automatic, easy removal
  - Enjoy same deployment model as AV

Can Shield Filter Browser Vulns?

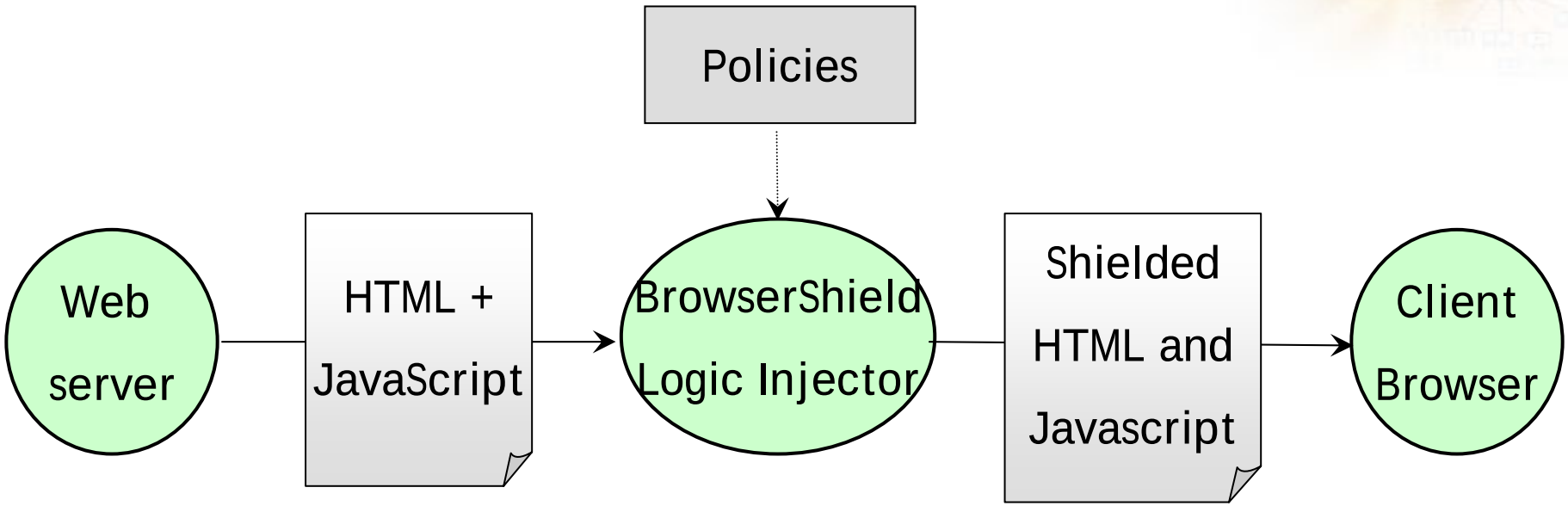
- Yes for static pages: HTML treated as another protocol layer above HTTP
- No for dynamic HTML:
  - Scripts in HTML can generate attacks at run time on the browser, evading detection
  - Does a script contain the logic for exploiting a vulnerability? An instance of the halting problem

BrowserShield Approach



- ***Key insight***: just like attackers who generate run-time attacks, we can also generate run-time protection.
- Our Approach:
  - Intercept HTML before being rendered by IE
  - Rewrite HTML and scripts to safe equivalents for browser to render
  - The safe equivalents contain logic that applies run-time checks and protection recursively to dynamically generated web content

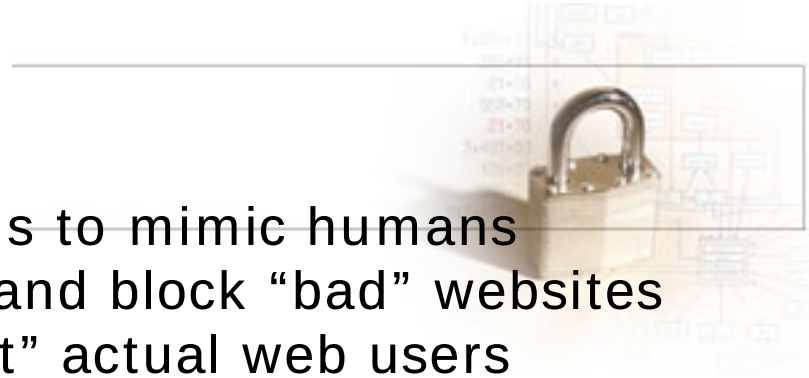
BrowserShield Overview



- Injector injects the rewriting logic; actual rewriting is offloaded to and executes on client browser
- Injector can be a firewall, browser extension, web publisher

Strider Monkeys

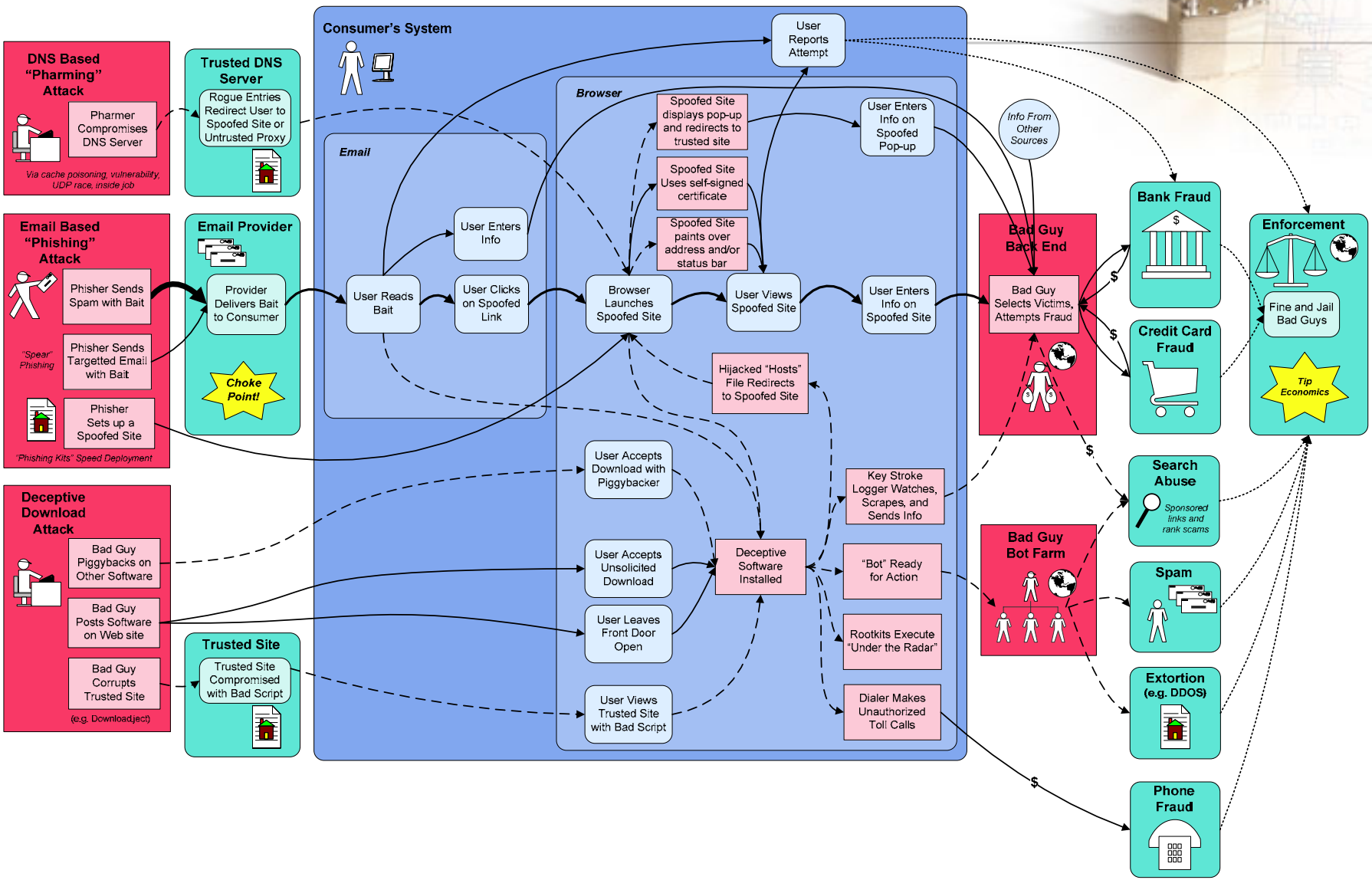
Strider Monkeys



- Basic idea: use monkey programs to mimic humans browsing the web and to detect and block “bad” websites before they get a chance to “hurt” actual web users

- Strider HoneyMonkey (http://research.microsoft.com/HoneyMonkey)
  - Use monkey programs running on unpatched virtual machines to scan the web and identify malicious websites that are exploiting (known or zero-day) browser vulnerabilities to install malware
- Strider Typo-Patrol (http://research.microsoft.com/URLTracer)
  - Use monkey programs to scan the “typo-neighborhoods” of popular websites to detect large-scale domain typo-squatters
- Strider Search Ranger (http://research.microsoft.com/SearchRanger)
  - Use monkey programs to scan search results of spammer-targeted keywords and perform traffic redirection analysis to identify large-scale search spammers (or called web spammers)

Internet "Battlefield"



Internet "Battlefield"