The Windows of Private DNS Updates

Andre Broido∗, Hao Shang, Marina Fomenkov, Young Hyun, kc claffy
CAIDA, SDSC, {broido, hao, marina, youngh, kc}@caida.org

ABSTRACT

This work is motivated by the observation of one particular type of unwanted traffic – dynamic DNS updates for private (RFC1918) addresses, which leaks to global network. This spurious traffic not only wastes network resources but also jeopardizes security and privacy of users.

We first look at the magnitude of these updates on two independent AS112 [1] servers. We then analyze which operating systems are responsible for these updates by using three levels of signature techniques and find that over 97% of updates come from Windows systems. While newer versions of Windows OSes are more stringent in sending private DNS updates, we did not observe an overall decreasing trend due to this evolution. Users, software vendors, and system administrators can take steps to reduce this RFC1918 traffic. However, since most end users are unlikely to interfere with vendor default settings, it should be the responsibility of software vendor and system administrators to take positive action to fix this problem.

Categories and Subject Descriptors

C.2.2 [**Network protocols**]: Applications—DNS; C.2.3 [**Network operations**]: Network management; C.2.5 [**Local and Wide-Area Networks**]: Internet; D.4.0 [**Operating Systems—General**]: Windows

General Terms

Measurement, Management, Standartization, Human Factors

Keywords

private addresses, Domain Name System, dynamic updates, misconfiguration, OS fingerprinting

1. INTRODUCTION

In this paper we study one particular category of unwanted traffic: erroneous attempts to update address-to-hostname mappings (PTR records) for private RFC1918 [2] addresses in the global Domain Name System (DNS). In previous work [3] we observed an astonishing number of such polluting DNS updates at an authoritative server for RFC1918 addresses that is located near F-root in Palo Alto, California. These updates averaged at several hundreds per second and reached their peak at this root server instance at 1300 per second in November 2002.

This spurious RFC1918 traffic leaves a window wide open into a user's private realm. Hosts sending these updates leak information

---

∗Andre Broido and Hao Shang are now at Google, Inc.

about the time they reboot or connect to the Internet, and continue to inform the outside world about their presence periodically. Furthermore, the users run the risk that their private and perhaps proprietary data – names, IP addresses, numbers of nodes on internal networks – will become visible to third parties, who can use it for hacking or social engineering attacks. Since it is perfectly legal to announce the AS112 block (192.175.48.0/24) from any location, it means that access to such user information is granted to everyone with a BGP router.

Unfortunately, users are often unaware of the fact that their machines are misbehaving, and that their internal information is exposed to outside parties. The net result is not only a waste of Internet resources but also a multitude of security, privacy and intellectual property concerns [4].

The contribution of this study is three-fold. First, we examine the magnitude and trend of the RFC1918 DNS updates by analyzing logs collected at two independent locations. Second, we determine the current prevalence of Windows as a source of such updates using three signature techniques. Third, we verify our findings by checking the default Windows setting in a lab controlled environment and explore measures to reduce the leakage of the RFC1918 DNS updates to the global Internet.

2. PRIVATE DNS UPDATES AND THE AS112 PROJECT

In 1996 the Internet Assigned Numbers Authority (IANA) specified three address blocks for use in private networks without coordination with an Internet registry. These so-called RFC1918 [2] addresses are: 10/8 with 16.8 million addresses, 172.16/12 with 1 million addresses, and 192.168/16 with 65,536 addresses.

The Dynamic Host Configuration Protocol (DHCP) [5, 6] and the dynamic updates in the domain name system [7] are ubiquitously used for reuse and conservation of IP addresses within a given network. DHCP allows an IP address to be assigned dynamically. Dynamic DNS updates allow the mapping information between the IP address and a domain name to be updated dynamically in the authoritative server for the corresponding zone. Configuration inconsistency between DNS servers and DHCP clients and servers leads to the polluting situation: a DNS update generated by a DHCP client or server for RFC1918 space is inappropriately directed to a global DNS server.

Table 1 illustrates the logical steps of sending DNS updates for both forward and inverse mappings between a domain name and an IP address. In the first two steps, a DHCP client obtains a private IP address from a DHCP server. Next, the DHCP client tries to update the forward mapping from the domain name (*hostname.ex.com* in our example) to its newly obtained IP address (type A RR). It first sends a query to its local domain name server (LDNS) and asks

for the authoritative server for the zone of its domain name (step 3). Once it receives a response (step 4), it sends the update to the indicated server (step 5). Similarly, steps 6-8 are for the update of the inverse mapping from the IP address to the domain name (type PTR RR). In the correct setup, the LDNS should point the DHCP client to a domain name server (could be itself) inside the internal network. However, in many cases when the DHCP and DNS setups are inconsistent, the LDNS may direct the DHCP client to a place outside the enterprise, resulting in leakage of private DNS updates to the global network. In the example shown, the LDNS is not configured with a local zone for 168.192.in-addr.arpa. The LDNS thus iteratively sends the SOA request starting with a root DNS server and eventually returns the *prisoner.iana.org* as an authoritative server to the client (step 7).

Table 1: Logical steps of sending DNS updates (not all packet exchanges are shown)

	From	To	Content
1	DHCP Clt	Broadcast	DHCP Request
2	DHCP Srv	DHCP Clt	DHCP ACK: 192.168.0.2
3	DHCP Clt	LDNS	Query: SOA? hostname.ex.com
4	LDNS	DHCP Clt	Response: SOA dns.ex.com
5	DHCP Clt	dns.ex.com	Update:A hostname.ex.com
6	DHCP Clt	LDNS	Query:SOA? 2.0.168.192.in-addr.arpa
7	LDNS	DHCP Clt	Response:SOA prisoner.iana.org
8	DHCP Clt	prisoner	Update:PTR 2.0.168.192.in-addr.arpa

Before the AS112 project [1] began, there were no delegated DNS servers for RFC1918 address space and root servers themselves have to handle DNS queries and updates in that address space. As the imposed load on root servers increased [8], the AS112 project was launched and deployed delegated servers to deflect the RFC1918 updates from the root servers. Each instance of the AS112 project includes three servers: prisoner.iana.org for handling RFC1918 updates (as in the above example) and blackhole-1.iana.org, blackhole-2.iana.org for handling RFC1918 queries. Each of these servers uses an anycast [9] address. Messages are directed to a particular physical server based on the BGP path selection at the intermediate routers.

3. OBSERVATIONS OF RFC1918 UPDATES

We examined more than three years of RFC1918 updates logged at AS112 servers in Palo Alto, California (topologically close to an F-root node) and Osaka, Japan (near an M-root node) between October 2002 and January 2006. Figure 1 presents counts of DNS updates per hour at both locations. The number of packets exchanged between the server and a client in each update transaction varies depending on the protocol used. Each UDP update involves one packet sent to the server and one packet returned. However, some updates follow a UDP exchange with a TCP one, which a) involves five packets sent to the server and four packets returned, and b) repeats up to three times. In this case one update corresponds to 29 packets.

The Palo Alto server's hourly update rate stays in the 0.5–0.7 M range for most of 2003 and the first eight months of 2004. It jumps to 1 M on August 25, 2004 and reaches 3M at the end of 2004, exceeding the server's logging capacity and causing apparent oscillations due to the loss of some log entries. In 2005 the update rate generally is within the 1.5–2.5M range, except for a few big oscillations at the end of October. Finally, in December 2005, the update counts suddenly drop by a factor of two.

The Osaka AS112 server's update rate jumps twice in the first half of 2003. It then displays slow but steady growth until mid-2004, hovering around 1M updates/hour. In October 2004 it drops
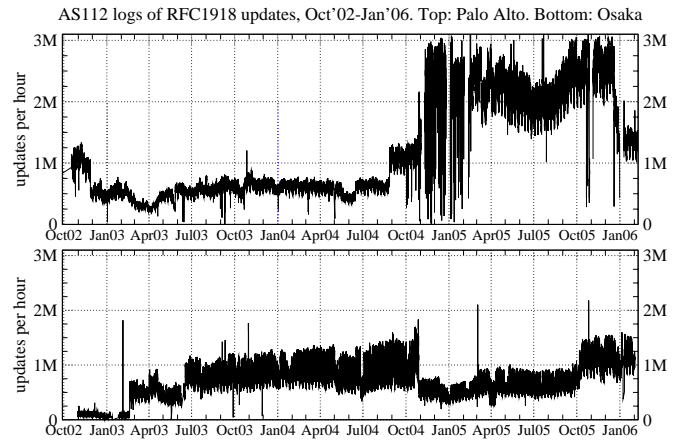

AS112 logs of RFC1918 updates, Oct'02-Jan'06. Top: Palo Alto. Bottom: Osaka

Figure 1: Rate of RFC1918 DNS PTR updates in Oct'02-Jan'06

down to its February 2003 level of about 0.5M updates/hour and than jumps back to the 1M updates/hour in October 2005.

The rate jumps in August 2004 and December 2005 at Palo Alto are caused by route changes. First, these abrupt load shifts occurred on the time scale of seconds, much faster than the time scale of the diurnal effects described in [3]. Second, the number of unique prefixes and ASes originating updates changes in proportion to the rate. For example, the average update rate for the day after the jump on 08-24-04 at the Palo Alto server increased by 68% compared with the day before. At the same time, the numbers of unique prefixes and ASes observed in the data increased by 62% and 72% respectively. These facts imply that the observed sharp stepwise changes in the rate of RFC1918 updates occur when a large chunk of IP addresses is rerouted from one instance of the anycast server to another.

The large volume of DNS RFC1918 traffic observed at two independent locations indicates that this type of spurious traffic is a global phenomenon rather than a regional problem. Our data is highly topologically diverse in terms of sources. Just three days of name server logs from Palo Alto prisoner contain updates coming from from 1/6 of routed prefixes and from almost 30% of ASes.[1] Even our 5-minute packet trace data sources (Sect. 4) include 10% of prefixes and 20% of ASes.

The AS112 servers shield the root servers from an assault of misguided traffic, but they do not solve the underlying problem. Traffic bound for AS112 prefixes is a network waste product, having no value to either the sender or any possible recipient. Networks who emit traffic toward AS112 destinations are the moral equivalent of a driver who throws empty styrofoam cups out of a moving car, except in this case the driver is paying transit providers for the privilege of removing this trash and then delivering negative acknowledgments for each piece thereof. Note that the RFC1918 traffic squanders considerable bandwidth. We observed that the inbound traffic to the AS112 server at Palo Alto peaked at 15.3Mbps (averaged at a 1-minute interval) on Feb 14, 2005. Moreover, the nature of anycast addresses used by the AS112 servers allows any party with a BGP router to collect RFC1918 traffic which exposes user private information such as names, IP addresses, OSes of their running machines.

---

[1]We use *semiglobal* [10] prefixes—those visible in Route Views [11] Mar 17 2005 12:00 snapshot.

4. IDENTIFICATION OF OSES PRODUCING THE RFC1918 UPDATES

In this section we analyze prisoner's inbound traffic in order to identify classes of operating systems that originate the RFC1918 DNS updates. We use a three-pronged approach that relies upon data from three different layers: *application* – via DNS payload; *transport* – via passive OS fingerprinting; and *network* – via IP TTL statistics.

We analyzed two 5-minute full packet traces collected at different times from the Palo Alto AS112 servers. These traces include inbound and outbound traffic to both prisoner and the two blackhole servers. As prisoner handles all RFC1918 updates, we only focus on inbound traffic to this server, which contributes about 70% of the total inbound traffic in both logs. The traffic volume in February 2006 is about 50% lower than in March 2005. We attribute this decrease to a route change on Dec 17, 2005 (cf. Figure 1). The number of unique source IPs, prefixes, and ASes seen in both traces differs by about the same percentage.

Table 2: Description of the inbound traffic to prisoner at Palo Alto

Date	Packets	TCP%	UDP%	SrcIPs	Prefixes	ASes
03-17-05	1.65M	89.5%	10.5%	69133	11954	2685
02-01-06	0.81M	86.7%	13.3%	37823	6314	1357

4.1 Application layer: DNS payload

The preferred transport protocol for DNS is UDP [12], but nearly 90% of `prisoner`'s inbound packets use TCP (Table 2). The TCP protocol is used by secure update attempts after the failure of a UDP update. In each TCP connection including five inbound packets, there is only one DNS message which includes a single TKEY resource record (RR) [13] to establish shared secret keys between a DNS server and a resolver.

In order to identify the OS origins of the TCP packets, we examine three fields in the TKEY RR and the location of the record in the DNS message. The fields used are query name, algorithm name, and key data. In our controlled lab environment, we found that the content of these fields and the record location are implementation-dependent for Windows 2000 SP4, Windows XP Professional SP2, and Windows Server 2003 SP1 (the top half of Table 3).

Table 3: Application-level signatures for Windows systems. We use the regular expression syntax of the *perl* language to describe the name patterns.

		Win2K	Win XP	Win2003
T	Query Name	`\d{12,13}-\d`	`\d{3,4}-ms-[0-9a-f-]*`	
C	Alg. Name	gss.microsoft.com	gss-tsig	
P	Key Data	NTLMSSP	NTLMSSP	
	TKEY Loc.	Answer Section	Additional Section	
U	RR Counts	1/2/0	0/1/0	
D	RR Types	cname.none/ptr.any/ptr.in	ptr.in	
P	TTL	20min	15min	

The query name field is highly implementation-dependent and only needs to be locally unique at the resolver. Query names coming from Windows 2000 systems have a 12 or 13 digit string followed by a dash and a single digit. The query names generated by Windows XP or 2003 server are made of a leading 3 or 4 digit string followed by "-ms-" and an arbitrary number of hexadecimal digit strings concatenated by "-".

The name of an algorithm determines how the TKEY RR is actually used to derive the algorithm specific key. This algorithm can be vendor specific or generic. For example, the name "gss.microsoft.com"

used by Windows 2000 is a vendor-specific name, while "gss-tsig" looks more generic. However, both algorithm names refer to Microsoft's GSS-TSIG method of securing dynamic DNS update transactions between a client and a server. A Microsoft team authored the Internet standard for GSS-TSIG (RFC 3645 [14]). We do not expect hosts running other OSes to refer to Microsoft's algorithm in key establishment negotiation, unless they deliberately masquerade.

The last indicative field is the key data field. We find the string NTLMSSP in the key data field for all Windows versions. The string stands for NT LAN Manager Security Support Provider, which is a component introduced in Windows NT 4.0 [15]. The presence of this vendor specific string is a strong indication of Windows OSes.

We also examine the location of the TKEY RR in the DNS message. The Windows 2000 OS places the TKEY RR in the Answer section, which violates an RFC 2930's requirement for the TKEY record to be in the Additional section. The DNS messages generated by Windows XP and Server 2003 conform to this requirement.

We combine all signatures listed in Table 3 to identify DNS messages sent by Windows 2000, XP, or Windows Server 2003. We realize that this identification method could yield a false-positive identification when DNS messages generated by other OSes would follow the same patterns and, hence, would be identified as Windows OSes. However, we rely upon not one, but a few vendor specific signatures. Chances are low for other implementations to follow exactly all the patterns except in the case of an intended masquerade. Under this assumption, we examined the two available packet traces from Table 2. Except for the flows that do not have a valid DNS message, practically all TCP flows are coming from Windows systems with Windows 2000 being by far the most prevalent system (Table 4).

Table 4: OS origins of TCP flows

Date	TCPFlows	NotDNS	Win2K	WinXP/03	Others
03-17-05	297271	1.4%	96.9%	1.5%	0.02%
02-01-06	140688	1.1%	94.4%	4.5%	0.03%

Using the same methodology, we investigated Windows signatures in UDP updates (the bottom half of Table 3). For Windows 2000 and XP systems, a UDP DNS update includes one prerequisite RR and two update RRs.[2] The TTL field in the last update RR is set at 20 minutes. In contrast, a UDP DNS update generated by Windows 2003 server includes only one update RR with TTL=15 min, which requires adding the included PTR record without prerequisite or any other actions. Again, we are aware that these signatures may not be unique for Windows systems. But these patterns are all implementation-dependent since the standard for DNS updates [7] does not specify the set of RRs for an update, nor does it suggest any particular TTL values. As we use the combination of all the three patterns to identify Windows systems, we believe the rate of false positives is negligible.

Table 5 lists the percentages of Windows systems for UDP updates. In both traces, only 2-3% of all UDP updates are from non-Windows systems. The 2005 trace shows that most updates come from Windows 2000 or XP systems, while in the 2006 trace more UDP updates are generated by the Windows 2003 Server.

Finally, we identify the operating system for each unique IP

---

[2]The prerequisite RR with CNAME type and NONE class indicates that the update requires no previous existence of the canonical name RRs for the to-be-updated name. The two update RRs order to remove all existent RRs for the same name and type before adding the new one.

Table 5: OS origins of UDP updates

Date	UDP Updates	Win2K/XP	Win2003	Others
03-17-05	171994	57.6%	39.2%	3.2%
02-01-06	108146	43.6%	54.1%	2.4%

address seen in our traces by combining the patterns of the TCP TKEY messages and of the UDP DNS updates together. We observed that all TCP TKEY messages (except those influenced by the trace boundary conditions) follow a UDP update.[3] Windows 2000 systems generate three TKEY messages in an attempt to send a secure update after the initial UDP update was refused [16]. But Windows XP and Windows 2003 systems try secure updates only if they are configured with Active Directory-integrated zones [17, 18]. With this knowledge in mind we examined all TCP and UDP messages sent by each unique IP and were able to separate the three Windows OS flavors.

In Figure 2, we subdivide the observed IP addresses into five sets: Windows 2000, Windows XP, Windows 2003, mixture, and unknown. The first three sets include IP addresses identified as sources of a single type of Windows OS. The mixture category includes IP addresses that represent multiple types of OSes including at least one Windows OS. The unknown category includes IP sources that did not generate any messages with Windows signatures. The first bar drawn for each set represents the percentage of unique IP addresses in this set. The second and the third bars are the percentages of TCP and UDP updates, correspondingly, generated by sources in each set relative to the total number of updates.

In both the 2005 and 2006 traces, Windows 2000 is the dominant OS that generates RFC1918 updates and also accounts for most TCP messages. Windows XP and 2003 rarely send TCP updates, but generate a considerable amount of UDP updates. Compared with year 2005, the 2006 trace shows a higher percentage of IP addresses and updates from Windows XP and 2003 OSes. Combining all these Windows OSes and the mixture category together, we observe that over 99.5% of IP addresses represent a Windows host or at least have one Windows host behind it. By considering the percentages of TCP and UDP packets in the traces (Table 2) and the shares of Windows OSes in each category (Tables 4 and 5), we calculate that 98.2% and 98.7% of total packets are from Windows systems for the 2005 and 2006 traces respectively.
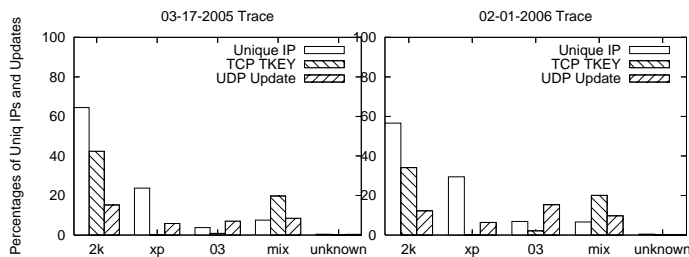
Figure 2: OS split of unique IP addresses and TCP/UDP messages

4.2 Transport layer: Passive OS fingerprinting

We used the passive OS fingerprinting tool p0f by M.Zalewsky [19] to gain an independent confirmation that the sources sending

---

[3]The reverse statement that a UDP update is always followed by TCP TKEY messages is not true.

secure DNS updates are hosts running different versions of Windows operating systems.

The p0f tool, a *de facto* standard in the security community, identifies an OS by examining the header of the SYN packet used to establish a TCP connection. Different OSes set up different initial values of various packet fields such as window size, TTL, fragment bit, flags, and selected options. Note that p0f can only identify the OS origin of TCP flows and it does not distinguish among Windows 2000 and XP, or Windows Server 2003. We summarize the OS cross-section identified by p0f in our traces in Table 6.

Table 6: p0f identification of OS origins of TCP flows

Date	Total	Windows	Other	Unknown
03-17-05	297302	98.5%	0.7%	0.8%
02-01-06	140130	97.9%	0.7%	1.44%

Results in Table 6 and Table 4 are consistent. The small discrepancy in the total number of flows is caused by the trace boundary conditions since the two methods use different packets to identify a flow. In Table 4 we identify a flow by a non-empty TCP packet, while in Table 6 we do it by observing a SYN packet. p0f shows a higher percentage of unrecognizable TCP flows in the 2006 log. Our conjecture is that its signature database has not been updated with the latest OS updates.

4.3 Network layer: IP TTL statistics

In this section we use IP layer fingerprinting by the TTL field to obtain the OS breakdown for both TCP and UDP update sources. Our fingerprinting method is based on the fact that different OS classes have distinct default values of the initial IP TTL (iTTL) setting. According to the p0f documentation [19], non-Windows OSes set the iTTL to 255, 64 or 60, while Windows 2000 and XP in all their flavors set the iTTL to 128[4]. Microsoft documentation [20, 21] confirms that the DefaultTTL parameter for Windows 2000, NT 4.0 and XP is 128, and for NT 3.51 it is 32. These default values apply to all IP protocols, including TCP and UDP. Users may conceal OSes from outside spying by changing the default TTLs via editing the registry or using special firewall software. However, most Windows users never change the default iTTL since there is no explicit TTL entry in a vendor shipped copy of the registry and Microsoft documentation discourages using the Registry Editor. Therefore, an iTTL of 128 strongly suggests one of the most widespread OSes (Windows 2000 or XP).

We studied TTL statistics in CAIDA traces of generic IP traffic mix collected in December 2004 and January 2005 at an OC-48 link connecting San Jose and Seattle in a Tier 1 backbone. These traces are close in time, space, and topology (host ASes are directly connected) to the 2005 AS112 traces from Palo Alto. We analyzed the first SYN packet captured for every source IP, for the total of 666K IP sources. Using p0f, we identified that 86.4% of traffic are from Windows systems. This result is comparable to data from a popular technical web site in UK [22] that shows 90.3% of its visitors run various flavors of Windows OS. We also found a comparable value in Beverly's statistics [23], where a Bayesian classifier identifies 93% of sources in an Internet exchange trace as Windows. Our data confirm that the iTTL and OS type as determined by p0f are tightly coupled for Windows machines. The conditional proba-

---

[4]The only non-Windows devices known by p0f to set initial TTL to 128 are some Nortel and Alteon switches, Linux-based Dell PowerApp cache, PocketPC 2002 and Sega Dreamcast Dreamkey. The installed base of these devices is negligible, and they are unlikely to send dynamic DNS updates.

bility for a packet with iTTL=128 [5] being from a Windows system is 98.8%. This number represents the true positive rate of using iTTL=128 to indicate a Windows system.

Having a statistical baseline established, we examine the iTTL distribution for packets reaching AS112 server prisoner. For both 2005 and 2006 traces, the fractions of all inbound packets with iTTL=128 are 98.9% and 98.8% respectively. Assuming that the true positive rate for Tier 1 backbone and prisoner's traffic are equal, and that the true positive rate for UDP is the same as for TCP, we infer that the percentage of packets from Windows Systems is $98.8\% * 98.8\% = 97.6\%$.

Inferring Windows-generated traffic from iTTL=128 can produce false negatives. We will not recognize packets from sources with changed iTTLs as coming from Windows. Missing this group of packets causes an underestimate of Windows' fraction of the updates, which is acceptable since we seek a lower bound. The lower bound of 97.6% packets from Windows systems confirms the results we obtained in Section 4.1 and 4.2.

5. SUMMARY OF RFC1918 UPDATE BEHAVIORS BY OS

In Section 3, we showed that a large amount of RFC1918 updates hit the AS112 servers. By applying three different signature techniques in Section 4, we found that at least 97.6% of these updates come from Windows systems. We then set up a controlled environment and examined how these updates are generated by Windows systems with default settings. Based on these experiments, we propose potential solutions to avoid or reduce these updates.

We used three new machines shipped with pre-installed Windows 2000, XP, and Server 2003. We updated each machine with its latest server pack: Windows 2000 SP4, Windows XP SP2, and Windows Server 2003 SP1, respectively. We configured both the Windows 2000 and the XP machines as DHCP clients, and the Windows Server 2003 as a DHCP server, all connected in the same private subnet. The Windows Server 2003 was also configured with the network address translation (NAT) service, which allowed traffic from the internal private subnet to be routed to the Internet connected to another interface.

By default, Windows 2000 sends updates for both A (name-to-address) and PTR (address-to-name) records. The update for an A record is destined to the local DNS as we configure all Windows machines with local domain names. The PTR update is sent to prisoner since the local DNS server is not configured as the authoritative zone for the private address in use. Once the PTR update is refused by prisoner, Windows 2000 sends three TKEY messages, each in a separate TCP connection, in an attempt to make a secure update. This procedure is repeated after 5 min, 10 min, and 60 min thereafter. Due to this unconditional generation of PTR updates and aggressive retries, Windows 2000 accounts for most of the updates in Fig. 2.

Windows XP generates DNS updates less aggressively than Windows 2000. It sends PTR updates only if its A record update is accepted. If a local DNS server is not configured for dynamic updates, Windows XP will not generate PTR updates. If its A record update succeeds, Windows XP will generate a PTR record whenever it renews or refreshes its DHCP address lease.

When Windows Server 2003 is configured as a DHCP server, its default setting is to make dynamic DNS updates only when requested by its DHCP clients. However, we observed that even when its DHCP clients indicate not to do so, the Windows 2003 DHCP

server still sends a PTR update every hour for each of its DHCP clients. This propensity explains a large amount of the updates coming from relatively few Windows 2003 IP sources (cf. Figure 2).

Both Windows XP and Server 2003 rarely send TCP updates since they have to be within Active Directory-integrated zones in order to attempt secure DNS updates.

For Linux and FreeBSD machines using ISC's DHCP client and server [24], no PTR updates are sent by default and turning the feature on requires a special configuration on both DHCP client and server [25]. In addition, we are not aware of any home router (being a DHCP server) that sends dynamic DNS updates for its DHCP clients.

6. REDUCING RFC1918 UPDATES

We envision three ways to eliminate these polluting updates: user efforts, vendor efforts, and system administrator efforts. The most straightforward approach is to let end users disable dynamic DNS updates on their Windows machines (see [26] for instructions). This feature should be enabled only when the end user knows that the update will be sent to the right DNS server and accepted. However, given that the vast majority of users are unlikely to change the vendor settings, a better way to avoid the spurious traffic is for software vendors to make sure their defaults prevent such traffic.

Software vendors could either set the dynamic DNS updates off by default or require RFC1918 updates to be sent only to a destination address on the same network or to a server sharing a long portion of its name with the host. Windows XP, compared to its predecessor Windows 2000, reduces the flood of RFC 1918 updates by making PTR updates conditional on successful A record updates. This step significantly decreases the chances for a Windows XP machine to send PTR updates, but so far has not impacted the load on the AS112 servers due to the dominance of older systems. In addition, even if all client machines adopt Windows XP, the RFC1918 DNS updates will not disappear if local DNS servers are not configured properly.

Besides software vendor actions to stem RFC1918 updates, administrators of local DNS servers should be diligent to confine these updates within their local zones. For a DNS server and DNS updating agents under the same administrative control, such as in an enterprise network, an administrator can consistently enable or disable DNS updates on the server and clients. For situations when a DNS server and updating clients are under different controls, such as home machines using private IP space, but referring to a DNS server from an ISP, the administrator of the DNS server should be cautious when observing SOA queries for inverse RFC1918 IP addresses. These SOA queries used to find authoritative domain servers are a prerequisite for sending PTR updates (step 6 in Table 1). In that case, an administrator could create SOA RRs that point the DNS server to itself as the authoritative name server for inverse RFC1918 address zones. Alternatively, administrators of DNS infrastructure could support an AS112 server [1] in their network to keep this traffic from leaking onto the global Internet.

7. CONCLUSIONS

We analyzed a specific type of unwanted traffic – RFC1918 DNS updates. These RFC1918 DNS updates are not only a waste of network resources, but also pose a threat to privacy and security of users. By analyzing update logs collected at two independent AS112 servers, we find that leakage of DNS updates is a global problem and the update volume can exceed millions per hour.

---

[5]We assume that observed TTLs between 64 and 127 come from iTTL=128.

We applied three signature techniques to two short packet traces and found that over 97% of DNS updates come from Windows systems and over 99% of unique IP addresses in our traces have at least one Windows machine at or behind it. Windows 2000 accounts for the majority of the update packets while Windows XP is more conservative in sending PTR updates. However, our long term data did not reveal an obvious decreasing trend in RFC1918 update rates due to the evolution of operating systems.

Users, software vendors, and system administrators should all take steps to avoid or reduce generation of such spurious traffic. Since it is unrealistic to expect most end users to deal with reconfiguring the defaults on their system software, cleaning up such pollution will require that software vendors and administrators of local DNS servers take responsibility for the behavior of their systems and mitigate polluting behavior with appropriate actions.

Reducing RFC1918 DNS updates can also help reduce traffic at the root servers since a DNS update usually incurs a preceding SOA query (cf. Table 1). However, we did not explicitly examine to what extent RFC1918 updates are related to queries hitting blackhole servers or root servers. We will leave this research for our future work.

8. REFERENCES

[1] "AS112 Project Home Page," www.as112.net.

[2] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, "Address Allocation for Private Internets, RFC1918," February 1996.

[3] A. Broido, E. Nemeth, and kc claffy, "Spectroscopy of private DNS update sources," in *WIAPP*, 2003.

[4] S. Cheshire, B. Aboba, and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses," draft-ietf-zeroconf-ipv4-linklocal-07.txt, 23rd August 2002 www.potaroo.net/ietf/ids/draft-ietf-zeroconf-ipv4-linklocal-07.txt.

[5] R. Droms, "Dynamic host configuration protocol, RFC2131," March 1997.

[6] S. Alexander and R.Droms, "DHCP options and BOOTP vendor extensions, RFC2132," March 1997.

[7] P. Vixie, S.Thompson, Y.Rekhter, and J.Bound, "Dynamic updates in the Domain Name System, RFC2136," April 1997.

[8] N. Brownlee, kc claffy, and E.Nemeth, "DNS Measurements at a Root Server," Globecom 2001.

[9] "Root Server Technical Operations Association," www.root-servers.org.

[10] A. Broido, E. Nemeth, and kc claffy, "Internet expansion, refinement and churn," European Transactions on Telecommunications, 13, No.1, Jan-Feb 2002, 33-51.

[11] David Meyer e.a., "University of Oregon Route Views Archive Project, www.routeviews.org," .

[12] P. Mockapetris, "Domain names - implementation and specification, RFC1035," November 1987.

[13] D. Eastlake 3rd, "Secret Key Establishment for DNS (TKEY RR), RFC2930," September 2000.

[14] S. Kwan, P. Garg, J. Gilroy, L. Esibov, J. Westhead, and R. Hall, "Generic Security Service Algorithm for Secret Key Transaction Authentication for DNS (GSS-TSIG), RFC3645," October 2003, potaroo.net/ietf/idref/rfc3645.

[15] "Microsoft Security Bulletin MS01-008," 2003, www.microsoft.com/technet/security/bulletin/MS01-008.mspx.

[16] Microsoft TechNet, "Windows 2000 DNS," www.microsoft.com/technet/prodtechnol/windows2000serv/plan/w2kdns2.mspx.

[17] Microsoft TechNet, "How DHCP technology works," Mar 2003, www.microsoft.com/technet/prodtechnol/windowsserver2003/library/TechRef/8006f246-2029-4bad-b9f0-4f31a56b0590.mspx.

[18] Microsoft TechNet, "How to configure DNS dynamic updates in Wiindows Server 2003," Oct. 2005, support.microsoft.com/default.aspx?scid=kb;en-us;816592.

[19] M. Zalewsky, "The new p0f: 2.0.5," 2004, lcamtuf.coredump.cx/p0f.shtml.

[20] Microsoft, "TCP/IP and NBT config parameters for Windows 2000 or Windows NT," 2004, support.microsoft.com/kb/q120642.

[21] Microsoft, "TCP/IP and NBT config parameters for Windows XP," 2004, support.microsoft.com/kb/314053/EN-US.

[22] "W3 schools browser statistics," 2005, www.w3schools.com/browsers/browsers_stats.asp.

[23] R. Beverly, "A Robust Classifier for Passive TCP/IP Fingerprinting," in *PAM*, 2004.

[24] "Internet Software Consortium," www.isc.org.

[25] "Secure dynamic dns howto," 2002, ops.ietf.org/dns/dynupd/secure-ddns-howto.html.

[26] "How to Disable Dynamic DNS Updates on Windows Systems," www.caida.org/research/dns/disable_dns_updates.xml.