Demystifying Service Discovery: Implementing an Internet-Wide Scanner

Derek Leonard

Joint work with Dmitri Loguinov

Internet Research Lab
Department of Computer Science and Engineering
Texas A&M University, College Station, TX USA

November 1, 2010

Agenda

- Introduction
- Service Discovery
  - Formalizing Politeness
  - GIW Algorithms
- Evaluation
  - Experiments
  - Feedback Analysis
- Conclusion

Introduction I

- Techniques for quickly discovering available services in the Internet benefit multiple areas
  - Characterizing Internet growth (# hosts, # servers)
  - Discovering/patching security flaws (DNS, SSH)
  - Understanding how worms create massive botnets
  - Distance estimation
- Several large-scale studies of the past describe potentially significant drawbacks
  - Long durations for individual tests (i.e., months)
  - Significant number of complaints
  - Sensitive TCP ports avoided due to negative publicity

Introduction II

- This paper chronicles our development of IRLscanner, an Internet-wide service discovery tool that addresses these drawbacks
- We propose the following objectives
  - Maximize politeness at remote networks
  - Allow scanning rates that cover the Internet in minutes/hours
- We then perform 21 varied Internet-wide scans
  - Experiments span multiple ports, protocols and options
- Analysis of feedback generated demonstrates that similar studies are feasible in the future

Agenda

- Introduction
- Service Discovery
  - Formalizing Politeness
  - GIW Algorithms
- Evaluation
  - Experiments
  - Feedback Analysis
- Conclusion

Service Discovery

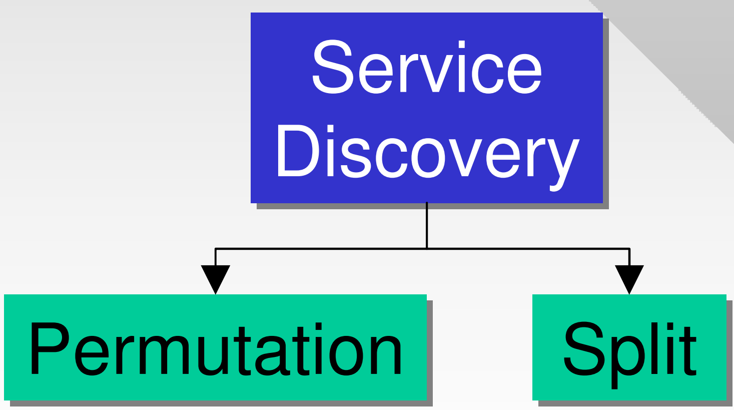
- Definitions:
  - Assume there are m local machines
  - In some set $\mathcal{F}$ there are $n = |\mathcal{F}|$ targets
- Service Discovery: Requests from local hosts are sent to targets in $\mathcal{F}$, which are marked as alive if they respond
  - We focus on techniques for horizontal scanning
- Let T be the time required to fully probe $\mathcal{F}$
  - Total Internet-wide sending rate is n/T pkts/sec
- Assume that $\mathcal{F}$ consists of all IPv4 addresses
  - In the paper non-routable addresses are omitted

Agenda

- Introduction
- Service Discovery
  - Formalizing Politeness
  - GIW Algorithms
- Evaluation
  - Experiments
  - Feedback Analysis
- Conclusion

Formalizing Politeness I

- Formal analysis of algorithms for service discovery has not previously been attempted
- To start, we propose two major components of service discovery and study each separately



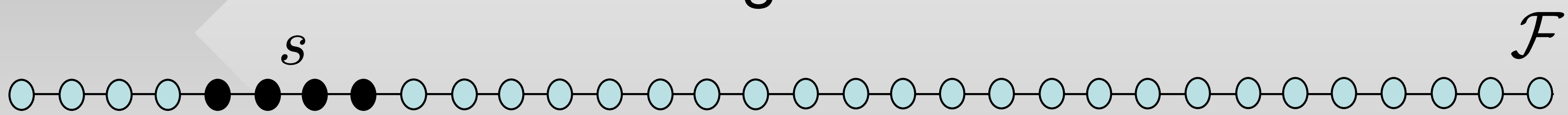
- Permutation: Order in which hosts in $\mathcal{F}$ are targeted
- Split: Method for dividing targets in $\mathcal{F}$ among m local machines

Formalizing Politeness II

- Choices made for permutation and split algorithms heavily impact:
  - Denial-of-service effects on target networks
  - Complaints to local network administrators
  - Number of firewalls blocking our traffic/network
- Researchers should minimize these effects when they undertake service discovery
- Previous work suggests that existing approaches exhibit prohibitive negative effects
  - We sought to design maximally polite techniques to fill this gap

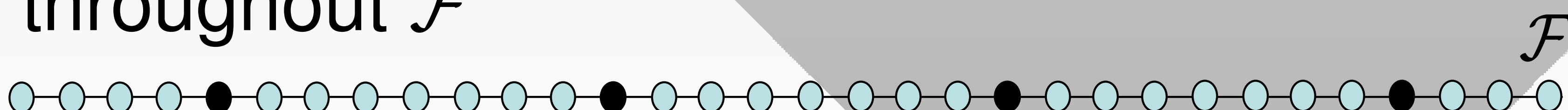
Formalizing Politeness III

- We define the concept of maximal politeness by first considering a single subnet s
  - Subnet: Block of contiguous IP addresses in $\mathcal{F}$



- Key observation: Bursts of traffic (i.e., high instantaneous load) to s trigger negative effects and must be minimized

- Permutation Goal: Spread probes to s evenly throughout $\mathcal{F}$



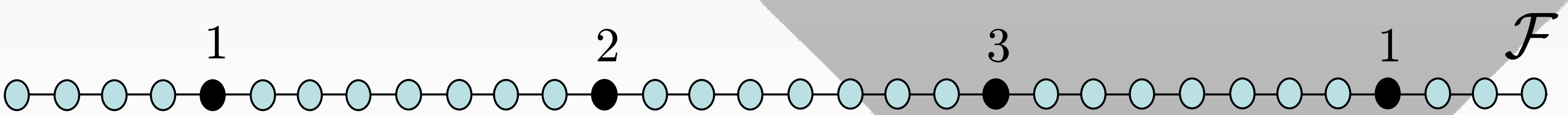
  - Any permutation that returns to s with a period $n/|s|$ we define as IP-wide at s

Formalizing Politeness IV

- Define Globally IP-wide (GIW) to be a permutation that is IP-wide at *all* subnets
- Assumption: Subnet boundaries and their actual sizes are not explicitly known
  - Sizes are powers of 2 however
- Observations about GIW
  - All networks are probed at constant rate $|s|/T$ proportional to their size
  - All s have the maximum inter-probe gap given T
- Next: Split that maintains GIW permutation

Formalizing Politeness V

- Intrusion Detection Systems (IDS) detect scan traffic to alert administrators of attacks
  - Detection is often based on the number of packets received by individual source IP addresses
- Key Observation: Repeated probes from a single local host trigger IDS more frequently and lead to firewall blocks and complaints
- Split Goal: Each local host should return to s *only* after all other local IPs have probed s



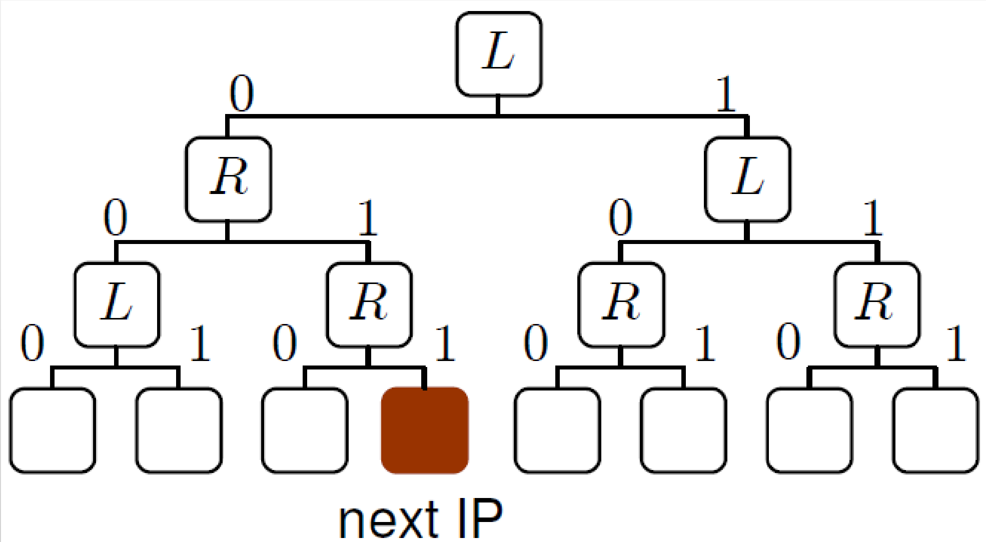
  - Individual IPs return to s with period $mn/|s|$

Agenda

- Introduction
- Service Discovery
  - Formalizing Politeness
  - GIW Algorithms
- Evaluation
  - Experiments
  - Feedback Analysis
- Conclusion

GIW Permutation I

- Start with permutation
- Alternating Gateway Tree
  - Binary tree of depth 32
  - Edges labeled with 0/1 bits
- Scanner traverses the tree to generate an IP
  - Accumulates bits along edges
  - Left or right traversal is determined by node state
- State flipped at each visit
  - No IP visited twice
  - Packets alternate between children at each node



Last 4 levels of a random AGT
Next IP ends with bits 011

- 2^{n-1} possible permutations
- Overhead
  - 512MB in RAM and for checkpointing on disk
  - 32 reads/writes (64 total) per IP generated

GIW Permutation II

- An alternative algorithm is desirable when the overhead required by AGT is not feasible
- Observation: If subnet s has depth b in the AGT, there are $n/|s|=2^b$ subnets of size $|s|$
  - GIW permutations must visit all remaining 2^b-1 subnets at depth b before returning to s
- To achieve this, a permutation must exhibit a full period in the upper b bits of the IP address
- Implication: A full period must be maintained in the upper b bits at every depth $1 \leq b \leq 32$ for a permutation to be GIW

GIW Permutation III

- By reversing the bits in each IP, the condition becomes much simpler
  - The full period must hold in the *lower* b bits
- Goal: Find a sequence of integers with full periods for all lower b bits of the integer, where $1 \leq b \leq 32$
- Reversing the bits of this sequence yields a GIW permutation of IP addresses
- Proof is in the paper

GIW Permutation IV

- An LCG of the form $x_k = ax_{k\text{-}1} + c$ is suitable
  - Requires only a single integer of state
  - Subsequent IPs can be calculated very quickly
  - Maintains a full period in all lower b bits when $a-1$ is divisible by 4 and c is odd (well-known result)
- We call this algorithm Reversed LCG (RLCG)
  - With constants $a = 214{,}013$ and $c = 2{,}531{,}011$, it produces uncorrelated random variables
- Initial seed x_0 can be used to change the scan order across multiple runs

GIW Split I

- Recall that in our desired split, individual local IPs return to s with period $mn/|s|$
  - Alternate in some order with full period m
- Round-robin (RR): Generate a single RLCG permutation $\{z_k\}$ and assign target z_k to host k mod m
  - IP addresses in the RLCG sequence are assigned in a round-robin fashion to local scanning hosts
- However, RR only achieves the desired split under certain conditions for m

GIW Split II

- Based on well-known properties of LCGs, we obtained the following result
- Theorem: RR-split with any GIW permutation scans s with $\min(|s|, m_s)$ sources, where

$$m_s = \frac{m}{\text{gcd}(\frac{n}{|s|}, m)}$$

- Odd m produces $m_s = m$ (i.e., a full period)
  - Even m leads to $m_s \leq m/2$
- Final Result: RLCG/RR with odd m produces a GIW split at every network s

Agenda

- Introduction
- Service Discovery
  - Formalizing Politeness
  - GIW Algorithms
- Evaluation
  - Experiments
  - Feedback Analysis
- Conclusion

Evaluation

- Internet-wide service discovery projects are sparse in the literature
  - Only 4 papers have described such projects
- Time and resources were major constraints
  - Single measurements took months to complete
  - Often several hosts were required
- Overwhelming number of complaints caused researchers to abort desired measurements
- Goal: Demonstrate that service discovery is viable by performing a variety of measurements, then analyze blowback

Agenda

- Introduction
- Service Discovery
  - Formalizing Politeness
  - GIW Algorithms
- Evaluation
  - Experiments
  - Feedback Analysis
- Conclusion

Experiments I

- Performed 21 Internet-wide measurements
  - Custom scanner described in detail in the paper
  - Fastest scans used $T = 24$ hours and a single host with local IPs aliased to the same network card
- Each target address is classified into one of four categories depending on how it responds
  - Open set $\mathcal{O}$: Hosts that responded positively (e.g., SYN-ACK to a TCP SYN)
  - Closed set $\mathcal{C}$: Hosts that responded negatively (e.g., TCP RST to a SYN packet)
  - Unreachable set $\mathcal{U}$: Destination unreachable error
  - Dead set $\mathcal{D}$: No response received
  - Note: $\mathcal{O} \cup \mathcal{C} \cup \mathcal{U} \cup \mathcal{D} = \mathcal{F}$

Experiments II

Name	Proto	Port	Type	Date	T	m	$\|\mathcal{O}\|$	$\|\mathcal{C}\|$	$\|\mathcal{U}\|$	pps	Mbps
DNS_1	UDP	53	DNS A	2-21-08	30d	1	15.2M	–	148M	709	0.48
DNS_2		53	DNS A	3-25-08	6d	5	15.2M	–	155M	3.5K	2.38
DNS_3		53	DNS A	5-07-08	1d	31	14.7M	–	168M	21.2K	14.28
DNS_4		53	DNS A	5-19-08	1d	31	14.5M	–	169M	21.2K	14.28
DNS_5		53	DNS A	5-20-08	1d	31	14.6M	–	168M	21.2K	14.28
DNS_6		53	DNS A	5-21-08	1d	31	14.5M	–	167M	21.2K	14.28
DNS_7		53	DNS A	5-22-08	1d	31	14.5M	–	169M	21.2K	14.28
ECHO		7	–	7-01-08	1d	31	322K	–	170M	22.1K	21.03
PING	ICMP	–	echo	6-24-08	1d	31	139M	–	99M	22.1K	14.85

- Received 30% more DNS replies than a similar study performed recently
  - 4.4M DNS servers responded to every scan
- ECHO has never been targeted in the literature
  - Useful for complaint analysis as ECHO is notoriously exploited by attackers for denial-of-service
- ICMP ping scan discovered 20% more responsive hosts than a recent study

Experiments III

Name	Protocol	Port	Type	Date	T	m	$\lvert\mathcal{O}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{U}\rvert$	pps	Mbps
SMTP_S	TCP	25	SYN	7-30-08	2d	61	17M	87.1M	119M	11.2K	7.55
SMTP_A		25	ACK	7-30-08	2d	61	–	116M		11.2K	7.55
EPMAP_S		135	SYN	8-05-08	2d	61	4.9M	40.2M	127M	11.3K	7.58
EPMAP_A		135	ACK	8-05-08	2d	61	–	68.4M		11.3K	7.58
HTTP_1		80	SYN	7-17-08	1d	123	30.3M	49.1M	78M	22.6K	15.19
HTTP_2		80	SYN	8-05-09	1d	61	44.3M	61.3M	97.1M	24.4K	16.39
HTTP_3		80	SYN	8-06-09	1d	61	44.0M	61.2M	85.1M	24.2K	16.26
HTTP_4		80	SYN	8-10-09	1d	123	44.2M	61.5M	94.7M	24.4K	16.39
HTTP_5		80	SYN	8-24-09	2d	123	44.5M	61.7M	96.4M	12.1K	8.15
HTTP_6		80	SYN	8-27-09	1d	61	44.1M	61.4M	80.7M	24.4K	16.37
HTTP_{AS}		80	ACK→SYN	9-02-09	1d	61	31.7M	49.6M	92M	25.8K	17.35
HTTP_{OPT}		80	SYN+OPT	7-15-10	1d	121	37.8M	48.1M	71.3M	26.3K	20.70

- SMTP (email) and EPMAP (reconnaissance) have not been scanned in the literature
- Combination ACK and SYN scans can be used to classify remote firewalls
- The final scan measures the deployment of several TCP options (for details see the paper)

OS Fingerprinting I

- Information about responsive hosts in open set $\mathcal{O}$ is often critical to the depth of studies
  - Fingerprinting: Use distinguishing characteristics of network traffic to infer interesting information
- Operating System (OS) is an important metric
  - Estimate the global impact of known vulnerabilities
  - Approximate Internet-wide market share
- Internet-wide OS fingerprinting has not been attempted in the literature
  - We use a technique that requires no additional sent packets (relies on TCP retransmission timeouts)
  - All code and data are publicly available (see paper)

OS Fingerprinting II

- We applied the technique to scan HTTP_2
  - Fingerprinted 39.6M servers
- General purpose hosts dominated the set at 82%
  - Machines that primarily host web sites
- "Market share" of web hosts given in the second table
  - 5.6% of Windows are Windows 2000 or earlier

Device Type	Found	%
General purpose	32.4M	81.8
Network device	2.7M	6.8
Printer	1.8M	4.6
Networked storage	1.5M	3.7
Media	929K	2.3
Other embedded	287K	0.7
Total	39.6M	

Categorized IPs

OS Class	Found	% of GP
Windows	16.3M	50.2
Linux	13.0M	40.2
BSD/Unix	2.2M	6.7
Mac	862K	2.7

General purpose devices broken down by OS class

Agenda

- Introduction
- Service Discovery
  - Formalizing Politeness
  - GIW Algorithms
- Evaluation
  - Experiments
  - Feedback Analysis
- Conclusion

Feedback Analysis I

- Email complaints are considered a strong deterrent
  - Bad publicity or legal threats
- We removed any network whose administrator complained
  - Blocking too many would render measurements useless

Service	Scans	Emails	Avg	IPs excluded	Avg
DNS	7	45	6.4	3.7M	530K
Echo	1	22	22	752K	752K
Ping	1	4	4	1K	1K
HTTP	8	27	3.4	459K	57K
SMTP	2	6	3	262K	131K
EPMAP	2	2	1	65K	32K
Total	21	106	5.05	5.3M	250K

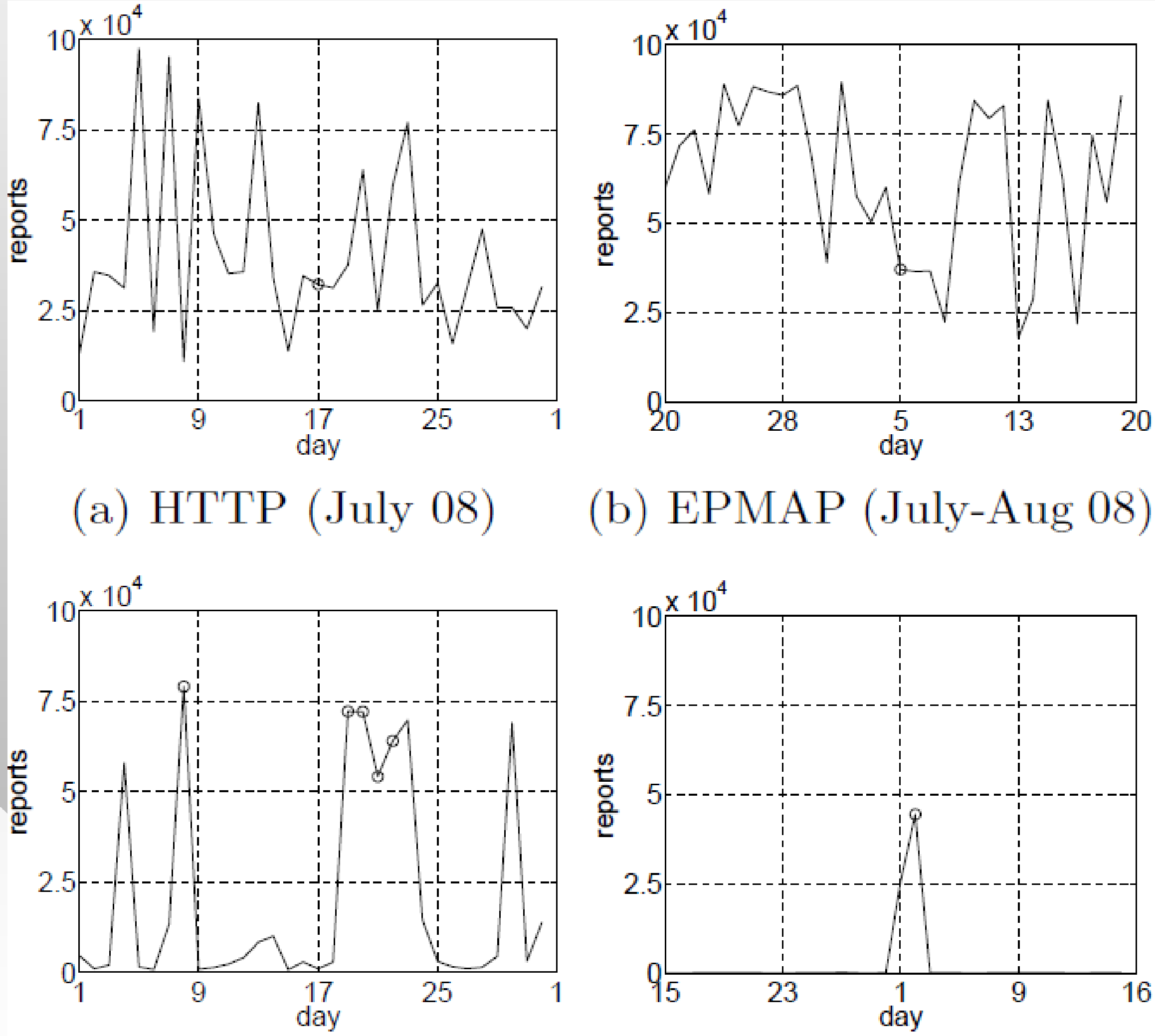
- TCP scans averaged 3 emails
  - Stark contrast to previous work
- Sensitive services did not lead to more complaints
  - Three legal threats, none credible
- 0.23% of routable space blocked
- Even with small T, email complaints are manageable

Feedback Analysis II

- Many administrators share firewall/IDS logs in online collaborative systems
  - Allows for a broader view of Internet-wide attacks
  - Example: SANS Internet Storm Center (ISC)
- Only suspicious packets are reported to ISC
  - Publicly lists IP address of scanners by service
  - Summary statistics are calculated daily
- These reports can be used to gain insight into how scans were perceived
  - We downloaded the number of daily targets for 30 days surrounding each scan

Feedback Analysis III

- HTTP and EPMAP regularly experience high load
  - Our traffic blended in
  - Fewer email complaints
- DNS and ECHO were scanned less often
  - Our traffic caused spikes
  - ECHO received most complaints
- Administrators are more concerned with traffic anomalies than sensitive services



(a) HTTP (July 08)
(b) EPMAP (July-Aug 08)
(c) DNS (May 08)
(d) ECHO (June-July 08)

Agenda

- Introduction
- Service Discovery
  - Formalizing Politeness
  - GIW Algorithms
- Evaluation
  - Experiments
  - Feedback Analysis
- Conclusion

Conclusion

- More IRLscanner design features
  - Reduction in scan scope over previous methods
  - Absence of largely ineffective retransmissions
  - Accurate extrapolation in partial scans
- Other novel techniques
  - Method for finding average service uptime
  - Analysis of DNS back-scans
  - First Internet-wide measurement of TCP options
  - ACK scans to bypass stateless firewalls
- See the paper for more details and information