KaPoW!

Protecting the Web with Transparent Proof-of-Work

Ed Kaiser, Wu-chang Feng

Portland State University

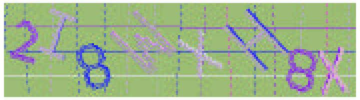
Supported by:

intel NSF

Motivation

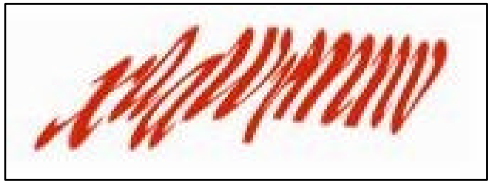
- Unwanted web traffic is everywhere
  - Denial of Service
  - Comment spam
  - Click fraud
  - Ticket robots
  - Fake web account signup
  - Duplicate on-line voting
- Observation
  - Most attacks are automated

CAPTCHAs to the rescue!

- Use a hard AI problem for security
  - Force users to solve a problem that is hard for a computer, but easy for a human
  - Turing test that does not require special client software
- Widely used
  - Google
  - Microsoft Live/Passport/Hotmail
  - Yahoo!
  - phpBB 

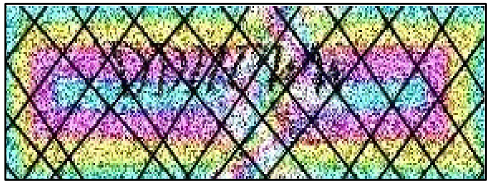
CAPTCHA Problem #1

- User-interface problem
  - Inaccessible to visually impaired
  - Some inaccessible to normal users



Blogger

Facebook

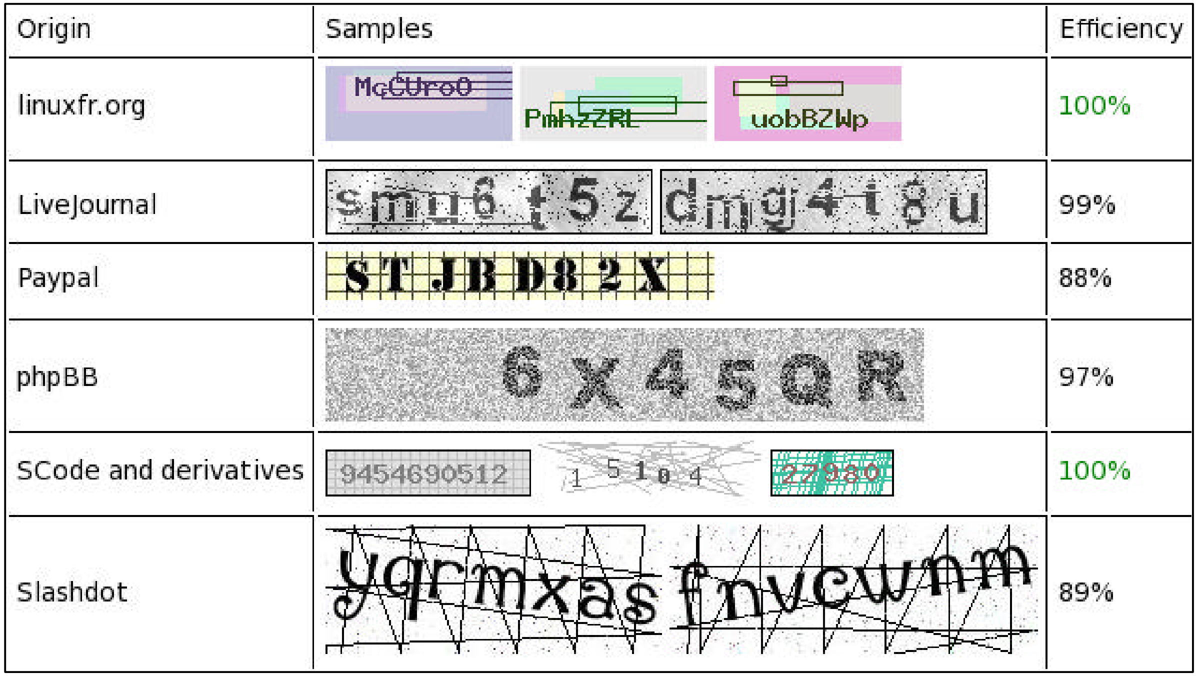


TicketMaster

  - Designed with several attempts in mind
    - frustrating, annoying, aesthetically unappealing experience
    - not suitable for frequent transactions

CAPTCHA Problem #2

- Adversaries solving the hard AI problem
  - Improvements to OCR erodes effectiveness
  - Examples
    - Yahoo! broken 1/2008
    - Windows Live/Passport, Google reported broken 2/2008
    - PWNtcha CAPTCHA solving library



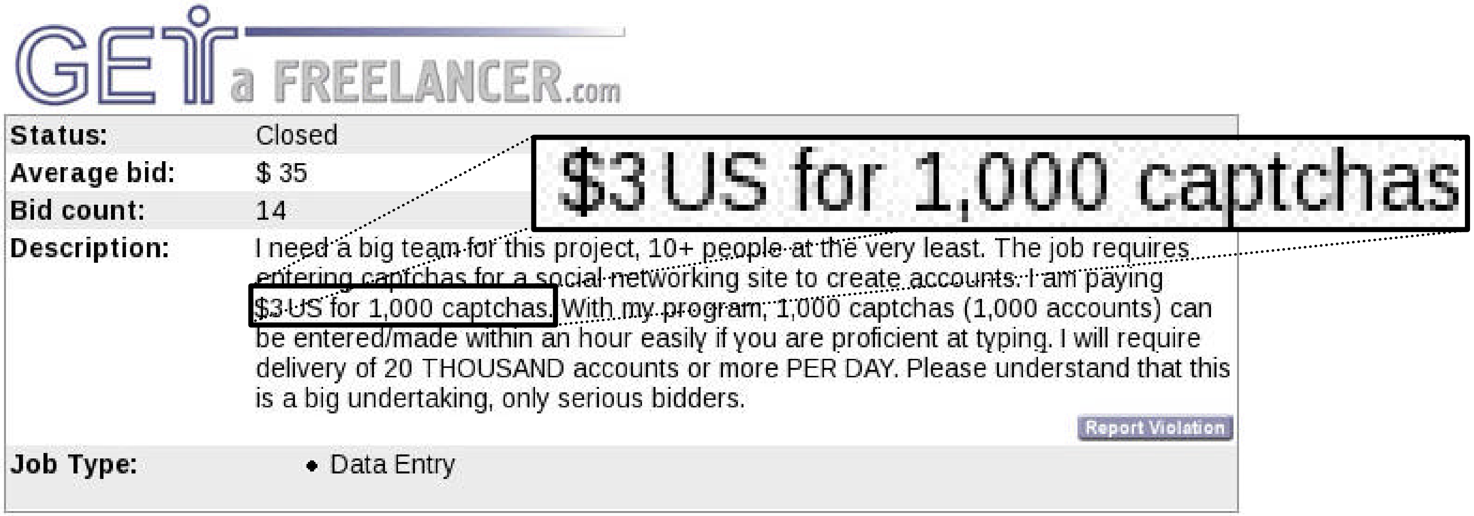
Origin	Samples	Efficiency
linuxfr.org	McCUro0 PmhzZRL uobBZWp	100%
LiveJournal	smu6 t5z dmg4i8u	99%
Paypal	ST JB D8 2X	88%
phpBB	6X45QR	97%

SCode and derivatives	9454690512
	1 5 1 0 4
	27980
		100%

Slashdot	yqrmxas fnvcwnm	89%

CAPTCHA Problem #3

- Economics broken
  - Fixed workload priced at 10 seconds of human time
    - Outsourced for under 1¢ per CAPTCHA



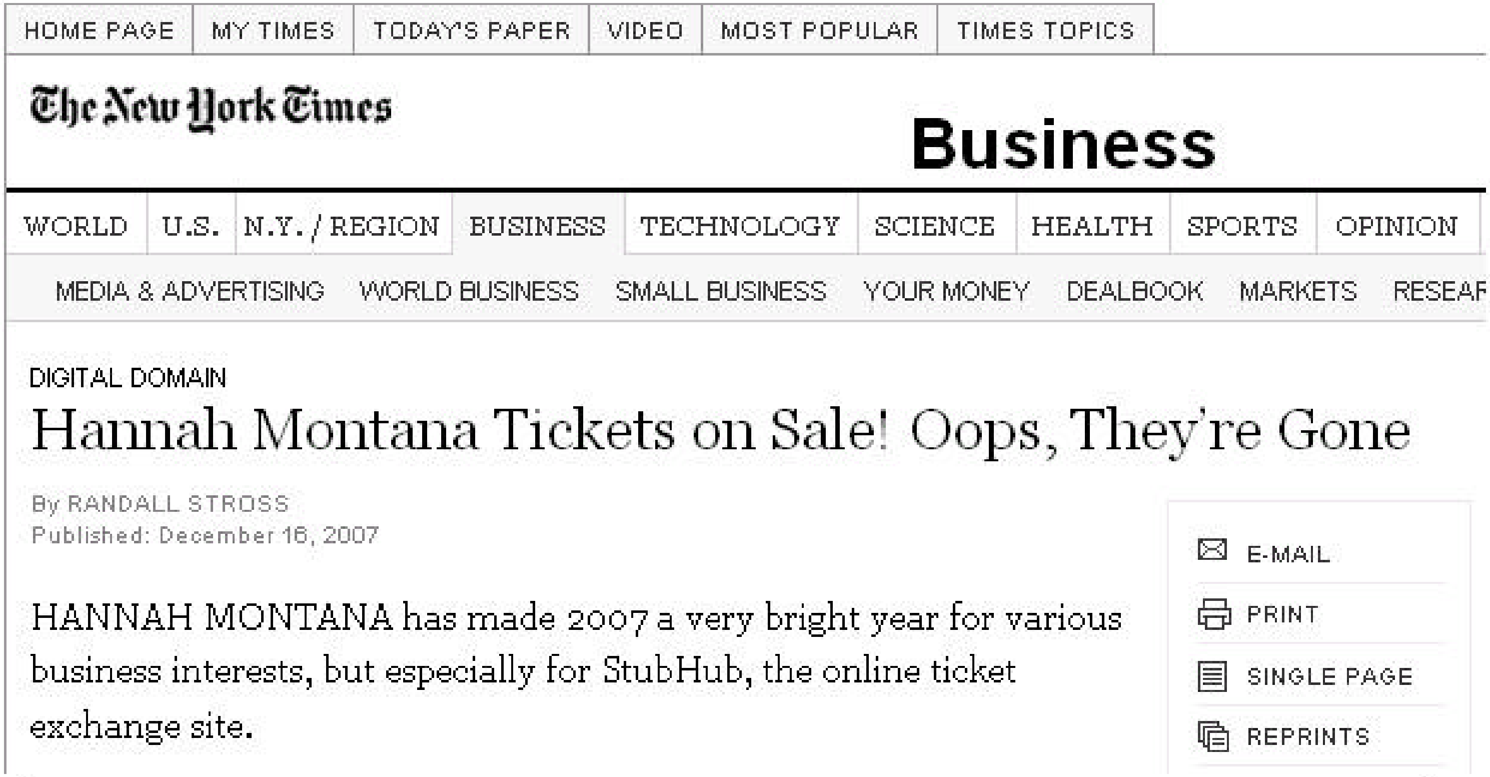
$3 US for 1,000 captchas

Status:	Closed
Average bid:	$ 35
Bid count:	14
Description:	I need a big team for this project, 10+ people at the very least. The job requires entering captchas for a social networking site to create accounts. I am paying $3 US for 1,000 captchas. With my program, 1,000 captchas (1,000 accounts) can be entered/made within an hour easily if you are proficient at typing. I will require delivery of 20 THOUSAND accounts or more PER DAY. Please understand that this is a big undertaking, only serious bidders.
Job Type:	• Data Entry

  - CAPTCHA pricing does not work
    - When adversary resources are vastly greater than legitimate ones
    - When value of what is being protected is more than 1¢

CAPTCHA Problem #3

- Example

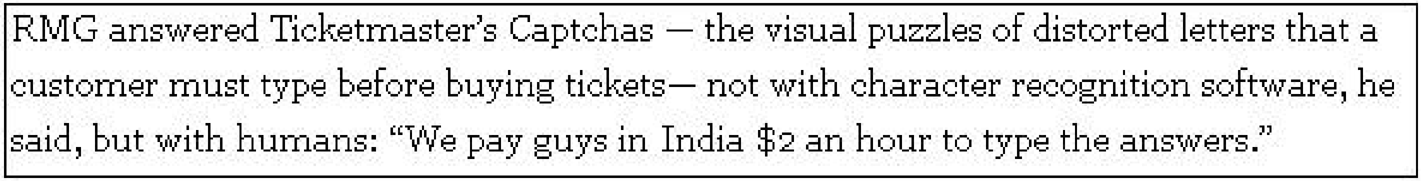


DIGITAL DOMAIN

Hannah Montana Tickets on Sale! Oops, They're Gone

By RANDALL STROSS
Published: December 16, 2007

HANNAH MONTANA has made 2007 a very bright year for various business interests, but especially for StubHub, the online ticket exchange site.



RMG answered Ticketmaster's Captchas — the visual puzzles of distorted letters that a customer must type before buying tickets— not with character recognition software, he said, but with humans: "We pay guys in India $2 an hour to type the answers."

Need a variable workload to price out adversaries!

Proof-of-Work (PoW)

- Alternative to CAPTCHA
  - Clients solve a computational puzzle to get access
- Addresses CAPTCHA problems
  - No user interface issues
  - Adversary must solve a hard cryptographic problem
  - Adjustable difficulty that treats CPU cycles as currency

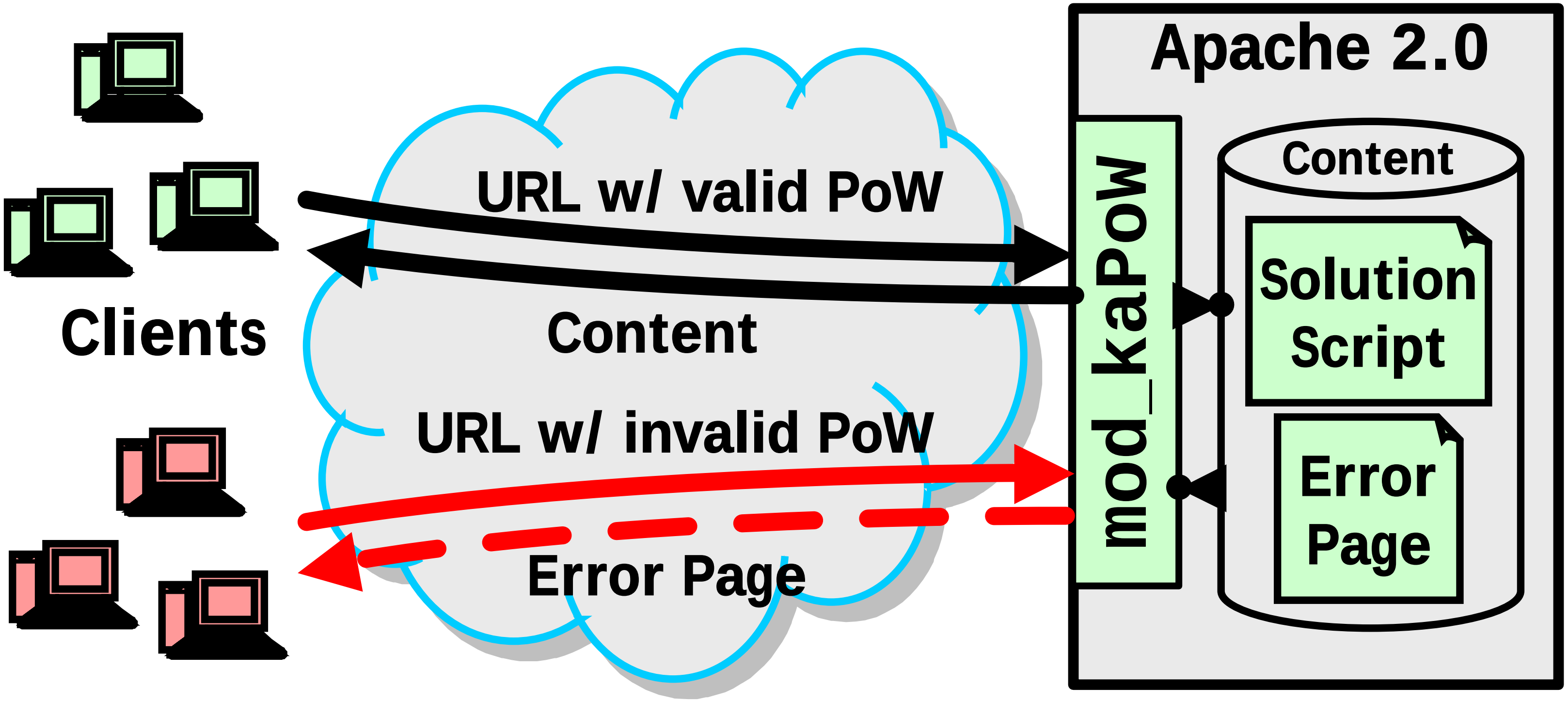
But…

- Landscape littered with unused PoW schemes!
  - Hash cash, TLS puzzles, TCP puzzles
  - IP puzzles, Public puzzles (two of our own stinkers)
- Why?
  - Introduces a big problem CAPTCHA does not
  - Forces changes to network protocols and software
  - Client must install PoW software to participate

Our approach: `mod_kaPoW`

- Provide benefits of PoW without changes to client
  - Apache module
    - Dynamically embedds PoW with client-specific difficulty into URLs
    - Attaches JavaScript solver for client to run
    - Verifies subsequent solutions
  - Client browser
    - Runs JavaScript solver to calculate answers
    - Attaches answers to subsequent URL requests
  - No protocol changes
  - No web browser changes
  - No web content changes

mod_kaPoW architecture



mod_kaPoW puzzle

- Based on targeted hash reversal

  Wu-chang Feng, Ed Kaiser, “The Case for Public Work”
  Global Internet 2007

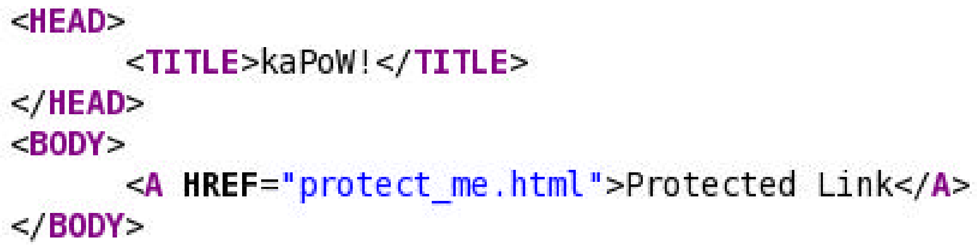
- Server attaches puzzle to embedded links
  - N_c = client-specific server-generated nonce
  - D_c = client-specific server-assigned difficulty
- Client JavaScript solver finds A such that

  $$\mathtt{SHA1}(N_c \mathbin{||} \mathtt{URL} \mathbin{||} A) = 0 \bmod D_c$$

  - Brute-force search requiring D_c SHA1 hashes on average to find
  - Attaches N_c, D_c, and A to URL to access content

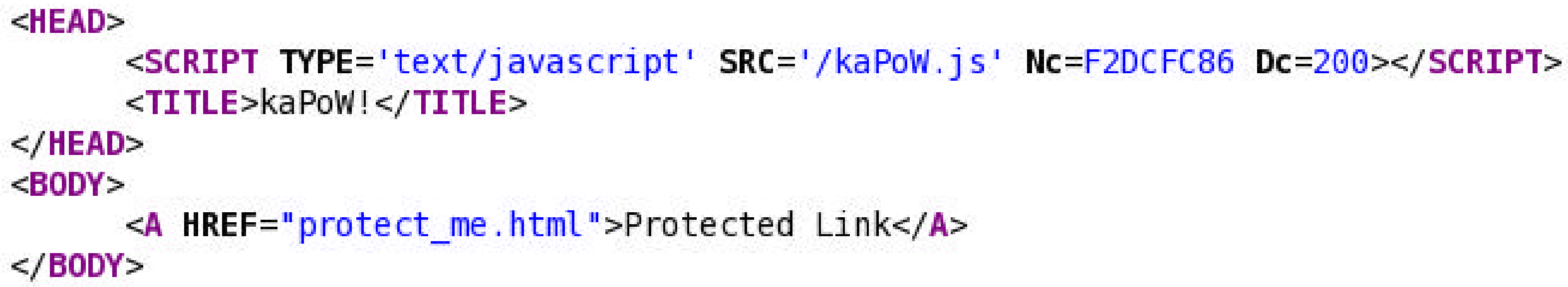
Example

- Original content on disk



```
<HEAD>
      <TITLE>kaPoW!</TITLE>
</HEAD>
<BODY>
      <A HREF="protect_me.html">Protected Link</A>
</BODY>
```

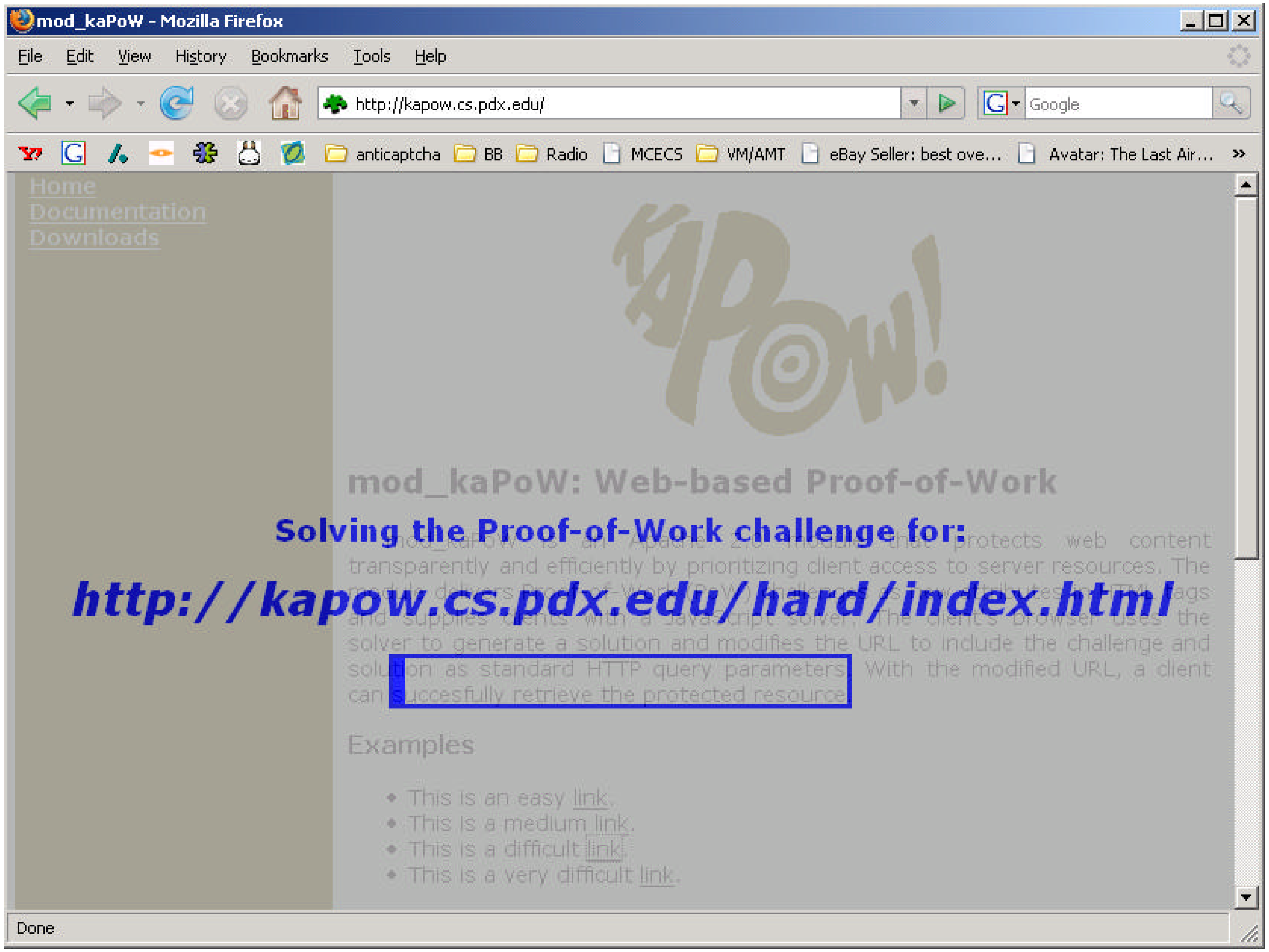
- Content after Apache embedding of PoW



```
<HEAD>
      <SCRIPT TYPE='text/javascript' SRC='/kaPoW.js' Nc=F2DCFC86 Dc=200></SCRIPT>
      <TITLE>kaPoW!</TITLE>
</HEAD>
<BODY>
      <A HREF="protect_me.html">Protected Link</A>
</BODY>
```

- JavaScript solver kaPoW.js
  - Registers “onLoad” and “onClick” event handlers
  - Implements SHA1 to solve PoWs of URLs given puzzle parameters
    - “onLoad” for embedded images
    - “onClick” for embedded links

Demo



mod_kaPoW: Web-based Proof-of-Work

Solving the Proof-of-Work challenge for:

http://kapow.cs.pdx.edu/hard/index.html

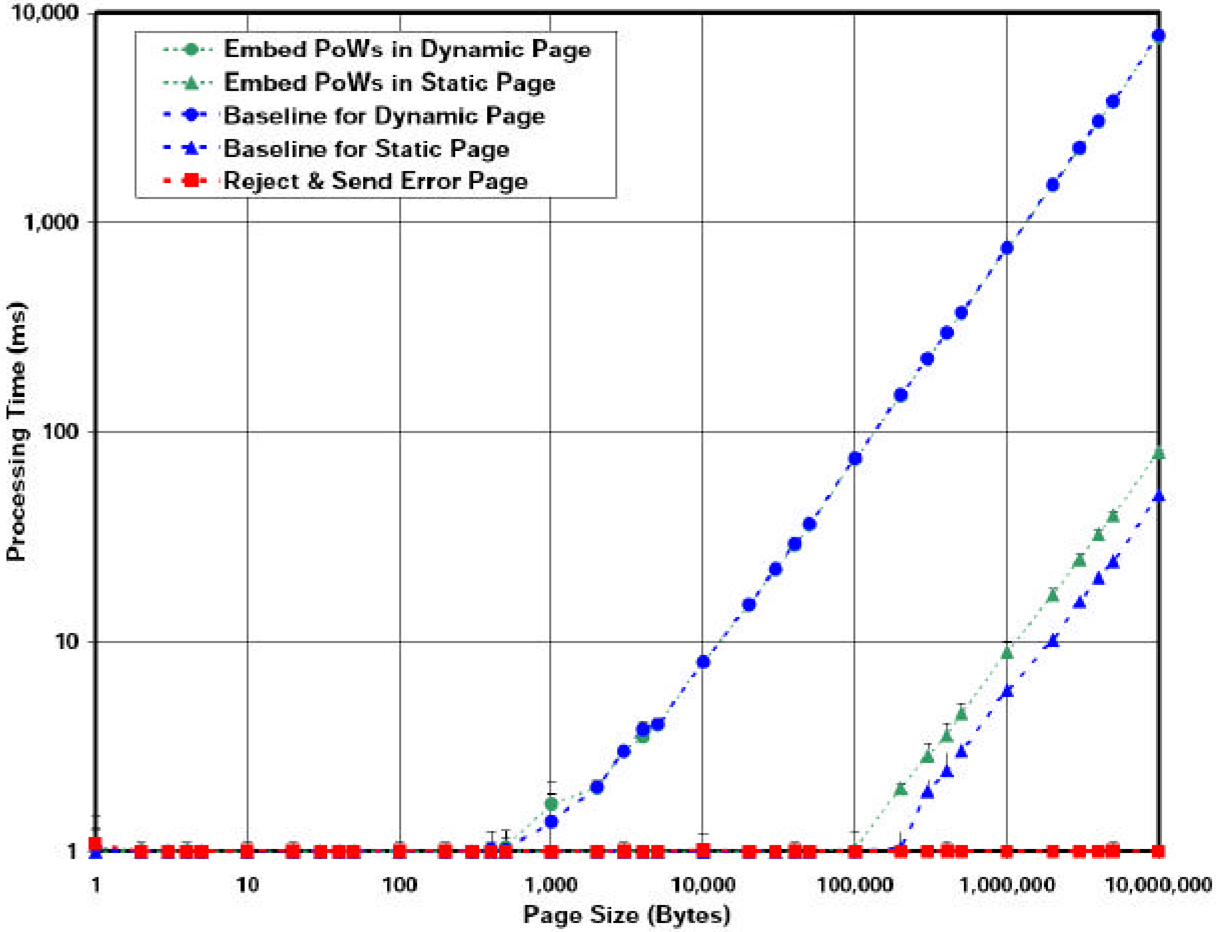
mod_kaPoW is an Apache 2.0 module that protects web content transparently and efficiently by prioritizing client access to server resources. The module delivers Proof-of-Work (PoW) challenges as new attributes to HTML tags and supplies clients with a JavaScript solver. The client's browser uses the solver to generate a solution and modifies the URL to include the challenge and solution as standard HTTP query parameters. With the modified URL, a client can succesfully retrieve the protected resource.

Examples

- This is an easy link.
- This is a medium link.
- This is a difficult link.
- This is a very difficult link.

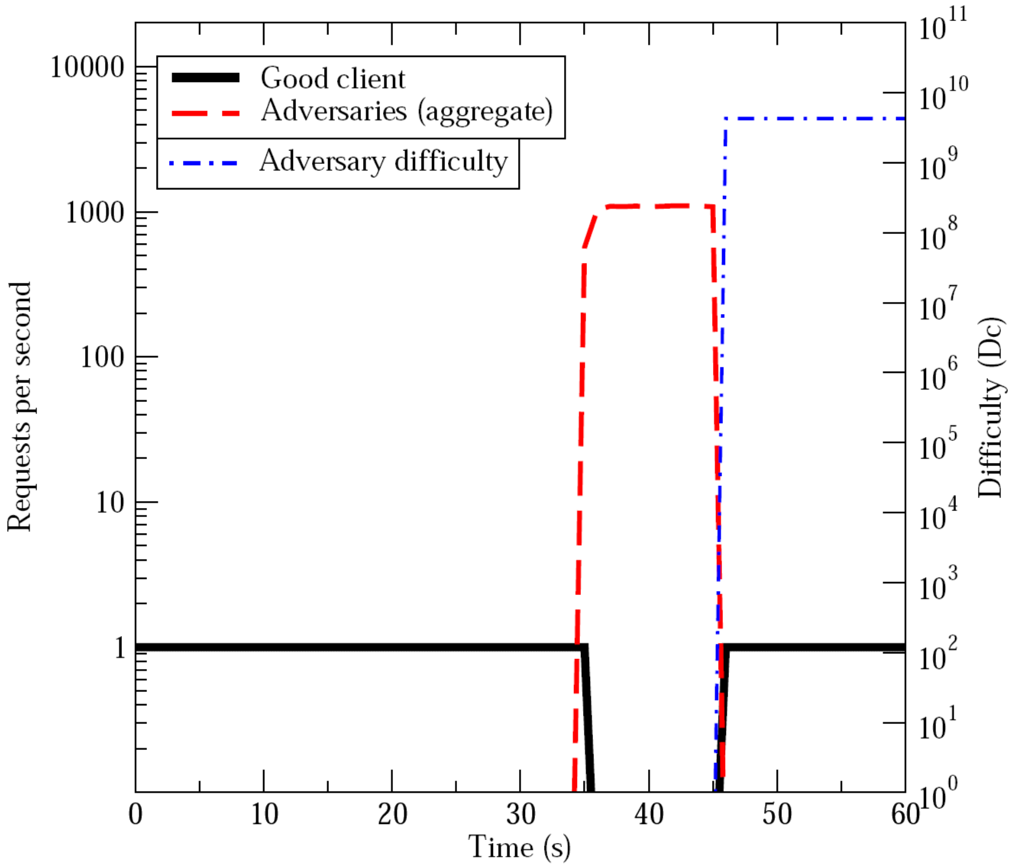
Overhead

- Negligible for dynamic page
- Small fixed amount for static page
- Fast verification and rejection



Thwarting DoS

- Simple experiment
  - Good client at 1 request per second
  - 6 flooding adversaries attack at 35 second mark
  - Counting Bloom Filter used to track usage and set difficulty



What next?

- Towards a computational approach for protecting Internet applications
- Building applications around kaPoW
  - Treat CPU cycles as currency and create virtual markets
  - Use cycles to create incentives for proper behavior
  - Force adversaries (spammers, ticket brokers, hackers) to “pay” for access
    - A tax paid to Intel!

Tackling comment spam

- Content-based difficulties
  - Force “spammy” comments to use a large amount of cycles
  - Send posts through SpamAssassin and use its score to determine puzzle difficulty
- Weighted voting
  - Allow users to “vote” on comments with their CPU cycles
  - Promote comments with the most committed cycles
- Community-assisted pricing
  - Allow users police the price for posting for each other based on prior posts
  - Use “karma” (Slashdot) to determine CPU cycles a particular user needs to post

Tackling click fraud

- Increase click costs on suspected fraud
  - Apply credit-card fraud techniques to detect possible fraud
  - Increase CPU tax on ad click-throughs that are suspicious
    - Use prior history of clicks to prevent Auction Experts employees from “clicking-through” Google ads

Tackling ticket robots

- Increase cost of “purchase” link geographically
  - Use MaxMind/GeoIP to determine where clicks originate
  - Increase costs on those far away
  - Forces ticket robots to be located in each city
    - Much better economics than $0.01 CAPTCHAs!

Roadmap

- Adding to LAMP stacks
  - Linux, Apache, MySQL, PHP/Perl
  - Allowing applications to control difficulty
  - phpBB, WordPress, Twiki, Drupal, guestbooks
- Using with CAPTCHA
  - Frequent transactions protected with kaPoW
  - Infrequent transactions protected by both

A brief plug on AMT work

- CS 576: Detecting Cheating in On-line Games
  - Repeating last year’s successful offering
  - Using Intel’s AMT as an undetectable debugger
  - What exploits used by cheat software could be reliably measured by the AMT?
- NSF FIND, GENI
  - Clean-slate design of the Internet
  - Building Future Networks Around Ubiquitous Use of AMTs
    - Trusted Third Parties make many security protocols easy
    - Can TPMs acting as TTPs fix problems in network protocol design?
    - An interesting academic exercise (for now)

Questions?

http://kapow.cs.pdx.edu

Extra slides

Addressing economics

- How do you construct a pricing system that works?
  - What is the cost of unattended (idle) CPU cycles?
  - Can costs be controlled to create sufficient disincentives for botnets of 20,000 idle machines?
  - How much is it worth to keep bots hidden?
  - How do you cope with price limits to legitimate users?