The Implementation and Evaluation of the Proxy Server's Access Control System

Jann-Perng Tseng, Huei-huang Chen

Department of Computer Science and Engineering, Tatung University

40 Chung-shan North Road, Section 3, Taipei, Taiwan, R. O. C

tseng@mail.moe.gov.tw, hhchen@cse.ttit.edu.tw

ABSTRACT

The Internet has become the largest worldwide source of information, containing information on virtually everything and anything imaginable. Yet having such a volume of information available with such ease of access raises the problem of suitability. No one will ban their children or students from accessing information via the Internet. Neither will they wish a 10-year-old child to access pornography or violent imagery.

With the rapidly growth of WWW server, the web browser becomes the popular information retrieval tool, which pulls the need of proxy server to reduce replicated transmission of data around the long distance or national communication link. We focus on the performance impact of access control scheme of the proxy server. There are many public domain softwares exist to do with the information filtering. As the Block-list grows huge, it seriously impacts the response time and performance of the proxy server.

In this paper we will study the available public domain software. As far as performance concerned, we choose the Squirm and SquidGuard, which are implemented in C programming language. We modified the source code and do the experiments with standalone testing of each program. The result is dramatically different as mentioned in section five. It turns out that the response time of SquidGuard nearly remains constant while increasing the number of URL block-list. Finally, there are works can be continued. That is the construction of block-list database and the maintenance interface program.

Keywords: *proxy, cache, access control, filter, blocking, performance.*

1. INTRODUCTION

1.1 Application on the Internet

The Internet has undergone explosive growth over the last few years. It becomes the largest worldwide source of information, containing information on virtually everything and anything imaginable. With each passing day, the Internet is growing and becoming more congested. It is estimated that by the end of January 1999, the Internet would be consisted of 45 million computers worldwide[1]. Today, the number of computers connected to the Internet has more than doubled since 1996. Subsequently, Internet traffic jams and bottlenecks, or what are known as flashpoints and hot-spots, have become daily occurrences

Yet having such a volume of information available with such ease of access raises the problem of suitability. No one will to ban his or her child or student from accessing information via the Internet. However, one does not wish a 10-year-old child accessing pornography or violent imagery. Similarly, many companies wish to grant employees Internet access for work-related purposes but do not what them to make use of it recreationally. Many similar situations exist, making it a necessity to create some form of large-scale content management.

1.2 Information retrieval via WWW browser

There are many ways to the information retrieving and browsing. The general tools used are WAIS, gopher and WWW. While all three of these information presentation systems are client-server based, they differ in terms of their model of data. In gopher, data is a menu, a document, an index or a telnet connection. In WAIS, everything is an index and everything that is returned from the index is a document. In WWW, everything is a possibly hypertext document which may be searchable.

In practice, this means that WWW can represent the gopher and WAIS data models as well as providing extra functionality. World Wide Web usage grew far

beyond Gopher usage in the late 1996, according to the statistics-keepers of the Internet backbone. WWW has long since reached critical mass, with new commercial and noncommercial sites appearing daily. From the Netcraft Web Server Survey[2], there are more than 6.6 millions of Web server on the Internet in July 1999.

A resource retrieval on the World Wide Web (WWW) starts with a request issued by a client. The WWW server replies with any requested resources. The client parses this response and displays it The pros of Web server[3][4] are as following.

- Access for all
- The power to create hypertext
- Anything can refer to anything
- Independent of everything else
- Minimalist design
- Working together: social efficiency, understanding and scaling
- Presentation- ideal for human communication
- Content-machine-aided human communication

1.3 The growth of WWW server pull the need of proxy server

The explosive growth of WWW Server which provided with text, image, audio and video data. The data object size ranging from a few Kbytes to several Mbytes. It means that people need much more bandwidth to meet the requests. Network administrators are facing with the difficulty of how to provide more efficient bandwidth and server utilization. In order to meet this challenge, many are turning to proxy caching as the solution. Some of the many Web cache projects include NLANR (National Laboratory for Applied Network Research) (United States); CHOICE Project (Europe); HENSA (United Kingdom); Academic National Web Cache (New Zealand); W3 CACHE (Poland); SingNet (Singapore); CINECA (Italy); and Japan Cache/JC (Japan).[5][6][7]

The World Wide Web and the phrase "traffic jam" have become as linked in the minds of many computer users as are the urban superhighway and "rush hour" to the early morning commuter. Insufficient bandwidth causing high latency is a daily headache. Caching is a standard solution for this kind of problem, and it was applied to the Web early on for this reason.

2. INFORMATION FILTERING ON THE INTERNET

2.1 Introduction to information filtering and blocking

The Internet is an extraordinary reference tool that can help children excel in learning. However, valuable as it is, the World Wide Web is dangerous in some way. Besides information and entertainment, the World Wide Web can be a source of pornography plus other material that contradicts your personal family values.These unsuitable information include racism, sexually explicit, drug/alcohol, gambling, violence and hate speech. Which we take the followings for example. The researcher confronts with a challenge to block or filter the bad information.

- Sexually-oriented or erotic full or partial nudity
- Adult products including sex toys, CD-ROMs, and video
- Recipes or instructions for manufacturing or growing illicit substances, including alcohol, for purpose other than industrial usage
- Online gambling or lottery web sites that invite the use of real money
- Sites that make available guns, artillery, other weapons, or Poisson substances

From the Information filtering research and paper[8][9], we can classify the filter blocking technologies into three categories.

1. **Keyword blocking**: blocking word patterns (breast, butt, death)
2. **Site blocking**: blocking pre-identified URLs
3. **Web Rating System**: with rating information embedded in each web page

Keyword blocking uses software to identify sites. It is cheap but inaccurate. Site blocking uses humans to select and categorize URLS, which is cost more but is less inaccurate. Both keyword and site blocking preventing site from the transmission of entire files or directories what's being blocked. It is the typical function of most filter software. Site lists also prevent local control. There are more other methods for filtering such as blocking by category, user or workstation ID, time of day, or protocol. From the view point of operation, filtering software could be categorized as client or server site.

- Client Software

  Cyber Patrol, Surfwatch, Net Shepherd

- Server-Based Software--usually a proxy-server

  Cyber Patrol, Websense, Smart Filter, I-Gear,

X-Stop

The web rating system will be described in section 2.3. With the circumstances of web prevalence, there are many server-based filters work with proxy servers. Proxy servers redirect Internet queries from your browser through the proxy server. We will focus on this type of server-based software in this paper.

2.2 Proxy server based information filtering

The general mechanisms for large-scale content control are the application of access control rules on proxy-routed requests and packet filtering of restricted IP addresses on routers and client PCs. These methods have been adopted as national-level controls by countries such as Singapore [6], China, and others with nationally controlled Internet service providers (ISPs). Both methods are similar in that they manually keep track of a list of questionable URLs and act upon the existence of a user-requested URL within this list.

The application of access control rules on proxy-routed requests include the redirection of all Internet requests through a compliant proxy server. Users are only granted Internet access via this proxy, ensuring that all relevant Internet requests are subject to the chosen content selection rules. Each URL request directed through this proxy is checked against the corresponding list of questionable URLs. If the requested URL is not present within the list, the request is allowed to continue uninterrupted. However, if it is present in the list, it is subjected to the access-control rules associated with it. These rules might specify total restriction to the page for all users, or may contain a subset of users to whom the restriction is to be applied.

2.3 Platform of Internet content selection (PICS)

A new series of methods has emerged based on the Platform for Internet Content Selection (PICS) infrastructure in 1995. PICS has been formulated by the World Wide Web Consortium (W3C) and allows for the classification of URLs through the use of associated PICS labels. Each label associated with a URL classifies that URL according to the ratings specified in the label format or ratings system.[10]

It is a new method for large-scale content selection using a PICS-aware proxy system. Internet requests can be redirected through a proxy. For each request, the proxy can fetch a corresponding PICS label and compare its ratings against the corresponding restriction criteria specified for the person making the request. If, upon comparison, any of the ratings contained within the label are not suitable, access to the URL in question can be denied. It is a thoroughly methodology change to the data mining field on the Internet. But it needs time to be waiting for the majority of the Internet people to apply this specification.

3. PROXY BASED ACCESS CONTROL LIST

3.1 The system architecture of the proxy server -- Squid

Squid is derived from software developed on the ARPA-funded Harvest Project. It offers high performance proxy caching for Web clients. It supports FTP, Gopher, and HTTP requests. The cache software, available only in source, is relatively fast because it never needs to fork, is implemented with non-blocking I/O, keeps meta data and hot objects in VM, and caches DNS lookups. The features, advantages, and disadvantages supported by Squid can be found in [11]. We will focus on the access control scheme of proxy server.

In order to study the access control system, we analyze the Squid source code. The system architecture of the Squid software is depicted in Figure 1. The main components are: proxy process which is the core running images of Squid, objects in-memory cache and disk, and access control list. Besides the main components, it also supports interfaces for HTTP client, other proxy server, and access function to DNS, WEB, FTP, WAIS etc. server. The proxy application acts as an intermediary between Web clients and servers. Without a proxy, clients make TCP connections directly to servers. These caching Web objects can greatly reduce access times for popular data. At the same time, Web caches also reduce network bandwidth by satisfying some requests directly from cached data.

3.2 The ACL function supported by Squid

Access-control list is a part of Squid's software to specify the availability to user or the other proxy server. The stopping or allowing people from using it as a proxy server is only one of the functions of ACLs. ACLs are also used for cache hierarchies. Thus you will define an ACL first, and then deny or allow access to a function of the cache. The ACL functions include the following functions.

- Internal cache manager access control
- Other proxy server access control(Parent or Child cache)
- Client source address access control
- Client destination address access control
- Time based access control

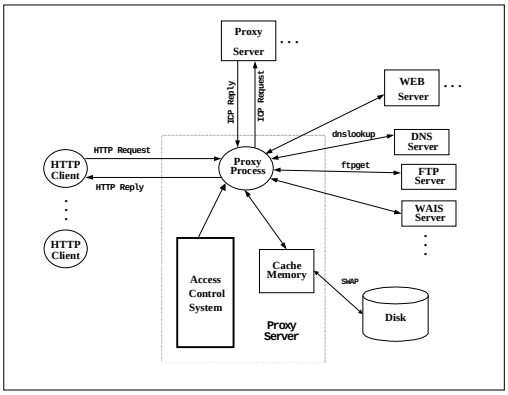

Figure 1. The system architecture of Squid.

Squid works its way through the http access list from top to bottom when deciding which class you fall into, and also as to if you are denied or allowed access. If a user from connects using TCP and request a URL, Squid will work its way through the list of http access lines. It works through this list from top to bottom, stopping after the first match to decide which one they are in. It comes with the problem of doing with a huge block-list. We say, maybe ten thousands or more over hundred thousand URL block-list. It becomes difficult and inefficient to do the maintenance of block-list. These circumstances also yield the proxy server downgrade and more latency for response time. Things will get worse if it is necessary to update the block-list periodically. This is the reason why we search a solution to solve the problem.

3.3 Squid related ACL software

We have pointed out the awful condition of squid's ACL in the previous section. There are various softwares to meet with the challenge. To name a few of the commercial products such as SafeSurf, The Internet filter, NetFilter, Net Rated. We will evaluate the public domain software in this paper not only for the reason of cost but it is also worth while to the academic field. The source code of the public domain software is commonly available. We can change the source code for experiment, additional research and modification is also possible in this case.

The squid related ACL software are as following[11][12]. They are also calling a redirector program of Squid, some of which are for specific platform, and some for specific purpose. We choose the software base on performance concerned. The objective is not only for campus wide usage but a circumstance of a regional network center or ISP. Thus, coding in C programming language is better than that in script language.

- Custodian
- Iain's redirector package
- jesred
- Junkbusters
- SquidGuard
- Squirm

4. IMPLEMENTATION

4.1 The Squirm and SquidGuard software

The SquidGuard is a free, flexible and efficient filter and redirector program for Squid. It defines multiple access rules with different restrictions for different user groups on a Squid cache. It uses Squid standard redirector interface. The filtering functions supported by Squid are as followings[12].

- Limit the web access for some users to a list of accepted/well known web servers and/or URLs only.
- Block access to some listed or blacklisted web servers and/or URLs for some users.
- Block access to URLs matching a list of regular expressions or words for some users.
- Enforce the use of domainnames/prohibit the use of IP address in URLs.
- Redirect blocked URLs to an "intelligent" CGI based info page.
- Redirect unregistered user to a registration form.
- Redirect popular downloads like Netscape, MSIE etc. to local copies.
- Redirect banners to an empty GIF
- Have different access rules based on time of day, day of the week, date etc.
- Have different rules for different user groups.

The Squirm is a configurable, efficient redirector for Squid by Chris Foote. The capabilities stated in the document of the Squirm are as following[13]

- Very fast
- Virtually no memory usage
- It can re-read its configuration files while running by sending it a HUP signal
- Interactive test mode for checking new configuration
- Full regular expression matching and

replacement

4.2 Implementation environment and related software

The environment we used for testing is described in the following. In addition to the hardware and software, we setup a block-list of filtering URLs. It is about ten thousands of URLs, which links to the pornographic picture or image. From the viewpoint of practical operation, the unsuitable information general resides in specific URL rather than the whole domain. It is also prevented from the blocking of unnecessary URLs under certain domain. That's why the evaluation is based on URLs instead of domain. The usage of URL lead to the huge amount of block-list.

- **Hardware**

  SUN workstation, CPU Cycle Time: 400 MHz, Cache Size: 1024M, Disk Capacity: 2 *18G, 100M Fast Ethernet.

- **Software**

  1. Squid: offers high performance proxy caching for web clients.
  2. SquidGuard: is a combined filter, redirector and access controller plug-in for Squid
  3. Bison: The GNU/FSF parser generator used by SquidGuard
  4. Flex: the fast lexical analyzer generator used by SquidGuard
  5. DB library: version (2.X) of the Berkeley DB used by SquidGuard
  6. Squirm: is a combined filter, redirector and access controller plug-in for Squid

4.3 The relationship between squid and redirector program

The system architecture of a redirector program is showing in the Figure 2. The Redirector program is an independent program, which intercept the HTTP client. The mechanism is depicted in the left site of Figure 2 with the sequence of number circled. The main components of the redirector program are in-memory runtime control block, ACL definition, and URL Block List database. The in-memory control block was setup in system start up time. It reads and parses the ACL definition and then load the URL Block List into memory. The dotted square box in figure 2 is not yet implemented.

There are many functions supported by SquidGuard. It includes blocking by source IP or domain, destination IP or domain, time of day, user ident, and regular expression. It also with the capability to support the rewrite rules and redirect rules. Since the impact is trivial except of the huge amount of URL specified, so we will focus on the number of URLs block-list.

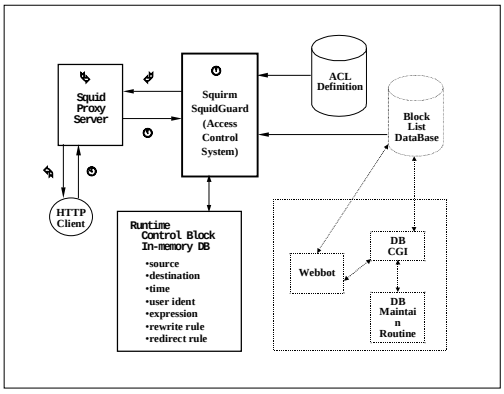

Figure 2. The system architecture of Squirm and SquidGuard.

5. TESTING AND EVALUATION

5.1 Testing environment and case

There are many factors, which affect the performance or response time of proxy server. To name a few, such as the system memory and cache, disk access speed, network bandwidth, software efficiency. Other factors like the structure of document, user preference and strategies of proxy software also inference the performance of proxy server. There are many researches and papers dedicated to specific issue mentioned above.

The performance evaluation of proxy cache is a complicated issue. It is currently no better performance measurement tool or utility to do with. As a matter of facts, there are quite a few researches or case by case evaluations engaged in the performance improvement of proxy server. There is still no Benchmark to be applied or state-of-the-art criteria for the evaluation of proxy server. Since the redirector program of Squid is running as a plug-in of Squid process that we can measure the additional overhead the redirector program imposes upon Squid. That why we proceed our evaluation with standalone testing of each program with definite number of URL block-list.

5.2 Standalone testing of Squirm and SquidGuard

1. Testing of Squirm

   We test the Squirm first. We calculate the response time with increasing number of

URL from 1000 to 10000. The request is ranging from 2000 to 20000. The result is showing in figure 2. The response time is with proportion to the number of URLs. It is a linear relationship between response time and the number of URLs while the number of URLs is less or equal to 4000. As the number of URL increase greater than 4000, the response time increase sharply but still remain the linear growth relationship.

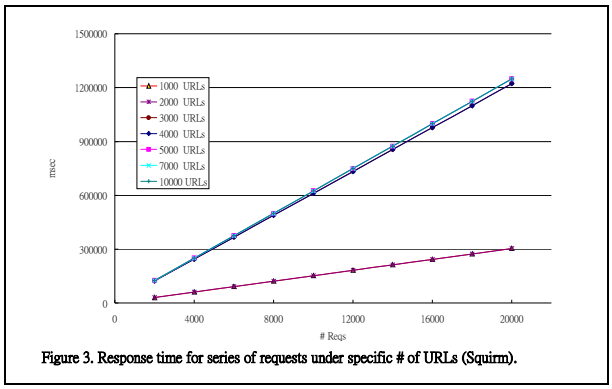

Figure 3. Response time for series of requests under specific # of URLs (Squirm).

2. Testing of SquidGuard

We test the SquidGuard with the same condition and data set which we done in Squirm. The result is showing in figure 4. The response time is with proportion to the number of URLs. It is also a linear relationship between response time and the number of URLs. It remains the linear growth relationship neglecting the number of URL.

5.3 Comparison of the testing result

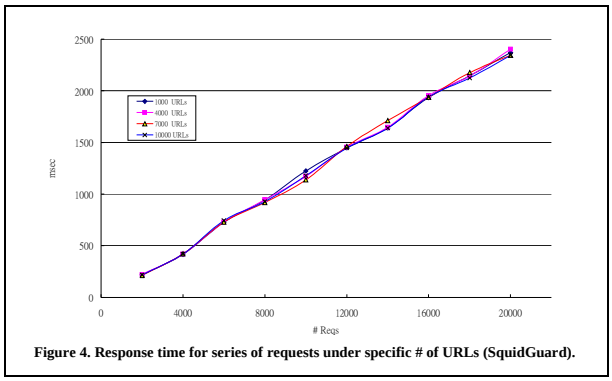

Figure 4. Response time for series of requests under specific # of URLs (SquidGuard).

We calculate the average response time of each program with respect to the number of URLs. The result is depicted in figure 5. The response time of the SquidGuard program remain constant as we increase the number of URLs. It sounds that the response time is independent to the number of URL applied. On the other hand, the average response time increases linearly with proportion to the number of URLs less than 4000. But the average response time remain constant while the number of the URLs greater than 4000.

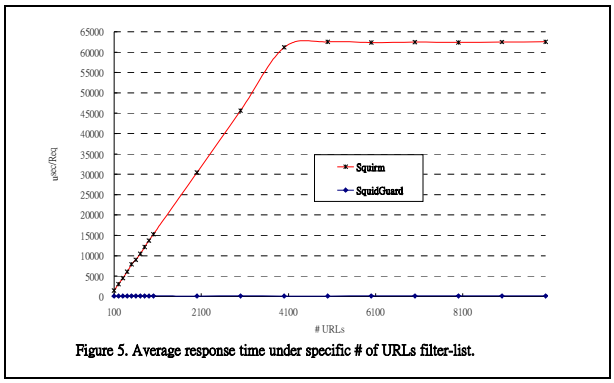

Figure 5. Average response time under specific # of URLs filter-list.

6. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we studied the architecture of the Squid proxy software and its redirector program. Then we modified the source code to do with testing. From the testing we can conclude that the SquidGuard is by far much more efficient than the Squirm, although they are both implemented with C code. Besides the URL blocking features, SquidGuard also afford with abundant of features for the access management of proxy server. It simplifies the management work with neglectible performance impact to the proxy cache.

There are still works as depicted in the figure 2 dotted line to be done in order to make the system workable. It includes the block-list database, which is the collection of forbidden URLs. Another issue is the operational testing of the software with proxy cache which will provide the commitment of this standalone testing. On the other hand it might yield some testing result which could not be obtained from the standalone testing. Although it is convinced that the utilization of regular expression gaining more overhead than URLs, it is expected to be survey under huge amount of URLs. If it remain truth under this circumstance, still left to be more experiment.

1. Block-List Database

As we all aware of the scope of Internet, and we believe that there is no single organization can accomplish the construction of the whole block-list database. With the rapid variation of the Internet environment, it becomes more difficult to do the job. We deeply hope that

there will be sorts of academic or nonprofit organization will support to afford this before the PICS gaining ground.

2. Webbot and DB Maintenance Interface

   The DB maintenance interface is a trivial work to be done. It is simply an application of CGI to database module with update functions. The Webbot with the dotted line in figure 2.is the software agent with the function of automatically gathering or searching the nationwide Internet contents. It generates suspect-list as the base of the block-list database.

We have accomplished the preceding work of the operational testing. The result is conform to the standalone testing in this paper. The response time of the SquidGuard remain constant with respect to the increasing number of URLs. On the other hand, the Squirm was crashed while the number of URLs increasing to about 4000. The detail experimental result has been submitted to NCS'99 conference.

REFERENCES

[1] Internet Software Consortium, "Internet Domain Survey Host Count", http://www.isc.org/

[2] Netcraft Web Server Survey, "Growth in Internet Web Sites August 1995 - July 1999", http://www.netcraft.com/

[3] Steve Putz, "Interactive Information Services Using World-Wide Web Hypertext", Proceedings of the First International Conference on the World-Wide Web May 25-26-27 1994, CERN, Geneva

[4] Martin Hamilton, "Introduction to WWW Caching", http://wwwcache.ja.net/intro/

[5] Michael Baentsch, Lothar Baum, Georg Molter, Steffen Rothkugel, and Peter SturmEnhancing, "the Web's Infrastructure: From Caching to Replication", IEEE Internet Computing, Vol. 1, No. 2, March - April 1997

[6] Dean Provey, John Harrison, "A Distributed Internet Cache", Proceedings of the 20th Australian Computer Science Conference, Sydney, Australia, Feb. 5-7 1997.

[7] Nakul Saraiya, R. Vasudevan, "Measuring Network Cache Performance", The first IRCache Web Cache Bake-off, Jan. 24 1999.

[8] "Filtering the Web using WebFilter", http://math-www.uni-paderborn.de/~axel/NoShit

[9] http://www.bluehighways.com/filters/, Karen G. Schneider, "Shining a Light on Filters in Libraries",

[10] Wayne B. Salamonsen, PICS-Aware Proxy System Versus Proxy Server Filter, INET'97 Proceedings

[11] "Squid Internet Object Cache", http://squid.nlanr.net/Squid/

[12] The Public Domain Filtering Software, http://info.ost.eltele.no/freeware/squidGuard/

[13] Chris Foote, "Squirm - A Redirector for Squid", http://www.senet.com.au/squirm/

[14] MARC J. ROCHKIND, Advanced UNIX Programming, pp 208-225 Prentice-Hall International, Inc. 1993.

[15] Maurice J. Bach, "The Design of the UNIX Operating System", Prentice-Hall International Editions, 1992.

[16] Jerry Peek, Tim O'Reilly, and Mike Loukides, "UNIX Power Tools 2/e", O'Reilly & Associates, Inc. Taiwan Branch, Jan. 1999.

APPENDIX

Table 1. Testing data and result of the squirm (msec).

#Req\ #URL	1000	2000	3000	4000	5000	7000	10000
2000	30228	30210	122454	122274	124994	125164	124766
4000	61178	60788	244404	244404	249660	249596	249512
6000	91276	91082	366580	366580	374642	374368	374394
8000	121720	121638	488984	488984	499256	499446	499438
10000	151946	151774	611044	611044	624846	623992	624060
12000	181860	181906	733692	733692	749306	749024	749598
14000	212780	212676	855562	855562	874270	873724	873580
16000	243120	242704	977966	977966	999532	999688	998664
18000	273200	272892	1100514	1100514	1124334	1123864	1123708
20000	303808	303836	1222360	1222360	1248910	1249040	1248340

Table 2. Testing data and result of the SquidGuard (msec).

#Req\ #URL	1000	4000	7000	10000
2000	219	222	212	218
4000	416	419	421	420
6000	728	730	730	744
8000	944	945	919	927
10000	1224	1172	1138	1179
12000	1447	1453	1462	1447
14000	1640	1647	1712	1639
16000	1934	1952	1936	1939
18000	2145	2144	2176	2125
20000	2369	2401	2343	2342

Table 3. Testing data and result of the Squirm and SquidGuard (usec).

#URL\ #Avg sec	Squirm	SquidGuard
100	1460	100
200	3030	100
300	4457	97
400	6050	100
500	7886	118
600	9010	112
700	10456	111
800	12150	111
900	13711	108
1000	15269	115
2000	30443	108
3000	45658	114
4000	61175	104
5000	62529	102
6000	62345	122
7000	62445	118
8000	62374	119
9000	62464	130
10000	62519	113