Improving DNS Service Record (SRV) weights in sks-keyservers.net

Kristian Fiskerstrand

May 28, 2012

1 Introduction

http://sks-keyservers.net provide a convenient way for end users of the security framework OpenPGP[10] to access synchronised and responsive HKP[11] keyservers, mainly based on the software SKS[9]. A pool of keyservers is available at hkp://pool.sks-keyservers.net in addition to various sub-pools[5]. This article will describe the implementation of DNS Service Records (**SRV**) weights as currently found in the sub-pools *eu.pool* and *na.pool* as a basis for a discussion and propose a better suited algorithm and implementation for determining the SRV weights.

SRV provide a mechanism for directing more of the traffic to specific servers and is supported in GnuPG[1] since versions 1.4.10 and 2.0.13[12], whereby it is currently not supported by PGP[14, 13]. Clients not supporting SRV weights directly will, however, still benefit as these are used as selection criteria for which of the servers are included with regular A and AAAA DNS records in the sub-pools.

2 Current implementation

Currently the SRV weights are based on a simple network timing, where that status page[4] of a given SKS keyserver is being downloaded, and the full calculation is being performed in SKS_GET_PEER_DATA.PHP [2]. The SRV weight is calculated as:

$$\text{weight} = (\textbf{int})\left(\frac{100}{\text{responsetime}}\right) \tag{1}$$

Responsetime is defined (calculated) in [2, lines 102–115] and a local adjustment is applied to any SKS server on the local network of the pool server[2, lines 119–122]. The SKS servers are then sorted in descending order by their weights, and the top 10 servers are included in the respective sub-pool. The SRV weights are calculated on each full update run of the pool, currently running every hour.

2.1 Advantages

The advantage of eq. (1) is that the calculation is transparent and easily constrained within the server discovery process. The resulting output values perform reasonably well, and a SRV capable client is performing most of the work in determining which server to direct the traffic to.

2.2 Disadvantages

Only a single measurement is performed at the time of server discovery, and the performance measurements as a client is only performed from a single point[1] for each of the pools[2]. In addition the size of the SKS status page is typically less than 3 KiB, which doesn't provide enough information for a proper measurement of the bandwidth capacity of the server. Other factors, such as the ability to serve multiple requests e.g. by using a reverse proxy in front of SKS to counter that *"a bandwidth-constrained client is capable of executing this DoS attack inadvertently"* are not considered[3].[6]

3 Proposal for new implementation

A new implementation ideally consider

(I) The response time (**R**)

(II) The bandwidth capacity (**B**)

(III) That the server is running behind a reverse proxy so that it can handle multiple requests (**P**)

The measurement for R is done in seconds, with millisecond precision. Multiple clients should be used for these measurements in order to avoid the problem of only one client measurement in the current implementation. A PHP client[7] and a Perl/CGI client[8] are already available, and it can easily be ported to other platforms. The clients can be run by multiple users at various locations, and are accessible to the server performing the update run, to get timings for downloading a specific OpenPGP key from various servers.

To get more stable result metrics over time some smoothing can be performed, e.g. for R in the form

$$R = \frac{\sum\limits_{j=1}^{C}\sum\limits_{i=0}^{n-1}(1-\frac{i}{n})R_{ji}}{\sum\limits_{j=1}^{C}\sum\limits_{i=0}^{n-1}1-\frac{i}{n}} \tag{2}$$

[1] The location of the server

[2] Two mirrors exists, one in Norway that is used for the EU pool and one in USA that is used for the North America pool

[3] Although this can be filtered separately similar to the High-Availibility pool ha.pool

where R_{ji} define responsetime in current and earlier measurements from the various clients, n is the number of entries to take into account, C is the number of clients to use in the update run, and R constitute the basis for further calculations. For entries where the client failed to retrieved the key, an error value -1 is stored. While determining R, if $R_{ji} = -1$ the value is skipped.

The use of these measurements should not be strictly linear but penalize slower servers based on an exponential factor of some form, e.g. $(x - z)^y$ where z is defined as a left-skew of the mean (μ), as determined based on an $N(\sigma)$ rejection criteria, in order to remove outliers from the calculation, and y is a static constant (even number) defined previous to implementation. As the calculation is dependant on data collected about all the servers, this should be implemented in SKS-STATUS.PHP[3]. Only servers that are considered to be part of the main pool[4] will be considered.

The bandwidth information could be collected from the server operators e.g. in the form Megabit per seconds in upload capacity.

The process could then be described as:

(1) Gather bandwidth information from the operators

(2) During an update run, determine **(I)** and **(III)** for each individual server.

(3) Calculate the mean (μ_R) and standard deviation (σ_R) of **(I)** across the servers in the pool.

(4) Exclude outliers to the results based on $N(\sigma)$ rejection criteria.

(5) Calculate new μ and σ for the remainder of the servers.

(6) Calculate the SRV weights for the individual servers.

SRV weights would be defined as:

$$\begin{aligned}\text{weight} = \alpha \\ &+ Max\left(\beta_R\left(\frac{1}{(R-(\mu_R-2.5\sigma_R))^y}\right), \phi\right) \\ &+ \beta_B \qquad \left(\frac{B}{\sum_{j=1}^{m} B_j}\right) \\ &+ \beta_P \cdot D_P \cdot \rho\end{aligned} \tag{3}$$

where:

- α is a constant to provide a base weight
- β_x is the loading (weight) of the respective factor in determining the SRV weight
- B is the bandwidth information collected (Mbit/s) on the individual server and B_j refer to the bandwidth capacity of other servers viable for inclusion (total number of m)
- D_P is a dummy variable that is 1 if a reverse proxy exists and 0 otherwise
- ρ is the additional weight given for a reverse proxy enabled server (presuming $\beta_P = 1$, or otherwise scaled in accordance to this variable)
- y is a constant (even number) that define penalization for deviation
- ϕ is the ceiling of weight to be added for R

As deviations in R are penalized in both directions, we left-skew μ_R by $2.5\sigma_R$ in order to give higher weights to more responsive servers. A subset of available clients for measuring response-time should be selected based on the geographical region of the sub-pool. A ceiling is in place for the R defined as ϕ in order for the distribution not to be disproportionate towards a single server.

The SRV weights are then calculated and converted to an integer value. The servers are then sorted by descending order of weights, whereby the top 15 servers are added to the pool.

[4] Needs to be responsive, updated with appropriate number of keys and proper SKS version

References

[1] http://gnupg.org

[2] http://code.google.com/p/sks-keyservers-pool/source/browse/trunk/sks-keyservers.net/status-srv/sks_get_peer_data.php?spec=svn101&r=100

[3] http://code.google.com/p/sks-keyservers-pool/source/browse/trunk/sks-keyservers.net/status-srv/sks-status.inc.php?spec=svn101&r=100

[4] http://keys.kfwebs.net:11371/pks/lookup?op=stats

[5] http://sks-keyservers.net/overview-of-pools.php

[6] http://lists.gnu.org/archive/html/sks-devel/2012-03/msg00006.html

[7] http://code.google.com/p/sks-keyservers-pool/source/browse/trunk/sks-keyservers.net/clients/key_retrieval.php

[8] http://code.google.com/p/sks-keyservers-pool/source/browse/trunk/sks-keyservers.net/clients/key_retrieval.pl

[9] http://code.google.com/p/sks-keyserver/

[10] http://tools.ietf.org/html/rfc4880

[11] http://tools.ietf.org/html/draft-shaw-openpgp-hkp-00

[12] http://lists.gnu.org/archive/html/sks-devel/2010-04/msg00013.html

[13] http://www.symantec.com/connect/forums/dns-service-records-srv

[14] http://www.symantec.com/theme.jsp?themeid=pgp