SPF/Sender-ID DNS & DDoS Threats

Operations Analysis and Research Center for the Internet

Douglas Otis
Doug_Otis@trendmicro.com
November 3, 2007

http://tools.ietf.org/html/draft-otis-spf-dos-exploit-01
http://tools.ietf.org/wg/dkim/draft-otis-dkim-tpa-ssp-01.txt

How SPF/Sender-ID uses DNS

- SPF/Sender-ID first passes initial parameters to a script parser
- SPF/Sender-ID stores scripts in TXT (16) or SPF (99) RRs
- SPF scripts might chain as many as 11 RRs using Redirect, Include, & Exp script based macro-expanded mechanisms
- SPF scripts may also contain any number of IPv4 or IPv6 inclusionary or exclusionary addresses using CIDR notation
- Each chained set may contain 10 macro-expanded mechanisms with domain name & CIDR overlays
- Mechanisms may also subsequently resolve A, AAAA, MX, r-PTR or test the existence of A or AAAA using the Exist macro
- Mechanisms for MX or r-PTR may require 10 additional DNS transactions each. (10 x 10 = 100)

Initial parameters supplied to SPF scripts

- IP Address of SMTP client
- EHLO host-name (when not for Sender-ID)
- MAIL FROM email-address (when not for Sender-ID)
- Sender-ID replaces MAIL FROM with either:
  - RFC 2822 Resent-Sender email-address
  - RFC 2822 Resent-From email-address
  - RFC 2822 Sender email-address
  - RFC 2822 From email-address
- Some propose also checking DKIM domains to limit possible signature replay abuse

Label Components created from SPF Macros

%{symbol | rev-label-seq | # of right-most labels | chars-into-“.” }

rev-label-seq = “r”

chars-into-“.” = “-” | “+” | “,” | “/” | “_” | “=”

# of right-most labels = 1 - 128

symbols =

- **s** = email-address or EHLO (initial parameter)
- **l** = left-hand side email-address (initial parameter)
- **o** = right-hand side of email-address (initial parameter)
- **d** = email-address domain or EHLO (initial parameter)
- **i** = SMTP client IP addr decimal octet labels (initial parameter)
- **p** = r-PTR domain with IP addr validated (not initial parameter)
- **v** = "in-addr" IPv4, or "ip6" if IPv6 (auto generated)
- **h** = EHLO host-name (initial parameter)

Estimating SPF exploit risks

A cached SPF record can leverage local-parts of a spam campaign that repeat beyond the negative caching period

DNS amplification would be 10:1 until the sequence is beyond the negative caching period, at which point the attack becomes free while spamming

DNS 2304 bit q + 2784 a = 5088 bits (example attack)

DNS 5088 bits x 100 = 508 kbits / email-address evaluation

Without consuming additional attacker's resources, each recipient evaluation could generate 508 kbits of DNS traffic targeting a victim domain not contained within the message

Message evaluations might occur 1-3 times per inbound system or MUA and may involve hundreds of recipients

Estimating SPF related DDoS Potentials

Each second:

- – An SMTP session is blocked per 16,400 mailboxes
- – A message is tagged per 7,040 mailboxes
- – A message is not tagged per 27,400 mailboxes

- Millions of compromised systems send > 70% of spam
- > 80% of email is spam
- 230 kbit/sec In & 278 kbit/sec Out per example SPF attack script
- 100k Bot campaign at a msg rate of one per minute and just 2 SPF ops/msg sent to an average of 10 recipients, delivers spam, and will reflect an attack at 8 Gb/s In and 10 Gb/s Out
- The attack is virtually free to the bad actors
- The victim may have had nothing to do with SPF or even email
- Congestion avoidance is circumvented in SPF libraries
- Millions of domains are added and deleted every day

SPF/Sender-ID Deployment Increases Risk

- ~3% of domains with MX RR publish SPF
- Fortune 100 & Top 20 domains:
  - 72% / 70% offer no SPF records
  - 6% / 10% SPF records fail Neutral
  - 13% / 10% SPF records fail Softfail
  - 9% / 10% SPF records fail Fail
- Abusive sources:
  - 77% offer no SPF records
  - 0.2% SPF records Pass
  - 14% SPF records offer Neutral results
  - 6% SPF records offer Softfail results
  - 2.6% SPF records offer Fail results

Hard to detect SPF enabled attacks

- Flood of DNS traffic from highly distributed sources
- Sources within otherwise well managed domains
- Queries may exhibit large random names for:
  - Wildcard SPF RRs
  - Wildcard MX RRs
  - Invalid address records
- Packet source/destination addresses are valid
- Email logs do not explain the high level of DNS traffic
- Traffic originates from DNS serving access points & MTAs
- Attack concurrent with legitimate DNS traffic
- Might also be concurrent with suspicious poisoning traffic
- Might also be concurrent with a high level of DNS timeouts

Preventing SPF attacks

- Authenticate the client before evaluating message content
- Avoid processing scripts referenced from unknown clients
- If one must publish SPF for white-listing:
  - Publish just IP addresses
  - Terminate SPF scripts with '+all' to nullify advantage in using SPF script libraries

- When SPF/Sender-ID becomes widely deployed & exploited:
  - Establish AUPs that prohibit use of SPF script processing
  - Return 0 answers for records containing SPF scripts

Providers Hiding their Role in Spam

- Sender-ID may require more than 400 DNS transactions to keep the SMTP client nameless (when IPv6 becomes common)
- A safer approach:
  - Confirm SMTP client by the IP address
  - Associate the host-name (even an IP address literal) with originating domains within 1 small DNS transaction
- DKIM's unnecessary limitation on linking identities will force customers into giving provider's their private-keys or access to their DNS... Why?
  - Obscuring SMTP client domains avoids complaints
  - Spam is someone else's problem, never the one sending it
  - Customers are causing the problem, not us

DKIM withThird-Party Authorization

DKIM still allows message replay, but Sender-ID does not offer a good fix. DKIM also expects providers to have use of their customer's keys which also greatly increases risks.

A solution:

•Allow third-party domains to authorize signing domain

•Provide a small single DNS lookup mechanism to associate originating domains with the signing domain (i.e. isp.com).

isp.com as a sha1/base32 reference within example.com:

hgssd3snmi6635j5743vdjhajkmpmfif._ssp.example.com.