vanhauser-thc / **thc-hydra**

hydra

168 commits **1** branch **0** releases **14** contributors

Branch: **master** New pull request New file Find file HTTPS https://github.com/vanhau Download ZIP

vanhauser-thc makefile fix Latest commit `6c1db2a` Jan 20, 2016

hydra-gtk	xhydra patch	Dec 9, 2014
web	three fixes	May 26, 2014
Android.mk	disabled firebird: SIGSEGV on alloc.cpp:520	Nov 29, 2015
CHANGES	HYDRA_PROXY_CONNECT warning (unused)	Jan 9, 2016
INSTALL	initial commit	Apr 24, 2014
LICENSE	initial commit	Apr 24, 2014
LICENSE.OPENSSL	initial commit	Apr 24, 2014
Makefile	makefile fix	Jan 20, 2016
Makefile.am	Update from original repository	Jun 25, 2015
Makefile.orig	initial commit	Apr 24, 2014

Makefile.unix	initial commit	Apr 24, 2014
README	-M and -6 clarification	Dec 12, 2014
TODO	-M port fix; and always print the specified target name when found	Nov 30, 2014
bfg.c	initial commit	Apr 24, 2014
bfg.h	initial commit	Apr 24, 2014
configure	ssl,ssh and http-form fixes	Jan 20, 2016
crc32.c	add gzip support for login and password files	Jul 1, 2015
crc32.h	add gzip support for login and password files	Jul 1, 2015
d3des.c	initial commit	Apr 24, 2014
d3des.h	initial commit	Apr 24, 2014
dpl4hydra.sh	initial commit	Apr 24, 2014
dpl4hydra_full.csv	Update from original repository	Jun 25, 2015
dpl4hydra_local.csv	Update from original repository	Jun 25, 2015
hmacmd5.c	initial commit	Apr 24, 2014
hmacmd5.h	initial commit	Apr 24, 2014
hydra-afp.c	added cidr support	May 3, 2014
hydra-asterisk.c	more pre-8.0 fixes	Apr 28, 2014
hydra-cisco-enable.c	added cidr support	May 3, 2014
hydra-cisco.c	removed warnings	Nov 28, 2014
hydra-cvs.c	initial commit	Apr 24, 2014
hydra-firebird.c	added cidr support	May 3, 2014

hydra-ftp.c	initial commit	Apr 24, 2014
hydra-http-form.c	ssl,ssh and http-form fixes	Jan 20, 2016
hydra-http-proxy-urlenum.c	added cidr support	May 3, 2014
hydra-http-proxy.c	added cidr support	May 3, 2014
hydra-http.c	Update from original repository	Jun 25, 2015
hydra-icq.c	added cidr support	May 3, 2014
hydra-imap.c	Update from original repository	Jun 25, 2015
hydra-irc.c	more pre-8.0 fixes	Apr 28, 2014
hydra-ldap.c	pre-8.0	Apr 27, 2014
hydra-logo.ico	initial commit	Apr 24, 2014
hydra-logo.rc	initial commit	Apr 24, 2014
hydra-mod.c	ssl,ssh and http-form fixes	Jan 20, 2016
hydra-mod.h	Trim no longer needed.	Aug 12, 2014
hydra-mssql.c	initial commit	Apr 24, 2014
hydra-mysql.c	mysql.h location can be both <mysql/mysql.h> and <mysql.h>	May 27, 2014
hydra-ncp.c	added cidr support	May 3, 2014
hydra-nntp.c	more pre-8.0 fixes	Apr 28, 2014
hydra-oracle-listener.c	reverted openssl des_ => DES_ renamin patch as this does not compile …	Mar 18, 2015
hydra-oracle-sid.c	initial commit	Apr 24, 2014
hydra-oracle.c	initial commit	Apr 24, 2014
hydra-pcanywhere.c	added cidr support	May 3, 2014

hydra-pcnfs.c	added cidr support	May 3, 2014
hydra-pop3.c	removed warnings	Nov 28, 2014
hydra-postgres.c	added cidr support	May 3, 2014
hydra-rdp.c	define ssl_cert_free as object-scoped	Nov 29, 2015
hydra-redis.c	Update from original repository	Jun 25, 2015
hydra-rexec.c	more pre-8.0 fixes	Apr 28, 2014
hydra-rlogin.c	more pre-8.0 fixes	Apr 28, 2014
hydra-rsh.c	more pre-8.0 fixes	Apr 28, 2014
hydra-rtsp.c	Add new module	Jun 25, 2015
hydra-s7-300.c	more pre-8.0 fixes	Apr 28, 2014
hydra-sapr3.c	initial commit	Apr 24, 2014
hydra-sip.c	more pre-8.0 fixes	Apr 28, 2014
hydra-smb.c	reverted openssl des_ => DES_ renamin patch as this does not compile …	Mar 18, 2015
hydra-smtp-enum.c	pre-8.0	Apr 27, 2014
hydra-smtp.c	Update from original repository	Jun 25, 2015
hydra-snmp.c	reverted openssl des_ => DES_ renamin patch as this does not compile …	Mar 18, 2015
hydra-socks5.c	initial commit	Apr 24, 2014
hydra-ssh.c	verbose ssh connect error	Jan 20, 2016
hydra-sshkey.c	removed warnings	Nov 28, 2014
hydra-svn.c	initial commit	Apr 24, 2014
hydra-teamspeak.c	add gzip support for login and password files	Jul 1, 2015

hydra-telnet.c	removed warnings	Nov 28, 2014
hydra-vmauthd.c	pre-8.0	Apr 27, 2014
hydra-vnc.c	pre-8.0	Apr 27, 2014
hydra-wizard.sh	v8.1 release	Dec 8, 2014
hydra-xmpp.c	Update from original repository	Jun 25, 2015
hydra.1	david sadly cant help maintain hydra anymore	Sep 11, 2014
hydra.c	error output fixes	Jan 19, 2016
hydra.h	add gzip support for login and password files	Jul 1, 2015
libpq-fe.h	initial commit	Apr 24, 2014
ntlm.c	pre-8.0	Apr 27, 2014
ntlm.h	initial commit	Apr 24, 2014
performance.h	more pre-8.0 fixes	Apr 28, 2014
postgres_ext.h	initial commit	Apr 24, 2014
pw-inspector-logo.rc	initial commit	Apr 24, 2014
pw-inspector.1	initial commit	Apr 24, 2014
pw-inspector.c	initial commit	Apr 24, 2014
pw-inspector.ico	initial commit	Apr 24, 2014
rdp.h	closed before merging, sorry, silly me. fix for Android	May 30, 2014
sasl.c	Update from original repository	Jun 25, 2015
sasl.h	initial commit	Apr 24, 2014
xhydra.1	david sadly cant help maintain hydra anymore	Sep 11, 2014

README

```
                                H Y D R A

                    (c) 2001-2014 by van Hauser / THC
                     <vh@thc.org> http://www.thc.org
            many modules were written by David (dot) Maciejak @ gmail (dot) com
                      BFG code by Jan Dlabal <dlabaljan@gmail.com>

                        Licensed under AGPLv3 (see LICENSE file)

                    Please do not use in military or secret service organizations,
                                  or for illegal purposes.



INTRODUCTION
------------
Number one of the biggest security holes are passwords, as every password
security study shows.
This tool is a proof of concept code, to give researchers and security
consultants the possiblity to show how easy it would be to gain unauthorized
access from remote to a system.

THIS TOOL IS FOR LEGAL PURPOSES ONLY!

There are already several login hacker tools available, however none does
either support more than one protocol to attack or support parallized
connects.

It was tested to compile cleanly on Linux, Windows/Cygwin, Solaris,
FreeBSD/OpenBSD, QNX (Blackberry 10) and OSX.

Currently this tool supports the following protocols:
```

Asterisk, AFP, Cisco AAA, Cisco auth, Cisco enable, CVS, Firebird, FTP,
HTTP-FORM-GET, HTTP-FORM-POST, HTTP-GET, HTTP-HEAD, HTTP-PROXY, HTTPS-FORM-GET,
HTTPS-FORM-POST, HTTPS-GET, HTTPS-HEAD, HTTP-Proxy, ICQ, IMAP, IRC, LDAP,
MS-SQL, MYSQL, NCP, NNTP, Oracle Listener, Oracle SID, Oracle, PC-Anywhere,
PCNFS, POP3, POSTGRES, RDP, Rexec, Rlogin, Rsh, SAP/R3, SIP, SMB, SMTP,
SMTP Enum, SNMP v1+v2+v3, SOCKS5, SSH (v1 and v2), SSHKEY, Subversion,
Teamspeak (TS2), Telnet, VMware-Auth, VNC and XMPP.

However the module engine for new services is very easy so it won't take a
long time until even more services are supported.
Your help in writing, enhancing or fixing modules is highly appreciated!! :-)

WHERE TO GET

You can always find the newest release/production version of hydra at its
project page at https://www.thc.org/thc-hydra
If you are interested in the current development state, the public development
repository is at Github:
  svn co https://github.com/vanhauser-thc/thc-hydra
 or
  git clone https://github.com/vanhauser-thc/thc-hydra.git
Use the development version at your own risk. It contains new features and
new bugs. Things might not work!

HOW TO COMPILE

To configure, compile and install hydra, just type:

```
./configure
make
make install
```

If you want the ssh module, you have to setup libssh (not libssh2!) on your
system, get it from http://www.libssh.org, for ssh v1 support you also need

```
to add "-DWITH_SSH1=On" option in the cmake command line.

If you use Ubuntu/Debian, this will install supplementary libraries needed
for a few optional modules:
 apt-get install libssl-dev libssh-dev libidn11-dev libpcre3-dev \
                 libgtk2.0-dev libmysqlclient-dev libpq-dev libsvn-dev \
                 firebird2.1-dev libncp-dev
This enables all optional modules and features with the exception of Oracle,
SAP R/3 and the apple filing protocol - which you will need to download and
install from the vendor's web sites.

For all other Linux derivates and BSD based systems, use the system
software installer and look for similar named libraries like in the
comand above. In all other cases you have to download all source libraries
and compile them manually.



SUPPORTED PLATFORMS
-------------------
All UNIX platforms (linux, *bsd, solaris, etc.)
Mac OS/X
Windows with Cygwin (both IPv4 and IPv6)
Mobile systems based on Linux, Mac OS/X or QNX (e.g. Android, iPhone, Blackberry 10, Zaurus, iPaq)



HOW TO USE
----------
If you just enter "hydra", you will see a short summary of the important
options available.
Type "./hydra -h" to see all available command line options.

Note that NO login/password file is included. Generate them yourself.
A default password list is hoever present, use "dpl4hydra.sh" to generate
a list.

For Linux users, a GTK gui is available, try "./xhydra"
```

```
For the command line usage, the syntax is as follows:
 For attacking one target or a network, you can use the new "://" style:
  hydra [some command line options] PROTOCOL://TARGET:PORT/OPTIONS
 The old mode can be used for these too, and additionally if you want to
 specify your targets from a text file, you *must* use this one:
  hydra [some command line options] [-s port] TARGET PROTOCOL OPTIONS

Via the command line options you specify which logins to try, which passwords,
if SSL should be used, how many parallel tasks to use for attacking, etc.

PROTOCOL is the protocol you want to use for attacking, e.g. ftp, smtp,
http-get or many others are available
TARGET is the target you want to attack
OPTIONS are optional values which are special per PROTOCOL module

FIRST - select your target
 you have three options on how to specify the target you want to attack:
 1. a single target on the command line: just put the IP or DNS address in
 2. a network range on the command line: CIDR specification like "192.168.0.0/24"
 3. a list of hosts in a text file: one line per entry (see below)

SECOND - select your protocol
 Try to avoid telnet, as it is unreliable to detect a correct or false login attempt.
 Use a port scanner to see which protocols are enabled on the target.

THIRD - check if the module has optional parameters
 hydra -U PROTOCOL
 e.g. hydra -U smtp

FOURTH - the destination port
 this is optional! if no port is supplied the default common port for the
 PROTOCOL is used.
 If you specify SSL to use ("-S" option), the SSL common port is used by default.


If you use "://" notation, you must use "[" "]" brackets if you want to supply
IPv6 addresses or CIDR ("192.168.0.0/24") notations to attack:
```

```
  hydra [some command line options] ftp://[192.168.0.0/24]/
  hydra [some command line options] -6 smtp://[2001:db8::1]/NTLM

Note that everything hydra does is IPv4 only!
If you want to attack IPv6 addresses, you must add the "-6" command line option.
All attacks are then IPv6 only!

If you want to supply your targets via a text file, you can not use the ://
notation but use the old style and just supply the protocol (and module options):
  hydra [some command line options] -M targets.txt ftp
You can supply also port for each target entry by adding ":<port>" after a
target entry in the file, e.g.:
  foo.bar.com
  target.com:21
  unusual.port.com:2121
  default.used.here.com
  127.0.0.1
  127.0.0.1:2121

Note that if you want to attach IPv6 targets, you must supply the -6 option
and *must* put IPv6 addresses in brackets in the file(!) like this:
  foo.bar.com
  target.com:21
  [fe80::1%eth0]
  [2001::1]
  [2002::2]:8080
  [2a01:24a:133:0:00:123:ff:1a]


LOGINS AND PASSWORDS
--------------------
You have many options on how to attack with logins and passwords
With -l for login and -p for password you tell hydra that this is the only
login and/or password to try.
With -L for logins and -P for passwords you supply text files with entries.
e.g.:
  hydra -l admin -p password ftp://localhost/
  hydra -L default_logins.txt -p test ftp://localhost/
```

```
  hydra -l admin -P common_passwords.txt ftp://localhost/
  hydra -L logins.txt -P passwords.txt ftp://localhost/
Additionally, you can try passwords based on the login via the "-e" option.
The "-e" option has three parameters:
  s - try the login as password
  n - try an empty password
  r - reverse the login and try it as password
If you want to, e.g. try "try login as password and "empty password", you
specify "-e sn" on the command line.


But there are two more modes for trying passwords than -p/-P:
You can use text file which where a login and password pair is seperated by a colon,
e.g.:
  admin:password
  test:test
  foo:bar
This is a common default account style listing, that is also generated by the
dpl4hydra.sh default account file generator supplied with hydra.
You use such a text file with the -C option - note that in this mode you
can not use -l/-L/-p/-P options (-e nsr however you can).
Example:
  hydra -C default_accounts.txt ftp://localhost/

And finally, there is a bruteforce mode with the -x option (which you can not
use with -p/-P/-C):
  -x minimum_length:maximum_length:charset
the charset definition is 'a' for lowercase letters, 'A' for uppercase letters,
'1' for numbers and for anything else you supply it is their real representation.
Examples:
  -x 1:3:a generate passwords from length 1 to 3 with all lowercase letters
  -x 2:5:/ generate passwords from length 2 to 5 containing only slashes
  -x 5:8:A1 generate passwords from length 5 to 8 with uppercase and numbers
Example:
  hydra -l ftp -x 3:3:a ftp://localhost/
```

```
SPECIAL OPTIONS FOR MODULES
---------------------------
Via the third command line parameter (TARGET SERVICE OPTIONAL) or the -m
commandline option, you can pass one option to a module.
Many modules use this, a few require it!

To see the special option of a module, type:
  hydra -U <module>
e.g.
  ./hydra -U http-post-form

The special options can be passed via the -m parameter, as 3rd command line
option or in the service://target/option format.

Examples (they are all equal):
  ./hydra -l test -p test -m PLAIN 127.0.0.1 imap
  ./hydra -l test -p test 127.0.0.1 imap PLAIN
  ./hydra -l test -p test imap://127.0.0.1/PLAIN



RESTORING AN ABORTED/CRASHED SESSION
------------------------------------
When hydra is aborted with Control-C, killed or crashs, it leavs a
"hydra.restore" file behind which contains all necessary information to
restore the session. This session file is written every 5 minutes.
NOTE: the hydra.restore file can NOT be copied to a different platform (e.g.
from little indian to big indian, or from solaris to aix)



HOW TO SCAN/CRACK OVER A PROXY
------------------------------
The environment variable HYDRA_PROXY_HTTP defines the web proxy (this works
just for the http/www service!).
The following syntax is valid:
  HYDRA_PROXY_HTTP="http://123.45.67.89:8080/"
For all other services, use the HYDRA_PROXY variable to scan/crack
```

```
via by default a web proxy's CONNECT call. It uses the same syntax. eg:
  HYDRA_PROXY=[http|socks4|socks5]://proxy_addr:proxy_port
for example:
  HYDRA_PROXY=http://proxy.anonymizer.com:8000

If you require authentication for the proxy, use the HYDRA_PROXY_AUTH
environment variable:
  HYDRA_PROXY_AUTH="the_login:the_password"



ADDITIONAL HINTS
----------------
* sort your password files by likelihood and use the -u option to find
  passwords much faster!
* uniq your dictionary files! this can save you a lot of time :-)
    cat words.txt | sort | uniq > dictionary.txt
* if you know that the target is using a password policy (allowing users
  only to choose password with a minimum length of 6, containing a least one
  letter and one number, etc. use the tool pw-inspector which comes along
  with the hydra package to reduce the password list:
    cat dictionary.txt | pw-inspector -m 6 -c 2 -n > passlist.txt



SPEED
-----
through the parallizing feature, this password cracker tool can be very
fast, however it depends on the protocol. The fastest are generally POP3
and FTP.
Experiment with the task option (-t) to speed things up! The higher - the
faster ;-) (but too high - and it disables the service)



STATISTICS
----------
Run against a SuSE Linux 7.2 on localhost with a "-C FILE" containing
```

295 entries (294 tries invalid logins, 1 valid). Every test was run three times (only for "1 task" just once), and the average noted down.

	PARALLEL TASKS							
SERVICE	1	4	8	16	32	50	64	100	128
telnet	23:20	5:58	2:58	1:34	1:05	0:33	0:45*	0:25*	0:55*
ftp	45:54	11:51	5:54	3:06	1:25	0:58	0:46	0:29	0:32
pop3	92:10	27:16	13:56	6:42	2:55	1:57	1:24	1:14	0:50
imap	31:05	7:41	3:51	1:58	1:01	0:39	0:32	0:25	0:21

(*)
Note: telnet timings can be VERY different for 64 to 128 tasks! e.g. with 128 tasks, running four times resulted in timings between 28 and 97 seconds! The reason for this is unknown...

guesses per task (rounded up):

295	74	38	19	10	6	5	3	3

guesses possible per connect (depends on the server software and config):

telnet	4
ftp	6
pop3	1
imap	3

BUGS & FEATURES

Hydra:
Email me or David if you find bugs or if you have written a new module.
vh@thc.org (and put "antispam" in the subject line)

You should use PGP to encrypt emails to vh@thc.org :

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v3.3.3 (vh@thc.org)

mQINBFIp+7QBEADQcJctjohuYjBxq7MELAlFDvXRTeIqqh8kqHPOR018xKL09pZT
KiBWFBkU48xlR3EtV5fC1yEt8gDEULe5o0qtK1aFlYBtAWkflVNjDrs+Y2BpjITQ
FnAPHw0SOOT/jfcvmhNOZMzMU8lIubAVC4cVWoSWJbLTv6e0DRIPiYgXNT5Quh6c
vqhnI1C39pEo/W/nh3hSa16oTc5dtTLbi5kEbdzml78TnT0OASmWLI+xtYKnP+5k
Xv4xrXRMVk4L1Bv9WpCY/Jb6J8K8SJYdXPtbaIi4VjgVr5gvg9QC/d/QP2etmw3p
lJ1Ldv63x6nXsxnPq6MSOOw8+QqKc1dAgIA43k6SU4wLq9TB3x0uTKnnB8pA3ACI
zPeRN9LFkr7v1KUMeKKEdu8jUut5iKUJVu63lVYxuM5ODb6Owt3+UXgsSaQLu9nI
DZqnp/M6YTCJTJ+cJANN+uQzESI4Z2m9ITg/U/cuccN/LIDg8/eDXW3VsCqJz8Bf
lBSwMItMhs/Qwzqc1QCKfY3xcNGc4aFlJz4Bq3zSdw3mUjHYJYv1UkKntCtvvTCN
DiomxyBEKB9J7KNsOLI/CSst3MQWSG794r9ZjcfA0EWZ9u6929F2pGDZ3LiS7Jx5
n+gdBDMe0PuuonLIGXzyIuMrkfoBeW/WdnOxh+27eemcdpCb68XtQCw6UQARAQAB
tB52YW4gSGF1c2VyICgyMDEzKSA8dmhAdGhjLm9yZz6JAjkEEwECACMCGwMCHgEC
F4AFAlIp/QcGCwkIAwcCBhUKCQgLAgUWAwIBAAAKCRDI8AEqhCFiv2R9D/9qTCJJ
xCH4BUbWIUhw1zRkn9iCVSwZMmfaAhz5PdVTjeTelimMh5qwK2MNAjpR7vCCd3BH
Z2VLB2Eoz9MOgSCxcMOnCDJjtCdCOeaxiASJt8qLeRMwdMOtznM8MnKCIO8X4oo4
qH8eNj83KgpI50ERBCj/EMsgg07vSyZ9i1UXjFofFnbHRWSW9yZO16qD4F6r4SGz
dsfXARcO3QRI5lbjdGqm+g+HOPj1EFLAOxJAQOygz7ZN5fj+vPp+G/drONxNyVKp
QFtENpvqPdU9CqYh8ssazXTWeBi/TIs0q0EXkzqo7CQjfNb6tlRsg18FxnJDK/ga
V/1umTg41bQuVP9gGmycsiNI8Atr5DWqaF+O4uDmQxcxS0kX2YXQ4CSQJFi0pml5
slAGL8HaAUbV7UnQEqpayPyyTEx1i0wK5ZCHYjLBfJRZCbmHX7SbviSAzKdo5JIl
Atuk+atgW3vC3hDTrBu5qlsFCZvbxS21PJ+9zmK7ySjAEFH/NKFmx4B8kb7rPAOM
0qCTv0pD/e4ogJCxVrqQ2XcCSJWxJL31FNAMnBZpVzidudNURG2v61h3ckkSB/fP
JnkRy/yxYWrdFBYkURImxD8iFD1atj1n3EI5HBL7p/9mHxf1DVJWz7rYQk+3czvs
IhBz7xGBz4nhpCi87VDEYttghYlJanbiRfNh3okCOAQTAQIAIgUCUin7tAIbAwYL
CQgHAwIGFQgCCQoLBBYCAwECHgECF4AACgkQyPABKoQhYr8OIA//cvkhoKay88yS
AjMQypach8C5CvP7eFCT11pkCt1DMAO/8Dt6Y/Ts10dPjohGdIX4PkoLTkQDwBDJ
HoLO75oqj0CYLlqDI4oHgf2uzd0Zv8f/11CQQCtut5oEK72mGNzv3GgVqg60z2KR
2vpxvGQmDwpDOPP620tf/LuRQgBpks7uazcbkAE2Br09YrUQSCBNHy8kirHW5m5C
nupMrcvuFx7mHKW1z3FuhM8ijG7oRmcBWfVoneQgIT3l2WBniXg1mKFhuUSV8Erc
XIcc11qsKshyqh0GWb2JfeXbAcTW8/4IwrCP+VfAyLO9F9khP6SnCmcNF9EVJyR6
Aw+JMNRin7PgvsqbFhpkq9N+gVBAufz3DZoMTEbsMTtW4lYG6HMWhza2+8G9XyaL
ARAWhkNVsmQQ5T6qGkI19thB6E/T6ZorTxqeopNVA7VNK3RVlKpkmUu07w5bTD6V
l3Ti6XfcSQqzt6YX2/WUE8ekEG3rSesuJ5fqjuTnIIOjBxr+pPxkzdoazlu2zJ9F
n24fHvlU20TccEWXteXj9VFzV/zbPEQbEqmE16lV+bO8U7UHqCOdE83OMrbNKszl
7LSCbFhCDtflUsyClBt/OPnlLEHgEE1j9QkqdFFy90l4HqGwKvx7lUFDnuF8LYsb
/hcP4XhqjiGcjTPYBDK254iYrpOSMZSIRgQQEQIABgUCUioGfQAKCRBDlBVOdiii
tuddAJ4zMrge4qzajScIQcXYgIWMXVenCQCfYTNQPGkHVyp3dMhJ0NR21TYoYMC5

```
Ag0EUin7tAEQAK5/AEIBLlA/TTgjUF3im6nu/rkWTM7/gs5H4W0a04kF4UPhaJUR
gCNlDfUnBFA0QD7Jja5LHYgLdoHXiFelPhGrbZel/Sw6sH2gkGCBtFMrVkm3u7tt
x3AZlprqqRH68Y5xTCEjGRncCAmaDgd2apgisJqXpu0dRDroFYpJFNH3vw9N2a62
0ShNakYP4ykVG3jTDC4MSl2q3BO5dzn8GYFHU0CNz6nf3gZR+48BG+zmAT77peTS
+C4Mbd6LmMmB0cuS2kYiFRwE2B69UWguLHjpXFcu9/85JJVCl2CIab7l5hpqGmgw
G/yW8HFK04Yhew7ZJOXJfUYlv1EZzR5bOsZ8Z9inC6hvFmxuCYCFnvkiEI+pOxPA
oeNOkMaT/W4W+au0ZVt3Hx+oD0pkJb5if0jrCaoAD4gpWOte6LZA8mAbKTxkHPBr
rA9/JFis5CVNI688O6eDiJqCCJjPOQA+COJI+0V+tFa6XyHPB4LxA46RxtumUZMC
v/06sDJlXMNpZbSd5Fq95YfZd4l9Vr9VrvKXfbomn+akwUymP8RDyc6Z8BzjF4Y5
02m6Ts0J0MnSYfEDqJPPZbMGB+GAgAqLs7FrZJQzOZTiOXOSIJsKMYsPIDWE8lXv
s77rs0rGvgvQfWzPsJlMIx6ryrMnAsfOkzM2GChGNX9+pABpgOdYII4bABEBAAGJ
Ah8EGAECAAkFAlIp+7QCGwwACgkQyPABKoQhYr+hrg/9Er0+HN78y6UWGFHu/KVK
d8M6ekaqjQndQXmzQaPQwsOHOvWdC+EtBoTdR3VIjAtX96uvzCRV3sb0XPB9S9eP
gRrO/t5+qTVTtjua1zzjZsMOr1SxhBgZ5+0U2aoY1vMhyIjUuwpKKNqj2uf+uj5Y
ZQbCNklghf7EVDHsYQ4goB9gsNT7rnmrzSc6UUuJOYI2jjtHp5BPMBHh2WtUVfYP
8JqDfQ+eJQr5NCFB24xMW8OxMJit3MGckUbcZlUa1wKiTb0b76fOjt0y/+9u1ykd
X+i27DAM6PniFG8BfqPq/E3iU20IZGYtaAFBuhhDWR3vGY4+r3OxdlFAJfBG9XDD
aEDTzv1XF+tEBo69GFaxXZGdk9//7qxcgiya4LL9Kltuvs82+ZzQhC09p8d3YSQN
cfaYObm4EwbINdKP7cr4anGFXvsLC9urhow/RNBLiMbRX/5qBzx2DayXtxEnDlSC
Mh7wCkNDYkSIZOrPVUFOCGxu7lloRgPxEetM5x608HRa3hDHoe5KvUBmmtavB/aR
zlGuZP1S6Y7S13ytiULSzTfUxJmyGYgNo+4ygh0i6Dudf9NLmV+i9aEIbLbd6bni
1B/y8hBSx3SVb4sQVRe3clBkfS1/mYjlldtYjzOwcd02x599KJlcChf8HnWFB7qT
zB3yrr+vYBT0uDWmxwPjiJs=
=ytEf
-----END PGP PUBLIC KEY BLOCK-----
```