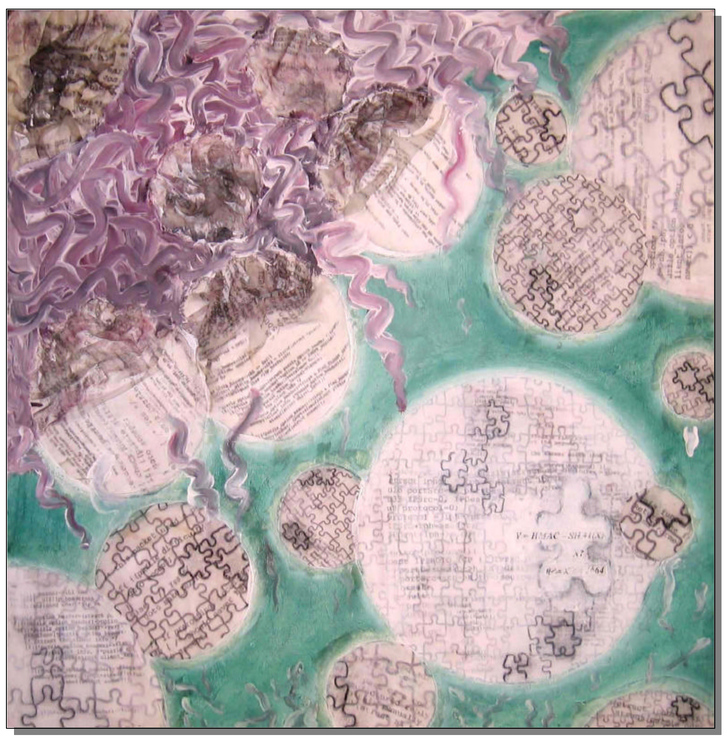
Reducing Malicious Traffic With IP Puzzles

Ed Kaiser, Wu-chang Feng, Wu-chi Feng, Antoine Luu



Motivation

Arrgh! There is so much bad traffic on the internet!

- DoS attacks
- Port scans
- Spam e-mail
- Worms
- Hacking
- Game cheaters

Question: What can be done?
Answer: Make clients accountable for their behavior by using a mechanism for punishing them if they behave badly.

Client puzzles offer an ideal punishment mechanism:
- Easy to assign punishment
- Can make punishment arbitrarily difficult
- False positives degrade but do not deny service

Other work secures individual protocol vulnerabilities, however the most effective solution should protect all network traffic; thus it must be placed in the IP layer.

Our approach:

IP layer client puzzles

Challenges

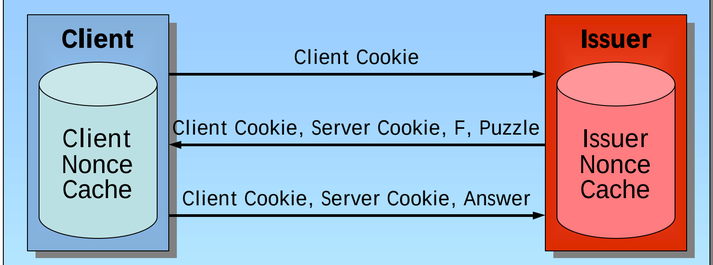
Flexible Deployment
- Puzzle issuers at arbitrary network locations

Minimal Overhead
- Puzzles can be generated at line speed
- Constant state at the puzzle issuer
- Minimal packet expansion

Tamper Resistance
- Replay attacks
- Spoofing attacks
- Work ahead attacks

Support for Real Time Apps
- Online games
- Streaming media

Puzzle Protocol



Protocol Field	Description
Client Cookie	TS_c, N_c
Server Cookie	TS_s, TS_m, TS_e, h(F, TS_c, N_c, TS_s, N_s, TS_m, TS_e)
Puzzle	Difficulty, Puzzle Parameters
Answer	Puzzle Answer
TS_c	Client Logical Timestamp
N_c	Client Nonce
TS_s	Issuer Logical Timestamp
N_s	Issuer Nonce
F	Flow Identifier
TS_m	Puzzle Maturity Time
TS_e	Puzzle Expiry Time
h()	Hash Message Authentication Code (HMAC)

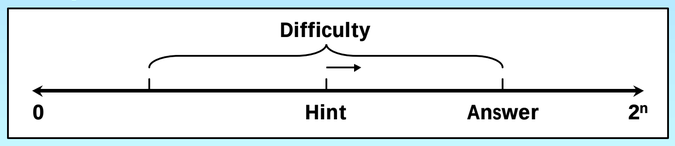
Puzzle Algorithm

Hint-Based Hash-Reversal

Requires:
- Keyed HMAC; h()
- high entropy random number generator; *rand*()

Creating the Puzzle:

1) Answer ← *rand*()
2) Hint ← Answer - (*rand*() mod Difficulty)
3) Puzzle Hash ← h(Answer)
4) discard the Answer



Solving the Puzzle:

1) Search Value ← Hint
2) if h(Search Value) = Puzzle Hash
   Answer ← Search Value
3) Search Value ← Search Value + 1
4) go to step 2

Protocol Extensions

IP Options Cookie:

Type = 25	Length	Control
Client Timestamp		Client Nonce

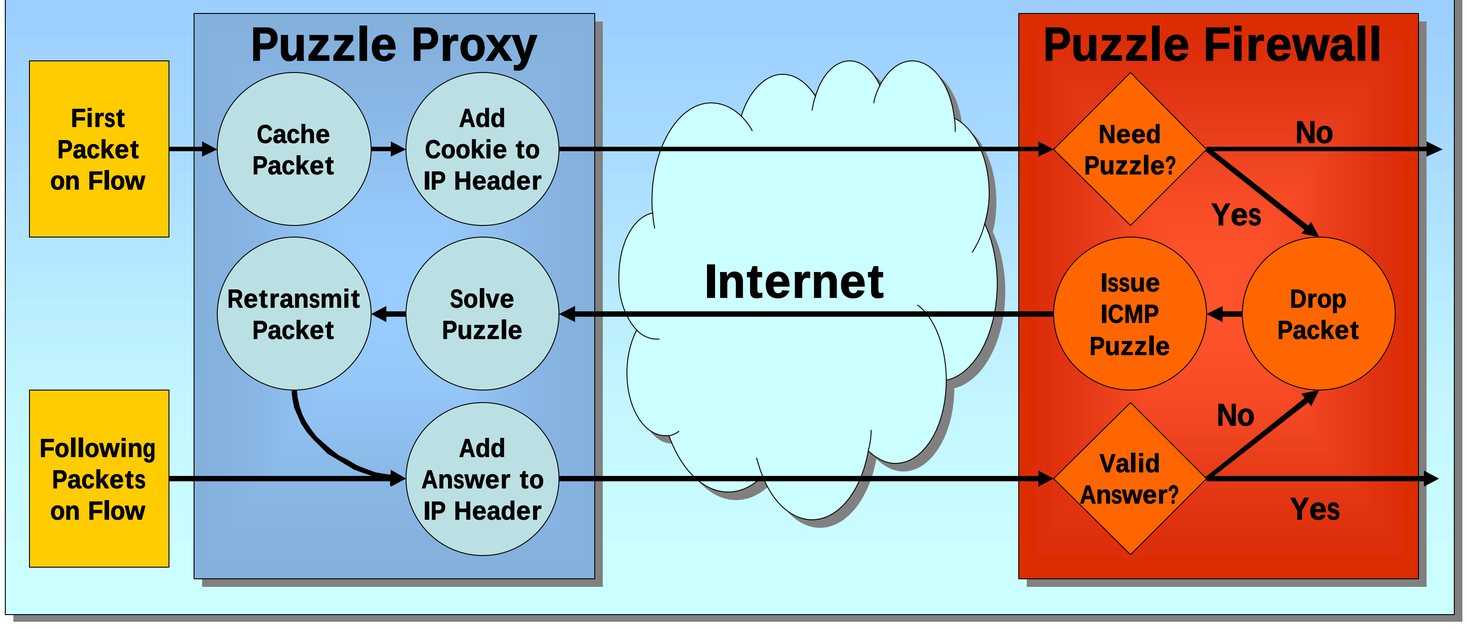
ICMP Puzzle:

Type = 38	Code	Checksum
PuzzleType	Length	Control
Client Timestamp		Client Nonce
Issuer Timestamp		Maturity Time
Expiry Time		Protocol
Client IP		
Server IP		
Client Port		Server Port
Hash of Parameters and Secrets		
Puzzle Difficulty		
Puzzle Parameters (variable length)		

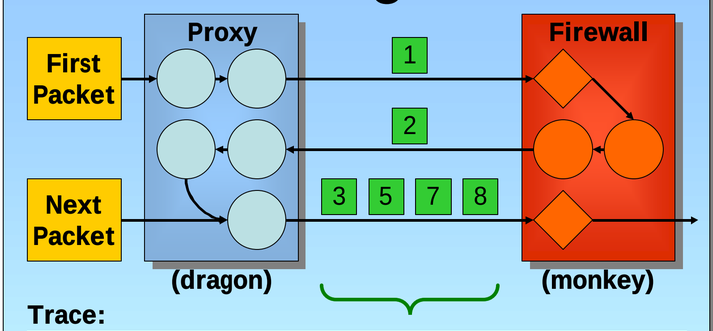
IP Options Answer:

Type = 26	Length	Control
Client Timestamp		Client Nonce
Issuer Timestamp		Maturity Time
Expiry Time		
Hash of Parameters and Secrets		
Puzzle Answer (variable length)		

`iptables` Implementation



Tracing `ssh`



Trace:

```
tcpdump: listening on eth0
20:54:05.570461 dragon.32803 > monkey.22: S        1
20:54:05.570644 monkey > dragon: icmp: type-#38    2
20:54:05.570679 dragon.32803 > monkey.22: S        3
20:54:05.570826 monkey.22 > dragon.32803: S
20:54:05.570853 dragon.32803 > monkey.22: .        5
20:54:05.572148 monkey.22 > dragon.32803: P
20:54:05.572190 dragon.32803 > monkey.22: .        7
20:54:05.572317 dragon.32803 > monkey.22: P        8
20:54:05.572445 monkey.22 > dragon.32803: .
 ...
```

Performance

Constant State at Issuer

Fast to Issue
- requires only one hash and two random numbers

Fine Grain Difficulty Control
- can linearly increment puzzle difficulty

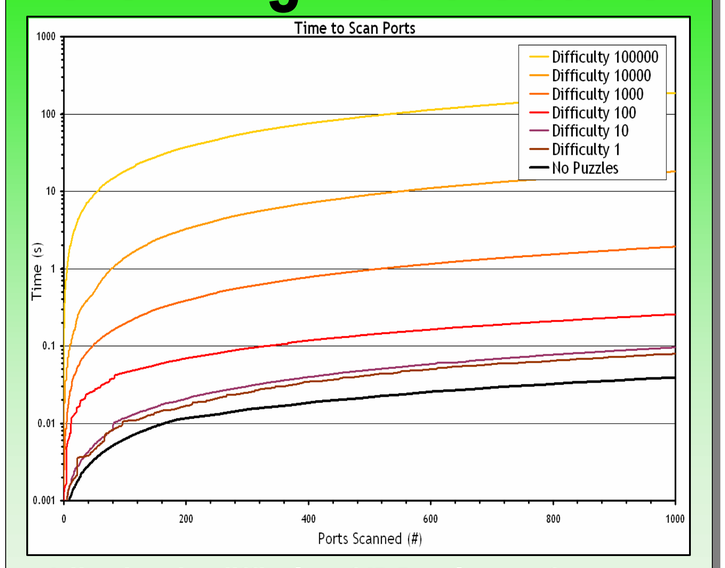
Throughput

Tests use:
- Dual 1.8GHz Intel Xeon machines
- Cisco Catalyst 4006 Gigabit switch

Firewall:
- validate and issue puzzles at 182,000 packets/s

Proxy:
- solve min-difficulty puzzles at 130,000 packets/s
- solve max-difficulty puzzles at << 1 packets/s

Slowing Port Scans



Adjusting the difficulty of IP Puzzles can force port scans to take a selectively long time to complete.

Future Work

Reputation-Based Networking
- Keep interaction history about clients
- Determine their reputability
- Use IP Puzzles to punish clients who are bad
- Share knowledge with other IP Puzzle firewalls

Publicly Auditable Puzzles
- Puzzle answers can be independently verified by intermediate IP Puzzle routers
- Answers can indicate amount of work done

Puzzles With Useful Answers
- Puzzle algorithms where the answers provide useful computation for the puzzle issuer
- Puzzle answer must be easily verifiable

IXP Implementation

OGI SCHOOL OF SCIENCE & ENGINEERING
OREGON HEALTH & SCIENCE UNIVERSITY

http://www.cse.ogi.edu/sysl/

Funded by: intel, National Science Foundation