

JOHN MENERICK

# OPEN SOURCE FAIRY DUST





# SECURITY DRAGON @ NETSUITE



THE VIEWS AND OPINIONS EXPRESSED HERE ARE MY OWN ONLY AND IN NO WAY REPRESENT THE VIEWS, POSITIONS OR OPINIONS - EXPRESSED OR IMPLIED - OF MY EMPLOYER (PRESENT AND PAST) OR ANYONE ELSE. MY THOUGHTS AND OPINIONS CHANGE FROM TIME TO TIME; THIS IS A NATURAL OFFSHOOT OF HAVING AN OPEN AND INQUISITIVE MIND.

# SECURITY DRAGON @ NETSUITE



THE VIEWS AND OPINIONS EXPRESSED HERE ARE MY OWN ONLY AND IN NO WAY REPRESENT THE VIEWS, POSITIONS OR OPINIONS - EXPRESSED OR IMPLIED - OF MY EMPLOYER (PRESENT AND PAST) OR ANYONE ELSE. MY THOUGHTS AND OPINIONS CHANGE FROM TIME TO TIME; THIS IS A NATURAL OFFSHOOT OF HAVING AN OPEN AND INQUISITIVE MIND.



WHAT WE ARE NOT TALKING ABOUT





WHAT WE ARE TALKING ABOUT



*Ladies and gentlemen, I'd like to present  
to you - the Internet!*





*Ladies and gentlemen, I'd like to present  
to you - the Internet!*

# THE INTERNET - CIRCA 2007



*Ladies and gentlemen, I'd like to present  
to you - the Internet!*



# World Internet Topology

Brought to you by **AT&T Labs**

Powered by **LUMETA**

# THE INTERNET - CIRCA 2007

This map represents the backbone of the Internet as of August 2007. Each line depicts the shortest outgoing route from a test computer to each of more than 320,000 network nodes around the world. The map does not represent the physical or geographic location of servers, but rather is a topological representation of the various networks that combine to form the Internet. It shows the incredible number of interconnected networks owned and maintained by private companies. These networks combine to form the world Internet backbone infrastructure. This map is a product of the Internet Mapping Project. It was compiled and created by Bill Cheswick and Steven North at AT&T Labs Research, using technology and methods developed by the Lumeta Corporation.

## AT&T's Network by the Numbers.

**9.81**

Petabytes of data transmitted across AT&T's networks on an average business day. It is the equivalent of moving the entire written contents of the Library of Congress every three minutes.

**1**

AT&T's rank among broadband providers in the United States.

**12.9 Million**

AT&T broadband customers in America.

**540,000**

Miles of Internet backbone fiber AT&T owns and operates.

**\$6 Billion**

Amount AT&T will spend by 2008 to bring fiber optics deeper into neighborhoods.

**36**

AT&T Internet data centers around the world.

**301,760**

AT&T employees worldwide.

**97%**

Percentage of the world economy reached by AT&T's networks.

**99.998%**

AT&T's network reliability.

**49,000**

Number of Wi-Fi hotspots AT&T provides or enables.

**166**

Number of United States cities where AT&T offers 3G wireless High-Speed Internet access

**3 Million**

AT&T wireless business data subscribers.

**160%**

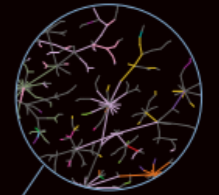
Increase in bandwidth demand per AT&T user between June 2004 and October 2006.

**7**

Nobel Prizes awarded for innovations developed at the AT&T Bell Labs.

**2**

Average number of new patent applications AT&T files - every day.



These clusters represent Internet service provider network hubs. Each cluster represents a service provider capable of connecting thousands of individual users to the Internet. The complexity of the smaller branches are more difficult to depict; they may represent a firewall, another router, or even a smaller DSL router.

## World Internet by the Numbers.

More than **320,000** individual network nodes found by the Internet Mapping Project.

**48 Million** Users on the Internet in 1995. (Source: IDC)

**1.133 Billion** Internet users in 2006. (Source: Internet World Stats)

**6.4 Million** New Internet users getting online every month. (Source: Internet World Stats)

**1.6 Billion** Email boxes in use in 2006. (Source: IDC)

**40 Million** New DNS Hosts every year. (Source: Internet Systems Consortium)

**35,000** Web pages it takes to equal the amount of data transferred when a user downloads 1 HD movie. (Source: Craig Moffett)

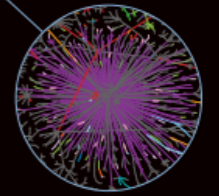
**100 Million** YouTube videos downloaded every day. (Source: YouTube)

**161** Exabytes of new electronic data created every year. (Source: IDC)

**12 Million** Miles of new fiber deployed in 2006. (Source: Telecommunications Industry Association)

**15 Million** Miles of new fiber to be deployed annually by 2009. (Source: Telecommunications Industry Association)

**\$72.5 Billion** Annual spending in support of network infrastructure in the United States by 2009. (Source: Telecommunications Industry Association)



These large groups represent routing hubs. This particular cluster is a routing hub at Global Crossing that feeds dozens of other routers. While industry engineers often "toss out" network data of many possible network engineering solutions employed by service providers. They do not necessarily indicate the relative size of the network connection.



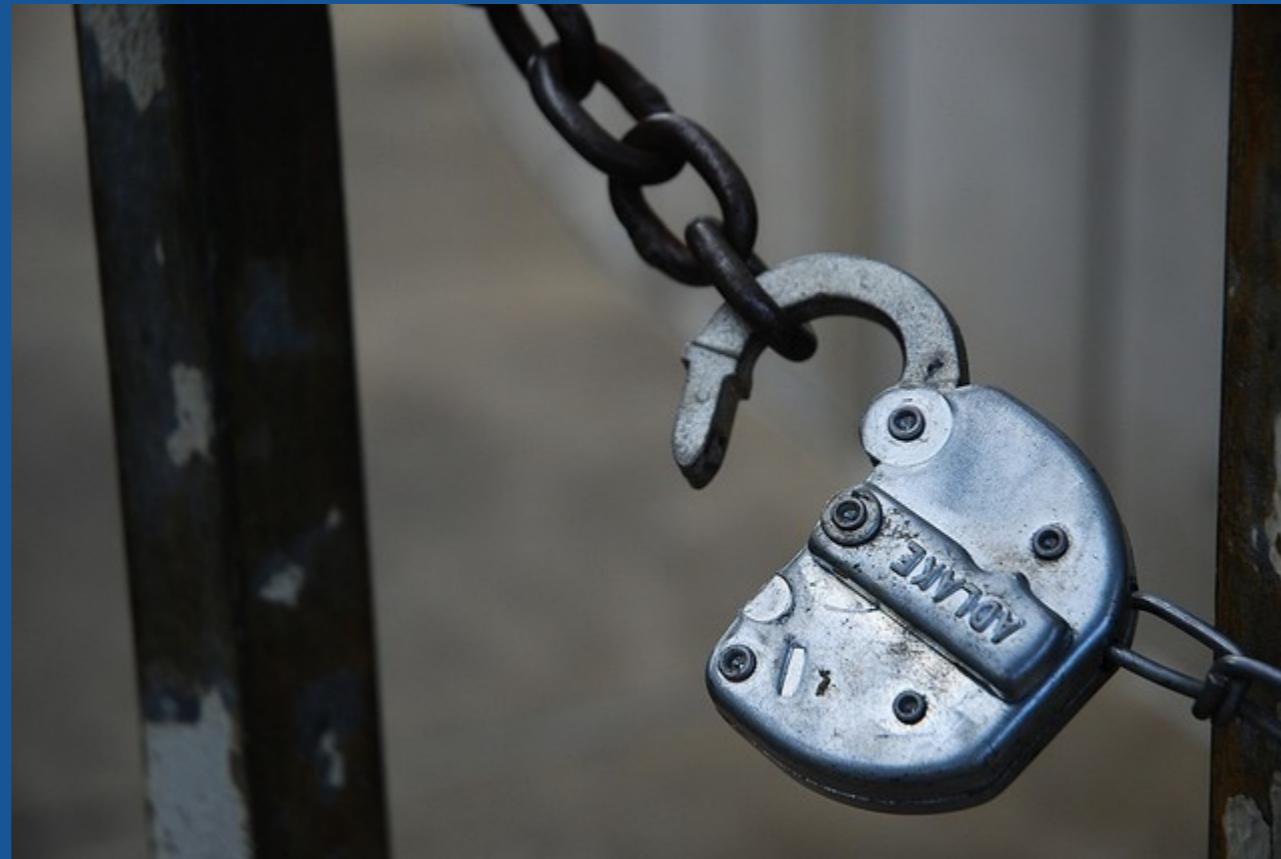


**OPEN SOURCE**

**SOFTWARE**

**FREE** **COPYRIGHT** **WORK** **CONCEPT** **NEW** **SYSTEM** **INNOVATION** **EDIT** **COMMONS** **CULTURE** **AVAILABLE** **DEVELOPMENT** **RESEARCH** **COMMUNITY** **TECHNOLOGY** **INTERNET** **PUBLIC** **INFORMATION** **COST** **DIGITAL** **SHARING** **NETWORK** **POLITICAL** **MOVIE** **GUIDE** **LICENSES** **HARDWARE** **CODE** **MANY CALLED** **BEER** **PDF** **PRESS** **BUSINESS** **PRODUCT** **COMPANIES** **EXAMPLE** **TECHNOLOGIES** **PRODUCTION** **EXPRESSION** **COSTS** **CONTENT** **PROCESS** **INITIATIVE** **ETHICS** **COMMUNITIES** **DESCRIBE** **POTENTIAL** **ORIGINAL** **SYSTEMS** **MADE** **ONLINE** **USE** **SYSTEM** **INNOVATION** **POLITICS** **CONCEPT** **CREATED** **NEW** **LICENSE** **USING** **WIKIPEDIA** **WITHOUT** **WORKS** **RICHARD** **RESOURCES** **LAW** **PRODUCTS** **CASE** **ECONOMIC** **SCIENTIFIC** **ONE** **FORMAT** **FILE** **STANDARDS** **PROPERTY** **MANAGEMENT** **ORGANIZATIONS** **DECISION** **STARTED** **AUTHORS** **HISTORY** **MAKING** **PROJECT** **NOW** **INTERNATIONAL** **MOVEMENT** **LIST** **COMPANY** **RAYMOND** **INCLUDING** **BLOGS** **MESSAGEBOARDS** **PHARMACEUTICALS** **PUBLISHING** **EXAMPLES** **ANYONE** **ALSO** **ECONOMICS** **SHARING** **SIMILAR** **DIGITAL** **NETWORK** **POLITICAL** **TERM** **CULTURAL** **CENTURY** **PLATFORM** **TECHNOLOGY** **OPEN-SOURCE** **COMPUTER** **PROJECTS** **SOCIETY** **INTELLECTUAL** **MANUFACTURERS**





NO ONE SAID IT WAS SECURE

NO ONE SAID IT WAS SECURE



OOPS!

NO ONE SAID IT WAS SECURE







"OPEN SOURCE IS MORE SECURE."

INTERNET SOCIETY PRESIDENT

OOPS!

EVERYBODY'S JOB IS NOBODY'S JOB



“This is a story about four people named

EVERYBODY'S JOB IS NOBODY'S JOB

"This is a story about four people named  
**Everybody,**

EVERYBODY'S JOB IS NOBODY'S JOB

"This is a story about four people named  
**Everybody,**  
**Somebody,**

EVERYBODY'S JOB IS NOBODY'S JOB



“This is a story about four people named  
**Everybody,**  
**Somebody,**  
**Anybody,** and

EVERYBODY'S JOB IS NOBODY'S JOB

“This is a story about four people named  
**Everybody,**  
**Somebody,**  
**Anybody,** and  
**Nobody.**

EVERYBODY'S JOB IS NOBODY'S JOB

“This is a story about four people named  
**Everybody,**  
**Somebody,**  
**Anybody,** and  
**Nobody.**

EVERYBODY'S JOB IS NOBODY'S JOB



There was an important job to be done and **Everybody** was asked to do it.

EVERYBODY'S JOB IS NOBODY'S JOB

**Everybody** was sure

EVERYBODY'S JOB IS NOBODY'S JOB

**Somebody** would do it.

EVERYBODY'S JOB IS NOBODY'S JOB



**Anybody** could have done it, but

EVERYBODY'S JOB IS NOBODY'S JOB

**Nobody** did it.

EVERYBODY'S JOB IS NOBODY'S JOB

**Somebody** got angry about that,

EVERYBODY'S JOB IS NOBODY'S JOB

because it was

EVERYBODY'S JOB IS NOBODY'S JOB



**Everybody's** job.

EVERYBODY'S JOB IS NOBODY'S JOB

**Everybody** thought

EVERYBODY'S JOB IS NOBODY'S JOB

**Anybody** could do it but

EVERYBODY'S JOB IS NOBODY'S JOB

**Nobody** realized that

EVERYBODY'S JOB IS NOBODY'S JOB

**Everybody** wouldn't do it.

EVERYBODY'S JOB IS NOBODY'S JOB

It ended up that **Everybody** blamed **Somebody** when **Nobody** did what **Anybody** could have done.”

EVERYBODY'S JOB IS NOBODY'S JOB



It ended up that **Everybody** blamed **Somebody** when **Nobody** did what **Anybody** could have done.”

**Should we rail against the OpenSSL developers for this?**

Don't even think about it. The OpenSSL team, which is surprisingly small, has been given the task of maintaining the world's most popular TLS library. It's a hard job with essentially no pay. It involves taking other folks' code (as in the case of Heartbeat) and doing a best-possible job of reviewing it. Then you hope others will notice it and disclose it responsibly before disasters happen.

EVERYBODY'S JOB IS NOBODY'S JOB

It ended up that **Everybody** blamed **Somebody** when **Nobody** did what **Anybody** could have done.”

EVERYBODY'S JOB IS NOBODY'S JOB

It ended up that **Everybody** blamed **Somebody** when **Nobody** did what **Anybody** could have done.”

The bug revealed this week was buried inside 10 lines of code and would have been spotted in an audit, according to Laurie, who works on the security team at Google. "Unless you're doing an audit, you're not going to see it," he said.

EVERYBODY'S JOB IS NOBODY'S JOB

Financial

Hobbyist

Fun

Activist

Financial

Hobbyist

Fun

Activist

“OPEN SOURCE PROJECTS PLAY A CRUCIAL ROLE IN THE DIGITAL AGE BUT ARE MAINTAINED BY A SMALL, STRAINED CADRE OF VOLUNTEERS.”

Functionality

Stability

Compliance

Performance

Usability

Security



Stability

Compliance

Performance

Usability

Security

Stability

Performance

Usability

Security

Stability

Performance

Security

Stability

Performance

Stability

Performance

“THERE ARE LOTS OF CRITICAL LIBRARIES MAINTAINED  
BY VOLUNTEERS THAT ARE NOT GIVEN ENOUGH  
ATTENTION”









"WHEN YOU CARRY POINTERS AROUND AND CANNOT TRACK WHETHER THEY ARE ALIVE AND HOW LONG THEY ARE, IT'S GOING TO HURT."







“IT IS NOT LIKE JAVA GOT INSECURE ALL OF A SUDDEN. IT HAS BEEN INSECURE FOR YEARS.”





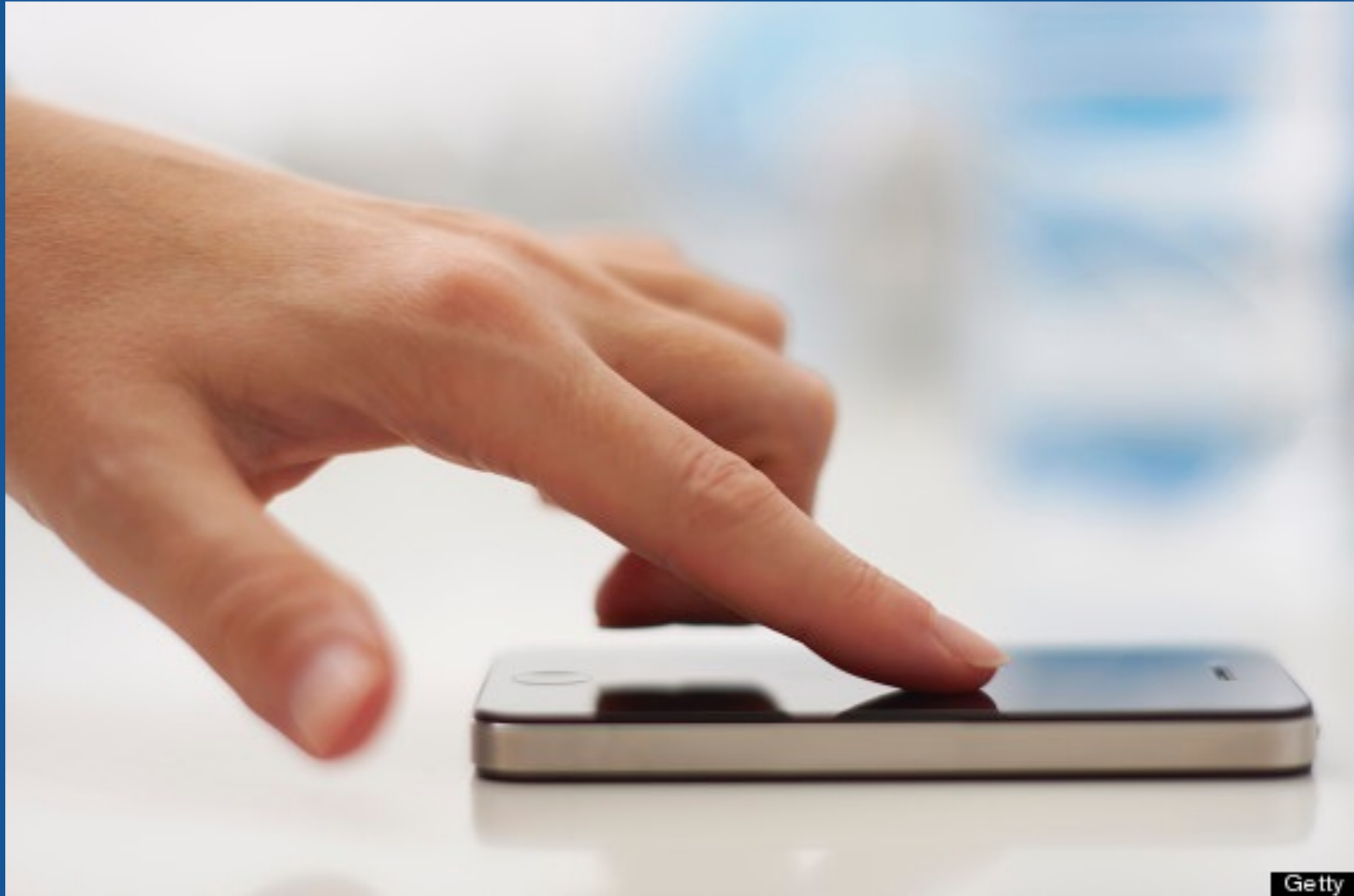
**SOMETHING'S FUCKY**





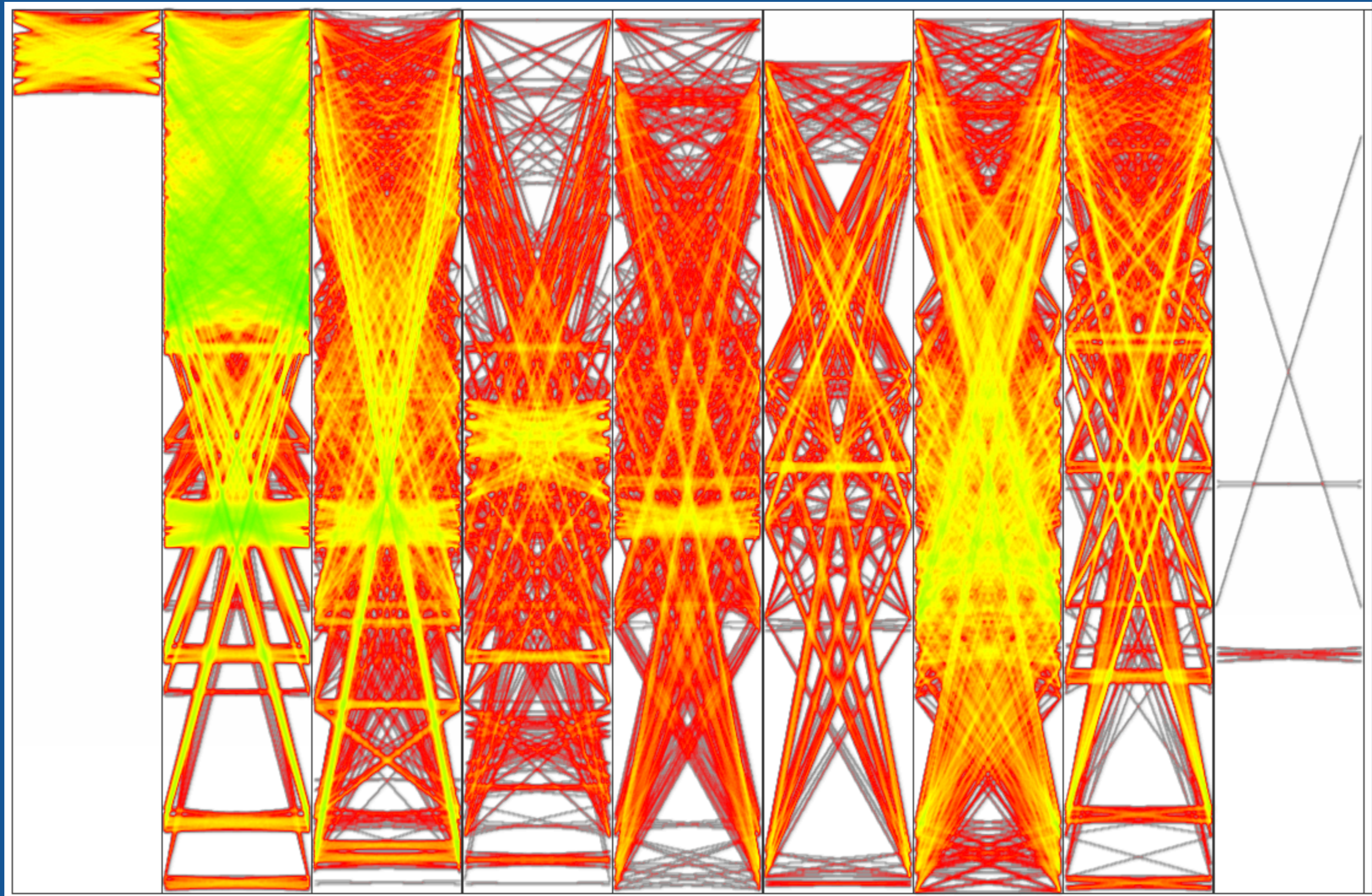


NO STRATEGY



NO SECURITY CONTACT OR REPORTING DEFINED






Inconsistent coding styles, usage, or complex code

## 403 Forbidden

61.147.76.91

China Telecom jiangsu province  
backbone

Added on 12.07.2014

 Nanjing

HTTP/1.0 403 Forbidden

Server: nginx/0.7.67

Date: Sat, 12 Jul 2014 04:59:32 GMT

Content-Type: text/html

Content-Length: 169


Connection: keep-alive

## 403 Forbidden

61.147.76.91

China Telecom jiangsu province  
backbone

Added on 12.07.2014

 Nanjing

HTTP/1.0 403 Forbidden

Server: nginx/0.7.67

Date: Sat, 12 Jul 2014 04:59:32 GMT

Content-Type: text/html

Content-Length: 169

Connection: keep-alive


“System administrators hate change when they have to bear the brunt of adverse effects of change.”

## 403 Forbidden

61.147.76.91

China Telecom jiangsu province  
backbone

Added on 12.07.2014

 Nanjing

HTTP/1.0 403 Forbidden

Server: nginx/0.7.67

Date: Sat, 12 Jul 2014 04:59:32 GMT

Content-Type: text/html

Content-Length: 169

Connection: keep-alive

“System administrators hate change when they have to bear the brunt of adverse effects of change.”

CHANGE IS HARD

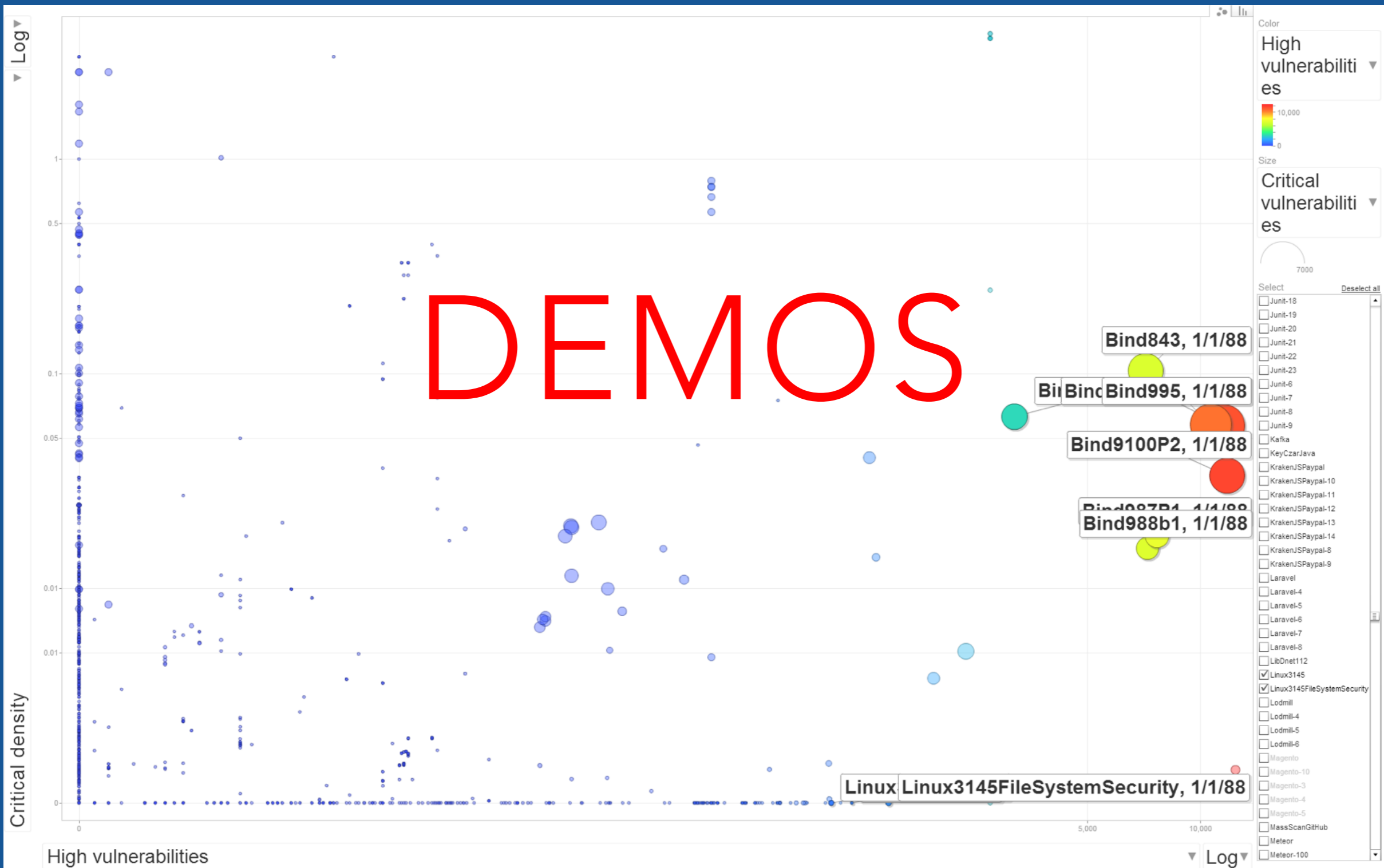


**WHAT  
DO WE  
DO  
NOW?**





# DEMOS



```

108     final String nnAddr = req.getParameter(JspHelper.NAMENODE_ADDRESS);
109     if (nnAddr == null){
110         out.print(JspHelper.NAMENODE_ADDRESS + " url param is null");
111         return;
112     }
113 <<<<
114     DFSClient dfs = getDFSClient(ugi, nnAddr, conf);
115     String target = dir;
116     final HdfsFileStatus targetStatus = dfs.getFileInfo(target);
117     if (targetStatus == null) { // not exists
118         out.print("<h3>File or directory : " + StringEscapeUtils.escapeHtml(target) + " do»
119         JspHelper.printGotoForm(out, namenodeInfoPort, tokenString, target,
120             nnAddr);

```

```

452     public static void printGotoForm(JspWriter out,
453                                     int namenodeInfoPort,
454                                     String tokenString,
455                                     String file,
456                                     String nnAddress) throws IOException {
457         out.print("<form action=\"browseDirectory.jsp\" method=\"get\" name=\"goto\">");
458         out.print("Goto : ");
459         out.print("<input name=\"dir\" type=\"text\" width=\"50\" id=\"dir\" value=\"\"+ HtmlQuoting.quoteHtmlChars»
460         out.print("<input name=\"go\" type=\"submit\" value=\"go\"/>");
461         out.print("<input name=\"namenodeInfoPort\" type=\"hidden\" "
462             + "value=\"\" + namenodeInfoPort + "\"/>");
463         if (UserGroupInformation.isSecurityEnabled()) {
464             out.print("<input name=\"\" + DELEGATION_PARAMETER_NAME
465                 + "\" type=\"hidden\" value=\"\" + tokenString + "\"/>");
466         }
467         out.print("<input name=\"\"+ NAMENODE_ADDRESS +\"\" type=\"hidden\" "
468             + "value=\"\" + nnAddress + "\"/>");
469         out.print("</form>");
470     }

```



```

108     final String nnAddr = req.getParameter(JspHelper.NAMENODE_ADDRESS);
109     if (nnAddr == null){
110         out.print(JspHelper.NAMENODE_ADDRESS + " url param is null");
111         return;
112     }
113 <<<<
114     DFSCClient dfs = getDFSCClient(ugi, nnAddr, conf);
115     String target = dir;
116     final HdfsFileStatus targetStatus = dfs.getFileInfo(target);
117     if (targetStatus == null) { // not exists
118         out.print("<h3>File or directory : " + StringEscapeUtils.escapeHtml(target) + " do»
119         JspHelper.printGotoForm(out, namenodeInfoPort, tokenString, target,
120             nnAddr);

```

```

452     public static void printGotoForm(JspWriter out,
453                                     int namenodeInfoPort,
454                                     String tokenString,
455                                     String file,
456                                     String nnAddress) throws IOException {
457         out.print("<form action=\"browseDirectory.jsp\" method=\"get\" name=\"goto\">");
458         out.print("Goto : ");
459         out.print("<input name=\"dir\" type=\"text\" width=\"50\" id=\"dir\" value=\"\"+ HtmlQuoting.quoteHtmlChars»
460         out.print("<input name=\"go\" type=\"submit\" value=\"go\"/>");
461         out.print("<input name=\"namenodeInfoPort\" type=\"hidden\" "
462             + "value=\"\" + namenodeInfoPort + "\"/>");
463         if (UserGroupInformation.isSecurityEnabled()) {
464             out.print("<input name=\"\" + DELEGATION_PARAMETER_NAME
465                 + "\" type=\"hidden\" value=\"\" + tokenString + "\"/>");
466         }
467         out.print("<input name=\"\"+ NAMENODE_ADDRESS +\"\" type=\"hidden\" "
468             + "value=\"\" + nnAddress + "\"/>");
469         out.print("</form>");
470     }

```

```

196 > /*
197 > *> @todo: This code is SHIT. It's badly formatted. It's
198 > *> hard to understand. It re-implements tons of things
199 > *> which are already in the server core.
200 > */
201 > memset(ht->buffer, 0, 1024);
202 > ht->table = (struct mypasswd **) rad_malloc (tablesize * sizeof(struct mypasswd *));
203 > if (!ht->table) {
204 >     /*
205 >     * Unable allocate memory for hash table
206 >     * Still work without it
207 >     */
208 >     ht->tablesize = 0;
209 >     return ht;
210 > }
211 > memset(ht->table, 0, tablesize * sizeof(struct mypasswd *));
212 > while (fgets(buffer, 1024, ht->fp)) {
213 >     if(*buffer && *buffer!='\n' && (!ignorenis || (*buffer != '+' && *buffer != '-')) ){
214 >         if(!(hashentry = mypasswd_malloc(buffer, nfields, &len)){

```

```

65 static struct mypasswd * mypasswd_malloc(char const* buffer, int nfields, size_t* len)
66 {
67 >     struct mypasswd *t;
68 >     /* reserve memory for (struct mypasswd) + listflag (nfields * sizeof (char*)) +
69 >     ** fields (nfields * sizeof (char)) + strlen (inst->format) + 1 */
70
71 >     *len=sizeof (struct mypasswd) + nfields * sizeof (char*) + nfields * sizeof (char ) »
72 >     t = (struct mypasswd *) rad_malloc(*len);

```

```

298 /*
299 *> Allocate memory, or exit.
300 *
301 *> This call ALWAYS succeeds!
302 */
303 void *rad_malloc(size_t size)
304 {
305 >     void *ptr = malloc(size);
306
307 >     if (ptr == NULL) {
308 >         ERROR("no memory");
309 >         fr_exit(1);
310 >     }
311
312 >     return ptr;
313 }

```

```

196 > /*
197 >  *> @todo: This code is SHIT. It's badly formatted. It's
198 >  *> hard to understand. It re-implements tons of things
199 >  *> which are already in the server core.
200 >  */
201 > memset(ht->buffer, 0, 1024);
202 > ht->table = (struct mypasswd **) rad_malloc (tablesize * sizeof(struct mypasswd *));
203 > if (!ht->table) {
204 >     /*
205 >     * Unable allocate memory for hash table
206 >     * Still work without it
207 >     */
208 >     ht->tablesize = 0;
209 >     return ht;
210 > }
211 > memset(ht->table, 0, tablesize * sizeof(struct mypasswd *));
212 > while (fgets(buffer, 1024, ht->fp)) {
213 >     if(*buffer && *buffer!='\n' && (!ignorenis || (*buffer != '+' && *buffer != '-')) ){
214 >         if(!(hashentry = mypasswd_malloc(buffer, nfields, &len))){

```

```

65 static struct mypasswd * mypasswd_malloc(char const* buffer, int nfields, size_t* len)
66 {
67 >     struct mypasswd *t;
68 >     /* reserve memory for (struct mypasswd) + listflag (nfields * sizeof (char*)) +
69 >     ** fields (nfields * sizeof (char)) + strlen (inst->format) + 1 */
70
71 >     *len=sizeof (struct mypasswd) + nfields * sizeof (char*) + nfields * sizeof (char ) »
72 >     t = (struct mypasswd *) rad_malloc(*len);

```

```

298 /*
299 *> Allocate memory, or exit.
300 *
301 *> This call ALWAYS succeeds!
302 */
303 void *rad_malloc(size_t size)
304 {
305 >     void *ptr = malloc(size);
306
307 >     if (ptr == NULL) {
308 >         ERROR("no memory");
309 >         fr_exit(1);
310 >     }
311
312 >     return ptr;
313 }

```

Free Radius 3.0.3



```
34 typedef struct filter_cmd {
35     struct filter_cmd *next;
36     int command;
37     BOOL seen;
38     BOOL noerror;
39     union argtypes args[1]; //8 bytes
40 } filter_cmd;
```

```
2331     if (addr->reply->once_repeat < 0)
2332     {
2333         *error_pointer = string_sprintf("Bad time value for \"once_repeat\" "
2334             "in mail or vacation command: %s",
2335             commands->args[mailarg_index_once_repeat]); //96 bytes
2336         return FF_ERROR;
2337     }
2338 }
```

```
34 typedef struct filter_cmd {
35     struct filter_cmd *next;
36     int command;
37     BOOL seen;
38     BOOL noerror;
39     union argtypes args[1]; //8 bytes
40 } filter_cmd;
```

```
2331     if (addr->reply->once_repeat < 0)
2332     {
2333         *error_pointer = string_sprintf("Bad time value for \"once_repeat\" "
2334             "in mail or vacation command: %s",
2335             commands->args[mailarg_index_once_repeat]); //96 bytes
2336         return FF_ERROR;
2337     }
2338 }
```









I AM  
LEGION

(AND SO CAN YOU!)

STREET CRED

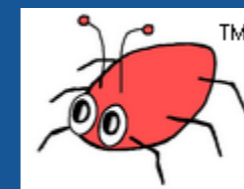
FULL DISCLOSURE

LULZ



WHAT IS YOUR INCENTIVE?





TOOLING



TUNING AND CORRELATION



```

1: APPLY:  $\mathcal{G}$  to  $\mathcal{P}$  to create subpopulations:  $\{\mathcal{P}_{\mathcal{G}_1}, \dots, \mathcal{P}_{\mathcal{G}_m}\}$ 
2: for all  $\mathcal{P}_{\mathcal{G}_i}$  do
3:    $\mathcal{P}_{\mathcal{G}_i}.\hat{p} \leftarrow \mathcal{P}_{\mathcal{G}_i}.s / \mathcal{P}_{\mathcal{G}_i}.n$ 
4: if  $\max_{\mathcal{P}_{\mathcal{G}_i}} \mathcal{P}_{\mathcal{G}_i}.n < 51$  then
5:    $p_0 \leftarrow \left(\frac{1}{m}\right) \sum_{i=1}^m \mathcal{P}_{\mathcal{G}_i}.\hat{p}$ 
6: else
7:    $p_0 \leftarrow 0.85$ 
8: for all  $\mathcal{P}_{\mathcal{G}_i}$  do
9:   if  $\mathcal{P}_{\mathcal{G}_i}.s = 0$  and using NO-SUCCESS HEURISTIC then
10:    Discard  $\mathcal{P}_{\mathcal{G}_i}$ 
11:   else
12:     $\mathcal{P}_{\mathcal{G}_i}.z \leftarrow (\mathcal{P}_{\mathcal{G}_i}.\hat{p} - p_0) / \sqrt{p_0(1 - p_0) / \mathcal{P}_{\mathcal{G}_i}.n}$ 
13: CREATE equivalence classes  $\{E_{z_1}, \dots, E_{z_k}\}$ :  $E_{z_i} \leftarrow \{\mathcal{P}_{\mathcal{G}_j} \cap \mathcal{E} | \mathcal{P}_{\mathcal{G}_j}.z = z_i\}$ 
14: SORT  $\{E_{z_1}, \dots, E_{z_k}\}$  by  $z_i$  in descending order. Designate that order as  $\{E_{z_{(1)}}, \dots, E_{z_{(k)}}\}$ .
15: for all  $i = 1, \dots, k$  do
16:   Inspect error reports in  $E_{z_{(i)}}$  using an auxiliary ranking scheme

```





```

1: APPLY:  $\mathcal{G}$  to  $\mathcal{P}$  to create subpopulations:  $\{\mathcal{P}_{\mathcal{G}_1}, \dots, \mathcal{P}_{\mathcal{G}_m}\}$ 
2: for all  $\mathcal{P}_{\mathcal{G}_i}$  do
3:    $\mathcal{P}_{\mathcal{G}_i}.\hat{p} \leftarrow \mathcal{P}_{\mathcal{G}_i}.s / \mathcal{P}_{\mathcal{G}_i}.n$ 
4: if  $\max_{\mathcal{P}_{\mathcal{G}_i}.n} < 51$  then
5:    $p_0 \leftarrow \left(\frac{1}{m}\right) \sum_{i=1}^m \mathcal{P}_{\mathcal{G}_i}.\hat{p}$ 
6: else
7:    $p_0 \leftarrow 0.85$ 
8: for all  $\mathcal{P}_{\mathcal{G}_i}$  do
9:   if  $\mathcal{P}_{\mathcal{G}_i}.s = 0$  and using NO-SUCCESS HEURISTIC then
10:    Discard  $\mathcal{P}_{\mathcal{G}_i}$ 
11:   else
12:     $\mathcal{P}_{\mathcal{G}_i}.z \leftarrow (\mathcal{P}_{\mathcal{G}_i}.\hat{p} - p_0) / \sqrt{p_0(1 - p_0) / \mathcal{P}_{\mathcal{G}_i}.n}$ 
13: CREATE equivalence classes  $\{E_{z_1}, \dots, E_{z_k}\}$ :  $E_{z_i} \leftarrow \{\mathcal{P}_{\mathcal{G}_j} \cap \mathcal{E} | \mathcal{P}_{\mathcal{G}_j}.z = z_i\}$ 
14: SORT  $\{E_{z_1}, \dots, E_{z_k}\}$  by  $z_i$  in descending order. Designate that order as  $\{E_{z_{(1)}}, \dots, E_{z_{(k)}}\}$ .
15: for all  $i = 1, \dots, k$  do
16:   Inspect error reports in  $E_{z_{(i)}}$  using an auxiliary ranking scheme

```

```

1  $I \leftarrow H$ 
2 while  $U \neq \emptyset$ 
3   do
4     for each  $r$  in  $U$ 
5       do
6          $r.prob \leftarrow \text{UPDATEPROBABILITY}(r, I, \mathcal{M})$ 
7          $r.info \leftarrow \text{INFOGAIN}(r, U, I, \mathcal{M})$ 
8          $r_{best} \leftarrow \text{FIND-MAX-BY-PROB-THEN-INFO}(U, \epsilon)$ 
9          $r_{best}.value \leftarrow \text{USER-INSPECT-REPORT}(r_{best})$ 
10         $I \leftarrow I \cup \{r_{best}\}$ 
11         $U \leftarrow U - \{r_{best}\}$ 

```



- [hacktheplanet.ninja/index.html](http://hacktheplanet.ninja/index.html)

ONE MORE  
THING...



**KEEP  
CALM  
AND  
HIDE THE  
INTERNET**





```
2785 procfs_ttyname (int rdev)
2786 {
2787     FILE *fdev = NULL;
2788     char name[PATH_MAX];
2789
2790     block_input ();
2791     fdev = fopen ("/proc/tty/drivers", "r");
2792
2793     if (fdev)
2794     {
2795         unsigned major;
2796         unsigned long minor_beg, minor_end;
2797         char minor[25]; /* 2 32-bit numbers + dash */
2798         char *endp;
2799
2800         while (!feof (fdev) && !ferror (fdev))
2801         {
2802             if (3 <= fscanf (fdev, "%*s %s %u %s %*s\n", name, &major, minor)
```

2788: char name[PATH\_MAX]; // 4,096 Bytes

2802: fscanf is 33,554,431 bytes